LINEAR RECURRENT MODELS FOR ROBUST AND INTERPRETABLE LONG-CONTEXT MODELING

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Computer Science

by

Tushaar Gangavarapu

December 2024

ABSTRACT

Transformers, powered by self-attention, have become foundational across modalities due to their scalability through parallelism. However, many real-world tasks—from byte-level language modeling to retrieval and reasoning—demand processing extended contexts, often spanning tens of thousands of tokens. The quadratic complexity of Transformers during training and linear KV-cache growth at inference makes them computationally prohibitive for such tasks, necessitating the exploration of more efficient architectures. Linear-time recurrent models and hybrid architectures, which combine linear recurrence with local attention, have emerged as promising alternatives to address these challenges.

This thesis investigates two interconnected dimensions of long-context modeling: the development of scalable architectures and the interpretability of their underlying recurrence mechanisms. In the context of scalable architectures, we adapt the Mamba linear recurrent byte-level sequence modeling, tackling the computational challenges posed by extended sequence lengths. By introducing speculative decoding that combines subword drafting with byte-level verification, we demonstrate that token-free models can achieve subword-like inference speedups while maintaining robustness to noise and generalization over long sequences.

On the interpretability front, we focus on hybrid models like RecurrentGemma, where recurrent hidden states serve as memory to bridge local attention windows. Using sparse autoencoders, we extract interpretable features from these recurrent states to explore their role in retaining long-range information.

Through tasks such as long-context passkey retrieval, we reveal how these mechanisms facilitate extended context dependencies and offer insights into the memory of these fixed-memory models.

Through these contributions, this thesis advances the understanding and development of architectures that combine scalability and interpretability for long-context modeling.

BIOGRAPHICAL SKETCH

Tushaar Gangavarapu is a Cornell graduate student who spends more time debugging code than sleeping—and more often than not, the code wins. Despite seven-plus years of formal computer science education (and academic writing), including four spent earning a bachelor's from the National Institute of Technology Karnataka, he still doesn't understand the difference between `\RequirePackage` and `\usepackage`. Tushaar's research focuses on linear-recurrent models, which he believes are the next big thing after Transformers—cool and compact. When he's not deep in code, you can find him by vending machines, with a stash of peanut M&Ms, and debating which algorithm is the best ever created (hint: it's FFT). His MS journey is a thrilling mix of caffeine, curiosity, and the occasional existential crisis about his code.

To my amma, for believing in me, even when I couldn't believe in myself. This thesis is as much yours as it is mine.

ACKNOWLEDGEMENTS

What would Sasha say? "Do you really need this?" "This color doesn't add anything to the figure!" Or the unforgettable, "No math means *no math*!" This work would not have been possible without the unyielding support and incisive guidance of my advisor, Alexander "Sasha" Rush. Sasha's strong opinions were both a source of motivation and, occasionally, fear. His high standards left no room for ambiguity, and I often braced myself when submitting drafts or slides, knowing they would be met with his razor-sharp critiques. He once told me, "I have infinite patience [to deal with you], and I will send this [slide] back until you get it right!" True to his word, every piece of feedback—no matter how daunting—helped refine my work and made me a better researcher. Beyond teaching me technical skills like experiment design, paper writing, and presentations, and guiding me through major research and career decisions, Sasha's most lasting lesson was the importance of asking "*why?*" His insistence on clarity and purpose transformed how I approach research.

This section would be incomplete and insincere without naming Cristian Danescu-Niculescu-Mizil. His support, guidance, and, most importantly, patience with me have been pivotal to my growth as an academic. I've always marveled at how, when I hit a dead end, Cristian would simply say, "Okay, good, let's now do this instead!" and effortlessly turn a dead end into a corner, finding new paths to explore. His ability to see solutions where I saw roadblocks is truly inspiring. I am, however, mildly suspicious that Cristian thinks I'm a robot—like Samantha from *Her*. Perhaps it was the time I entered "NetID password" (literally, as is) when prompted for my NetID password, or how I instinctively align with the model during presentations, saying things like, "*we* do better than humans," as if I weren't one myself. Cristian, thank you for your unwavering

support and for always encouraging me to navigate the corners and edges of research.

One of my greatest fortunes has been working with—or rather, learning from—Lillian Lee. I still remember when she said, "It is fascinating to think that an approximation of such a human and complex phenomenon as understanding language could be encapsulated by something so apparently mathematically clean/cold-blooded as assigning probabilities to strings." That perspective stayed with me—not just for the poetic way it captured the heart of computational linguistics, but for how it reflected her unique ability to blend sharp technical insights with a profound appreciation for the beauty of language. From our conversations, which were always both thought-provoking and deeply grounding, to her constant encouragement, she shaped not only my research but also how I think as a scientist. For that, and so much more, *my debt to you is unbounded*.

Next, I wish to thank the wonderful research collaborators who are a living embodiment of *A Beautiful Mind*: Jonathan Chang, Junxiong Wang, Jing Nathan Yan, Johannes Knittel, Hendrik Strobelt, and Ben Hoover. Among them, there are a few I want to single out. First, Jonathan has been a constant presence from the beginning of my master's journey to the end and has profoundly influenced how I approach problems—from writing code to presenting my results. Whether it was brainstorming cool titles (though, let's be honest, you mostly came up with them while I nervously paced across the room) or navigating countless debugging sessions where I often found myself embarrassed by the simplicity of the issues you spotted, every interaction has been a learning opportunity. Johannes and Hendrik have been close collaborators over the past year. Their endless willingness to entertain my wild ideas—often half-baked,

poorly sketched, and essentially a brain dump of every analysis I could think of—has been both humbling and motivating. I am immensely grateful for their patience, creativity, and generosity in helping turn chaotic sparks into coherent insights. Junxiong was instrumental in the development of this research, including many of the results presented in this thesis. I deeply admire his infinite patience through my ad-hoc scheduling, as well as his ability to entertain any crazy solution I proposed and code it to life. Through those midnight debugging sessions, lonely Ithaca winter dinners, and sunny Pennsylvania poster presentations, we've struggled and grown together throughout this journey—emerging as independent researchers. Here's to the future of linear recurrent models!

Beyond collaborations, I want to thank the members of Team Zissou: Son Tran, Nicholas Chernogor, Yilun Hua, Dave Jung, Vivian Nguyen, Tony Wang, and Luke Tao, as well as Sasha's lab: Woojeong Kim, Celine Lee, and Jack Morris. Just spending time with you and having discussions—both technical and non-technical—has been invaluable. I also want to thank all the TAs of CS 4740: Darren Key, Kai Horstmann, Aaishi Uppuluri, Dave Jung, Lionel Tan, Logan Kraver, and Pun Chaixanien. You all are amazing! It was incredible to have a team that believed in my vision for the two assignments we created. Whether it was a late-night (8pm-forever) meeting or me pinging you on Slack at 12am, I always knew I could count on you to back me up. Our names are etched on the assignments for years to come!

During my three years in Ithaca, these people taught me that there was life outside of Gates and Rhodes Halls. You brought joy, balance, and perspective to my time here, and even taught me how to cross a road (or at least tried!), for which I am deeply grateful. And to all the friends who made these years bearable: Yann Hicke, Barry Wang, Sumanth Aluri, Lu-

cas Matos Molter, Divya Darshni Suresh, Nikitha Sharma, Shaden Shaar, Kai Horstmann, Sean Zhang, Pamraat Parmar, Sidhee Hande, Shubhangi Sinha, Madhav Aggarwal, Ramya Lakshminarasimhan, Sukruth Gowda, Sandhya Pillai, Nathaniel Navarro, and Abhishek Masand—thank you. Also, Katherine O'Connor, thanks for putting up with me through endless "it's only 1am, there's four more hours for me to work!" sessions and Peanut (not peanut butter) M&M dinners. I'm sure I'm forgetting many, but this thesis is, in part, thanks to each and every one of you. As Gloria Pritchett from Modern Family so elegantly puts it: "You be the wind in his back, not the spit in his face." You were all the wind in my back, helping carry me over the finish line.

Parts of the research described in this thesis were supported by NSF IIS-1901030 and NSF CAREER 2037519.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Sequence-to-sequence models, which transform sequential data into rich semantic representations capturing the structure of the input, have achieved significant advancements across various complex data modalities. These include natural language processing, time-series analysis, genomics, speech, and audio (Sutskever et al., 2014; Oord et al., 2016; Ismail Fawaz et al., 2019; Brown et al., 2020). Their applicability has even extended to non-sequential modalities, such as images, by recasting them as sequences (Dosovitskiy et al., 2021). These models rely on foundational mechanisms like convolution (in convolutional neural networks), recurrence (in recurrent neural networks), or attention (Vaswani et al., 2017) (in Transformers).

While all these model classes have significantly advanced machine learning, Transformers, with attention at their core, stand out as particularly impactful today. Recent advancements in modern hardware, access to massive computational resources, and large-scale training data have enabled Transformer-based models to excel as versatile multi-taskers (Brown et al., 2020; Rae et al., 2022; Hoffmann et al., 2022; Team et al., 2024a; Touvron et al., 2023; Team et al., 2024b; Team, 2024; Dubey et al., 2024; OpenAI et al., 2024). For instance, GPT-4 (OpenAI et al., 2024) can debug code and explain the fixes in natural language (Bubeck et al., 2023), Stable Diffusion models (Rombach et al., 2022) generate realistic images from text prompts, and AlphaFold (Jumper et al., 2021) predicts 3D protein structures with remarkable accuracy.

The tremendous success of Transformers can largely be attributed to their scalability, both in model and data. This is evident in their progression from

BERT (Devlin et al., 2018), trained on 3.3B tokens in 2018, to Llama-405B (Dubey et al., 2024), trained on 15T tokens in 2023. Their ability to model long-range dependencies efficiently and their inherent parallelizability have driven these advancements. However, as Transformers have grown larger and deeper, equipping them with longer-context capabilities is challenging. This limitation arises from the quadratic time and memory complexity of global attention with respect to sequence length, which constrains their effectiveness in handling extremely long sequences. Furthermore, the linear growth of key-value (KV) cache with sequence length makes Transformers slow at inference.

To address these limitations, linear recurrence models[1]—including state space models (Gu et al., 2022a; Mehta et al., 2023; Wang et al., 2023; Smith et al., 2023; Fu et al., 2022; Poli et al., 2023; Gu and Dao, 2024; Dao and Gu, 2024), linear recurrent models (Peng et al., 2023; Orvieto et al., 2023; De et al., 2024; Botev et al., 2024; Beck et al., 2024; Feng et al., 2024; Peng et al., 2024), and linear attention models (Katharopoulos et al., 2020; Yang et al., 2024; Qin et al., 2024; Katsch, 2024)—have emerged as compelling alternatives. By maintaining and evolving a (large) hidden state with fixed memory, these models enable efficient linear-time sequence modeling, making them well-suited for tasks requiring long-context processing without the quadratic overhead of attention-based methods. A key innovation in modern recurrent models is the introduction of an input-dependent state-transition matrix and gating the hidden state, thus allowing the model to selectively filter out irrelevant context while retaining and processing critical information indefinitely (Cirone et al., 2024). This design marks a departure from traditional recurrence mechanisms like LSTMs (Hochreiter and Schmidhuber,

[1]While subtle distinctions exist between various model families within linear recurrence frameworks, we use the term "linear recurrence models" to refer broadly to those that employ the linear evolution of hidden state memory.

1997) and GRUs (Cho et al., 2014), which rely on non-linear recurrence of the hidden state. In contrast, these newer models maintain linearity in the hidden state while incorporating potentially non-linear transformations of the input. This approach not only boosts expressive power but also supports efficient parallelization on GPUs during training, through parallel scans (Martin and Cundy, 2018; Smith et al., 2023). These attention-free models have demonstrated competitive performance with Transformers across a variety of modalities and at scale, suggesting their potential as a promising avenue in advancing sequence modeling (De et al., 2024).

Given these advances, the large fixed-state memory of recurrent models, which remains independent of context length, is of particular interest to us. This fixed-state memory allows recurrent models to handle long-context tasks with linear complexity, making them a natural fit for scenarios requiring efficient long-context processing and retrieval.

1.1 Dissertation overview

This thesis focuses on two primary objectives:

- In Chapter 2, we demonstrate the viability of token-free language modeling using linear recurrent models. A key insight in this work is that, whether processing subwords or bytes, the underlying recurrent model must compress a similar amount of information into its hidden memory, making such models particularly well-suited for this task.
- In Chapter 3, we investigate the role of recurrence in long-context retrieval. Building on recent advances in mechanistic interpretability, we utilize

sparse autoencoders to extract orthogonal features that correspond to the recurrent memory. We then identify key features responsible for guiding long-context retrieval, providing a deeper understanding of how recurrence facilitates information retention over long sequences.

The overarching theme of this thesis revolves around harnessing and understanding linear recurrent models to efficiently handle long-context tasks, particularly in language modeling. Through both establishing their capabilities—such as in byte-level modeling—and exploring the mechanistic underpinnings of recurrence, this work seeks to outline the strengths and limitations of these models. We envision this thesis to be a useful tool in analyzing, and perhaps developing, future architectural advances.

1.2 Our contributions

1.2.1 Byte-level language modeling

Byte-level modeling offers robustness to text corruptions and advantages in multilingual and multimodal contexts, but it is computationally expensive due to the $4\times$ longer sequence lengths. Without representational compression, this challenge is particularly pronounced in Transformers, which scale quadratically at training and linearly at inference with respect to sequence length. To address this, Chapter 2 introduces MambaByte, a token-free selective state space model, an application of the Mamba recurrence (Gu and Dao, 2024) to byte sequences. While Mamba alleviates significant modeling and efficiency bottlenecks during training by evolving a large hidden state memory independent of context length,

inference remains slow as it still requires generating one byte at a time. To improve inference efficiency, Chapter 2 also presents an adaptation of speculative decoding for byte-level models, enabling subword-like speedups at inference.

We conduct careful benchmarking in both compute-matched and parameter-matched settings against several state-of-the-art Transformer subword and byte models, as well as evaluate other token-free capabilities of MambaByte. While our results suggest a promising future for token-free modeling, they also highlight the strengths of recurrent models in tasks that require efficient information compression and long-context modeling.

1.2.2 Role of recurrence in long-context retrieval

While Chapter 2 establishes the impressive capabilities of MambaByte in modeling long contexts, in Chapter 3 we attempt to understand the internal mechanisms of these linear recurrent networks that enable the model to utilize information from arbitrary locations within the input. Specifically, we focus on the passkey retrieval abilities of RecurrentGemma-9B (Botev et al., 2024; De et al., 2024), an efficient linear recurrent model with remarkable long-context capabilities. Motivated by the question: what is contained in the hidden state when the model can or cannot retrieve the passkey?—we employ sparse autoencoders to extract and analyze features from the recurrent hidden states.

Chapter 3 identifies several key features that guide the model's associative recall. These features offer insights into how recurrence facilitates the retention and retrieval of distant information. While further research is needed to assess the generalizability of our findings, we believe that uncovering these features is

an important step toward understanding the limitations of recurrent models in long-context retrieval. Future designs of recurrent models should incorporate strategies that promote the retention of these features over extended context spans, thereby improving performance in long-context retrieval scenarios.

It is worth noting that some recent works have already explored the expressiveness—or lack thereof—of linear recurrent models in studying specific toy tasks (Jelassi et al., 2024) and in examining their limitations within the framework of formal language theory (Merrill et al., 2024). Compared to these studies, which outline interesting failure cases, we focus on understanding how these large-scale models succeed in practical scenarios, such as associative recall in long-context retrieval.

CHAPTER 2

MAMBABYTE: TOKEN-FREE SELECTIVE STATE SPACE MODEL

[This chapter is adapted from joint work with Junxiong Wang, Jing Nathan Yan, and Alexander M. Rush entitled "MambaByte: Token-free selective state space model" (Wang et al., 2024a).]

This chapter introduces MambaByte, a token-free state space model (SSM) for modeling long byte-sequences. MambaByte outperforms other byte-level models over several datasets and shows competitive results with subword Transformers while being significantly robust to text corruptions, thus serving as a promising tokenization alternative. Due to their recurrent nature, SSMs enable significantly faster text generation to Transformer models. We further improve the generation efficiency through speculative decoding using subword drafting and show MambaByte to achieve to achieve a decoding efficiency similar to the subword Mamba, making byte models practical. Our findings establish the possibility of token-free language modeling in future large models.

2.1 Introduction

When defining a language model, a base tokenization is typically used—either words (Bengio et al., 2000), subwords (Schuster and Nakajima, 2012; Sennrich et al., 2015; Wu et al., 2016; Wang et al., 2020), or characters (Gao et al., 2020a). Of these, subword tokenization has been the most popular choice, as it achieves a natural compromise between training efficiency and the ability to handle out-of-vocabulary words. However, several works, e.g., Xue et al. (2022), have noted issues with subword tokenizers, such as a lack of robustness to typos, spelling

and capitalization variations, and morphological changes.

Modeling byte sequences, i.e., mapping from raw data to predictions without any intermediate tokenization, offers an alternative approach with less inductive bias (Choe et al., 2019; Al-Rfou et al., 2019; Clark et al., 2022; Tay et al., 2022; Xue et al., 2022; Yu et al., 2023). Compared to subword models, byte-level language models can generalize more easily across orthographic and morphological variants. Of course, modeling text as bytes means that the resultant sequences are significantly longer than their subword counterparts. This change pushes the modeling and efficiency issues upstream into the architecture itself.

These issues are particularly pronounced for autoregressive Transformers (Vaswani et al., 2017), which dominate language modeling (Brown et al., 2020; Touvron et al., 2023). Due to the quadratic nature of attention, Transformer efficiency scales poorly for long (byte) sequences (Zhang et al., 2022). Researchers have *compressed* the internal Transformer representation to work with long sequences, for instance, developing length-aware modeling approaches (Dai et al., 2020; Nawrot et al., 2023), where groups of tokens are merged within the intermediate layers. The MegaByte Transformer (Yu et al., 2023) is of particular relevance, which uses compression in the form of fixed-size patches of bytes as a subword analog in combination with a byte-level decoder. These methods lower computational costs[1] but change the modeling behavior to match the data.

In this work, we propose MambaByte, a byte-level language model without representational compression. The model is an application of the recently introduced Mamba architecture (Gu and Dao, 2024). Mamba builds off the approach pioneered by state space models (SSMs) (Gu et al., 2022a; Gupta et al., 2022;

[1]However, our experiments (see Figure 2.1) indicate that patching can also lower the model performance compared to the standard Transformer.

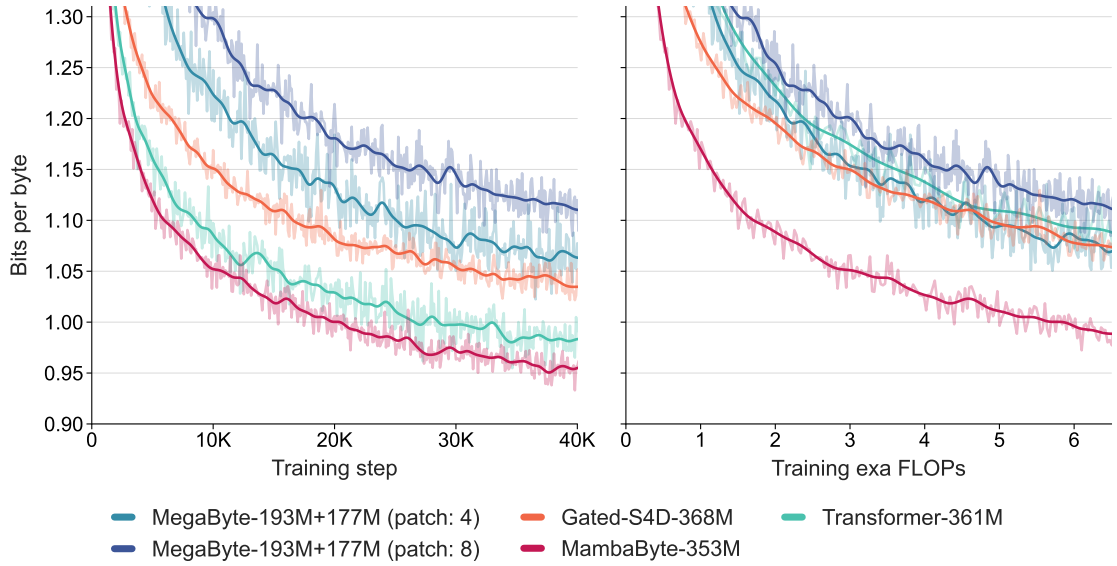


Figure 2.1: **Benchmarking byte-level models with a fixed parameter budget.** Language modeling results on PG19 ($8,192$ consecutive bytes), comparing the standard Transformer (Vaswani et al., 2017; Su et al., 2021), MegaByte Transformer (Yu et al., 2023), gated diagonalized S4 (Mehta et al., 2023), and MambaByte. (Left) Model loss over training step. (Right) FLOP-normalized training cost. MambaByte reaches Transformer loss in less than one-third of the compute budget.

Gu et al., 2022b; Smith et al., 2023; Fu et al., 2022) by introducing a selection mechanism that has been shown to be nearly as effective as Transformers for discrete data. Our key observation is that, unlike Transformers, Mamba has a (large) fixed-sized memory state that is independent of context length, roughly analogous to a large recurrent neural network hidden state. This naturally removes a major modeling and efficiency issue for byte-level language modeling without requiring specialized architectures such as global patching.

Even with effective training, byte-level models still suffer from the challenge of efficient decoding, as generating one character at a time requires running the language model in serial one byte at a time. To improve the inference efficiency, we propose an adaptation of speculative decoding (Leviathan et al., 2023; Chen

et al., 2023; Xia et al., 2023) to byte-level models. The approach uses a fast subword model for autoregressive drafting, followed by byte-level verification. While this approach could be applied to any byte-level model, it is particularly efficient for SSM-style models since the byte-level verification step can use the same parallel scan code path that makes these models efficient to train.

Experiments compare MambaByte to Transformers, SSMs, and MegaByte (patching) architectures in a fixed parameter and fixed compute setting on several long-form language modeling datasets. Figure 2.1 summarizes our main findings. Compared to byte-level Transformers, MambaByte achieves better performance faster and is significantly more compute-efficient. We also compare MambaByte with tokenized subword baselines using Transformers and SSMs, and find that MambaByte is competitive in loss while also demonstrating improved robustness in handling subword noise, such as input text corruptions. Through our speculative subword drafting and byte-level verification approach, we show that MambaByte can be run as fast as the subword Mamba for text generation. We believe these results validate the potential for tokenizer-free models as a practical alternative to subword Transformers for language modeling.

2.2 Background: Mamba state space model

Method: Selective SSMs. SSMs model the evolution of a hidden state across time through a first-order differential equation. Linear time-invariant SSMs (Gu et al., 2022a; Gupta et al., 2022; Gu et al., 2022b; Smith et al., 2023) have shown promising results in deep learning across several modalities. However, Gu and Dao (2024) have recently argued that the constant dynamics of these

approaches lack input-dependent context *selection* in the hidden state, which may be necessary for tasks such as language modeling. To this end, they define the time-varying continuous state dynamics for a given input $x(t) \in \mathbb{R}$, hidden state $h(t) \in \mathbb{R}^n$, and output $y(t) \in \mathbb{R}$ at time t as:

$$\frac{\mathrm{d}h(t)}{\mathrm{d}t} = \mathrm{A}h(t) + \mathrm{B}(t)x(t); \quad y(t) = \mathrm{C}(t)h(t), \tag{2.1}$$

which is parameterized by a diagonal time-invariant system matrix $\mathrm{A} \in \mathbb{R}^{n\times n}$ and time-dependent input and output matrices $\mathrm{B}(t) \in \mathbb{R}^{n\times 1}$ and $\mathrm{C}(t) \in \mathbb{R}^{1\times n}$.

To model discrete-time sequences, the continuous-time dynamics in (2.1) must be approximated through discretization. This results in a discrete-time hidden state recurrence with new matrices at each timestep, $\overline{\mathrm{A}}$, $\overline{\mathrm{B}}$, and $\overline{\mathrm{C}}$, such that

$$h[k] = \overline{\mathrm{A}}[k]h[k-1] + \overline{\mathrm{B}}[k]x[k]; \quad y[k] = \overline{\mathrm{C}}[k]h[k]. \tag{2.2}$$

Observe that (2.2) resembles a linear version of a recurrent neural network and can be applied in this recurrent form during language model generation. The discretization requires a timestep, $\Delta[k]$, for each input position, corresponding to treating $x[k] = x\left(t_k\right)$ for $t_k = \sum_{j=1}^{k} \Delta[j]$. The discrete-time matrices $\overline{\mathrm{A}}$, $\overline{\mathrm{B}}$, and $\overline{\mathrm{C}}$ can then be computed from $\Delta[k]$.

Architecture: Mamba. In Mamba, the SSM terms are input-selective, i.e., B, C, and Δ are defined as functions of the input $x[k] \in \mathbb{R}^d$:

$$\Delta[k] = \mathrm{softplus}(W_\Delta(W_R x[k])); \quad \mathrm{B}(t_k) = W_\mathrm{B} x[k], \tag{2.3}$$

where $W_\mathrm{B} \in \mathbb{R}^{n\times d}$ (C is similarly defined), $W_\Delta \in \mathbb{R}^{d\times r}$ and $W_R \in \mathbb{R}^{r\times d}$ (for some $r \ll d$) are learnable weights, and softplus ensures positivity. Note that the

SSM parameters A, B, and C are identical for each input dimension d, but the timesteps Δ are distinct; this results in a hidden state size of $n \times d$ per timestep k. (See Appendix A.5 for an illustration of how Mamba models discrete sequences and other specifics on discretization and selectivity.)

Mamba embeds this SSM layer into a full neural network language model. Specifically, the model utilizes a stack of gated layers inspired by the previous gated SSM (Mehta et al., 2023). Figure A.1 (right) in Appendix A.2 shows the Mamba architecture combining the SSM layer with a gated neural network.

Implementation: Parallel scans for linear recurrences. At training time, we have access to the entire sequence x, allowing us to compute the linear recurrence more efficiently. Smith et al. (2023) demonstrated the use of work-efficient parallel scans (Blelloch, 1990) for efficiently computing the sequential recurrence in linear SSMs. For Mamba, we first map the recurrence to a sequence of L tuples, with $e_k = (A_k, b_k) \coloneqq (\overline{\mathrm{A}}[k], \overline{\mathrm{B}}[k]x[k])$, then define an associative operator $\bullet$ such that $e_j \bullet e_k = (A_k A_j, A_k b_j + b_k)$. Finally, we apply a parallel scan to compute the sequence $[(\overline{\mathrm{A}}[1], h[1]), (\overline{\mathrm{A}}[2]\overline{\mathrm{A}}[1], h[2]), \ldots]$. In general, this requires $\mathcal{O}(T_\bullet \log_2(L))$ time, using $L/2$ processors, where $T_\bullet$ is the cost of a matrix-matrix multiplication. Noting $\overline{\mathrm{A}}$ to be a diagonal matrix, the linear recurrence can be computed parallelly in $\mathcal{O}(n \log_2(L))$ time and $\mathcal{O}(nL)$ space. A parallel scan with a diagonal matrix is also efficient in operation, requiring $\mathcal{O}(nL)$ FLOPs. (Appendix A.4 details parallel scans.)

2.3 Methods

2.3.1 Modeling long byte-sequences

MambaByte is an application of the Mamba architecture to byte-level language modeling. Our main observation is that unlike Transformers, whose memory scales linearly in sequence length, Mamba maintains a large fixed-size memory state, which makes it suitable for direct byte-level modeling. Formally, an m-layer Mamba model, each with a hidden state $h(t) \in \mathbb{R}^{n_{\text{state}} \times d}$, efficiently maintains and evolves a memory of $m \times n_{\text{state}} \times d$ floats. Noting that the Mamba hidden state memory size is independent of input context length, L_{ctx}, processing subword sequences or byte sequences requires the underlying model to compress roughly L_{ctx} bytes in its fixed hidden state memory, irrespective of the input representation. In all but extreme cases, $m \times n_{\text{state}} \times d \gg L_{\text{ctx}}$, leaving enough space of a hidden state $h(t)$ to encode L_{ctx} information. Therefore, if Mamba can be used for tokenized models, MambaByte should enable modeling byte-level sequences without the need for length-compression trade-offs (Dai et al., 2020; Nawrot et al., 2023; Yu et al., 2023).

Utilizing a fixed-sized memory representation may also help avoid quadratic dependencies and improve generalization. While Transformers are designed to capture long-range dependencies, researchers have noted that the sheer number of potential interactions in a long byte-level sequence can dilute the model's focus, making it challenging to capture crucial dependencies amid a vast number of less relevant ones (Tworkowski et al., 2024). Bytes level information is much more granular, thus necessitating the model to learn from a much larger context to make meaningful predictions.

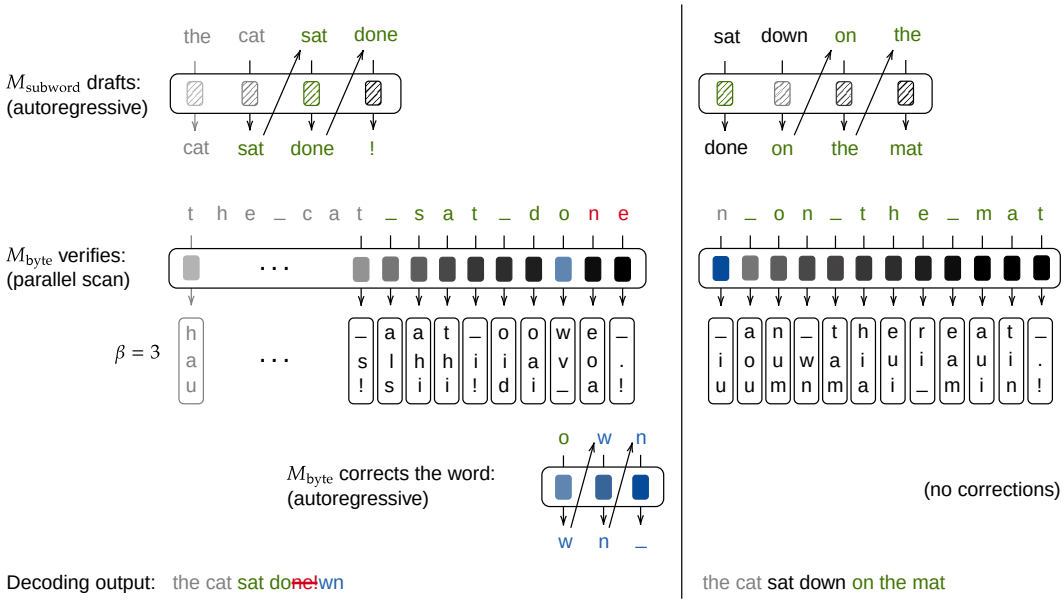


Figure 2.2: **Speculative decoding through subword drafting and byte-level verification.** The green subwords are suggestions made by the smaller subword (Mamba) model M_{subword}, whose associated bytes fell in the top-β autoregressive candidates of the byte-level verifier (MambaByte) model M_{byte}. The red and blue bytes are the rejections and corresponding corrections made by the verifier model. (Two steps shown using the prompt: "the cat".)

Finally, training Mamba for long byte-sequences has an inherent computation benefit at scale. The computational cost for Mamba at training is $\mathcal{O}(L_{\text{ctx}})$, while even compressed models such as MegaByte (Yu et al., 2023) have a complexity of $\mathcal{O}(L_{\text{ctx}}^2/p^2 + L_{\text{ctx}}p)$ for a patch size p. Even with a large patch size of $L_{\text{ctx}}^{1/3}$, the resulting complexity is $\mathcal{O}(L_{\text{ctx}}^{4/3})$.

2.3.2 Speculative decoding through subword drafting

While MambaByte is computationally efficient at training, it encounters challenges in decoding, primarily because each byte is processed sequentially. To mitigate this sequential bottleneck, we propose a adaptation of speculative decoding through subword drafting and byte-level verification. Our observation is that

most inference steps do not require the granularity of byte-level decoding and can benefit from faster subword drafting. Consequently, we can train token-free models, which are known to be robust to noise, and simulate subword-model-like generation, which is significantly faster. We decompose every decoding iteration into two steps: *draft* using a smaller subword (Mamba) model, then *verify and correct* using a larger byte-level (MambaByte) model, as illustrated in Figure 2.2.

The subword Mamba model, M_{subword}, drafts m subwords autoregressively while recording the associated hidden states at each timestep. The drafted subwords are converted to bytes, fed to the byte-level MambaByte model, M_{byte}, and verified using a parallel scan. We then find the bifurcation byte position c, the furthest position in the byte sequence verified to be in the top-β autoregressive candidates of M_{byte}. We also find the subword bifurcation position associated with c, i.e., the largest position of the subword whose bytes are all verified to be correct. Drafted bytes after position c are discarded. Noting that the drafter is tokenized, while the verifier is token-free, we cannot just correct for b_{c+1}, i.e., one byte after the bifurcation position, and continue drafting—this causes issues with drafting, especially if the tokenizer cannot find the newly updated partial subword in its pre-trained vocabulary. To avoid the possibility of the corrected partial subword being marked as out-of-vocabulary, we use the verifier model to generate bytes autoregressively until a boundary byte (e.g., space) is generated. The final decoded tokens include the verified drafted subwords and the corrected subword generated by the byte-level model. We cache the final hidden state from the MambaByte verifier and the bifurcation hidden state from the subword Mamba model for the next iteration. For completeness, we provide the algorithm for speculative decoding through subword drafting in Appendix A.7.

To enable resuming during the parallel scan, we extended the fast CUDA kernel from Mamba (Gu and Dao, 2024), allowing verification to restart from the mismatched position instead of beginning from the start.

2.4 Experimental setup

Expt	Models	FLOPs per train byte
Medium-scale	MegaByte-758M+262M : MambaByte-353M	$1.02 : 1$
Large-scale	MegaByte-1.3B+350M : MambaByte-972M	$0.54 : 1$
	MegaByte-1.3B+218M : MambaByte-972M	$0.40 : 1$

Table 2.1: **Relative training FLOPs by model size.** MegaByte models use a patch size of 8.

Our experiments compare MambaByte to a range of other tokenizer-based and token-free Transformers and SSMs. All our models employ the same training recipes. We utilize a set of diverse long-form text datasets: PG19 (Rae et al., 2020), Stories (Trinh and Le, 2018), Books (Gao et al., 2020b), ArXiv (Gao et al., 2020b), and Code (Gao et al., 2020b). We consider models of different sizes: for MambaByte, this is indicated by the number of parameters in the model; for MegaByte, which is the primary baseline used, size is indicated by the number of parameters in the patched model and the unpatched generation head. Dataset sizes and average document lengths are included in Appendix A.1; model details are given in Appendix A.2.

Performance comparison across architectures requires care. To this end, we consider two settings: compute-matched and parameter-matched. This setup is

necessary as the default MegaByte Transformer employs a global module that works with $8\times$-patched representations of the input, thus using $8\times$ fewer feed-forward FLOPs per byte than a raw Transformer, while having significantly more parameters. Table 2.1 shows the MegaByte and MambaByte model sizes employed in our experiments. The (forward pass) FLOPs computation for various model architectures and the associated hyperparameters employed are detailed in Appendix A.2.

All MambaByte models were trained using the open-source Mamba code base.[2] At training, we shuffle the documents and use contiguous sequences of $8,192$ bytes (one per document), starting from a random position. We enable mixed precision training using BF16 for training efficiency at scale. The optimizer, learning rate scheduler, and other training details are specified in Appendix A.3.

2.5 Results

2.5.1 Language modeling performance

Table 2.2 shows language modeling performance in bits per byte (BPB) across each dataset. For this experiment, the MegaByte-758M+262M and MambaByte models use the same number of FLOPs per byte (see Table 2.1). We observe MambaByte to outperform MegaByte consistently across all datasets. Furthermore, MambaByte outperforms MegaByte with $0.63\times$ less compute and training data. Additionally, MambaByte-353M also outperforms byte-level Transformer and PerceiverAR.

[2] https://github.com/state-spaces/mamba.

Byte-level model	Context	Bytes trained	Test BPB ↓				
			PG19	Stories	Books	ArXiv	Code
Transformer-320M	1,024	80B	1.057	1.064	1.097	0.816	0.575
PerceiverAR-248M	8,192	80B	1.104	1.070	1.104	0.791	0.546
MegaByte-758M+262M	8,192	80B	1.000	0.978	1.007	0.678	0.411
MambaByte-353M	8,192	30B*	**0.930**	**0.908**	**0.966**	**0.663**	**0.396**

Table 2.2: **Medium-scale token-free experiments.** MegaByte-758M+262M (patch: 8) and MambaByte-353M use the same FLOPs per byte. (The BPB for Transformer, PerceiverAR, and MegaByte are from Yu et al. (2023).)

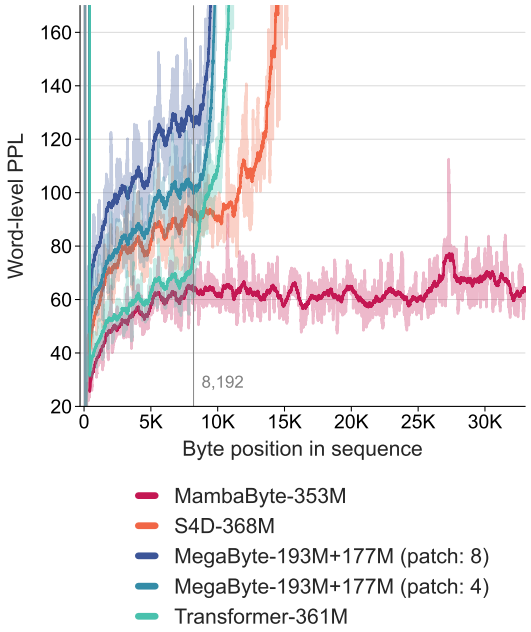


Figure 2.3: **Length extrapolation.** All models are trained with 8,192-long byte sequences. MambaByte can extrapolate to much longer sequences without performance degradation.

Figure 2.1 further explores this relationship by looking at models with the same number of parameters. The graphs indicate that for MegaByte models of the same parameter size, models with less input patching perform better, but when compute-normalized, they perform similarly. In fact, a full-length Transformer, while slow in an absolute sense, also performs similarly to the MegaByte model when compute-normalized. In contrast, switching to the Mamba architecture

significantly improves both the compute usage and the model performance.

Following these findings, Table 2.3 compares a larger version of these models on the PG19 dataset, both with and without tokenization. For this experiment, we compare MambaByte-972M with MegaByte-1.3B+350M and other byte-level models, as well as several state-of-the-art subword models. (The conversion from BPB and subword-level perplexity to word-level perplexity (PPL) is detailed in Appendix A.6). We find that MambaByte-972M, even just trained for 150B bytes, outperforms all the byte-level models and achieves competitive performance with subword models.

Figure 2.3 records another interesting aspect of MambaByte: its ability to extrapolate to significantly longer sequences (at least $4\times$ longer than the training length) compared to other byte-level Transformer and SSM baselines, suggesting that MambaByte can effectively refine the recurrent hidden state for significantly longer sequences. As expected, limited by the position embeddings, Transformer models don't extrapolate beyond the training length.

2.5.2 Token-free capabilities

To control for the benefits of the Mamba architecture, we retrained a subword Mamba-1.03B model in a compute-matched setting (see Table 2.3). Interestingly, the (subword) Mamba and MambaByte perform similarly at the same parameter size and training compute. As previously mentioned, these models effectively have the same memory capacity despite significant differences in the input sequence length. We also find that Mamba achieves near-optimal performance more efficiently than MambaByte, though not $4\times$ faster as expected, but $2.2\times$

(#Layers) Model	Vocab	Effective ctx (in bytes)	Effective bytes trained	Val PPL↓	Test PPL↓
(36) Transformer-XL (Rae et al., 2020)	32K	$2,048/4,096$	400B	45.5	36.3
(36) Compressive (Rae et al., 2020)	32K	$2,048/2\times2,048$	400B	43.4	33.6
(22) Routing-490M (Roy et al., 2021)	82K	$32,768$	330B	–	33.2
(60) PerceiverAR-975M (Hawthorne et al., 2022)	32K	$8,192$	1.68T	45.9	28.9
(24) Block-Recurrent-1.3B (Hutchins et al., 2022)	32K	$4,096$/recur.	–	–	**26.5**
(48) Mamba-1.03B	32K	$8,192$	600B*	40.1	33.9
(–) Transformer-320M (Yu et al., 2023)	256	$8,192$	400B	81.6	69.4
(–) PerceiverAR-248M (Yu et al., 2023)	256	$8,192$	400B	119.1	88.8
(24+24) MegaByte-1.3B+350M (Yu et al., 2023)	256	$8,192$/patch: 8	400B	42.8	36.4
(48) MambaByte-972M	256	$8,192$	150B*	**39.6**	33.0

Table 2.3: **Large-scale experiment on PG19.** The observed BPB scores are converted to word-level PPL for comparison with past works.
(Top.) Comparison with subword models.
(Bottom.) All the byte-level models are compute-matched.
Mamba-1.03B and MambaByte-972M are evaluated using $4\times$ longer context and a sliding window of $16,384$ bytes. MambaByte-972M significantly outperforms other byte-level models and is competitive with state-of-the-art subword models.
(Accompanying citation indicates the work from which the corresponding result was taken; fields marked – are unknown.)

faster. Furthermore, the perplexity for Mamba-1.03B does not improve significantly beyond 150B training bytes, consistent with the observations made by Rae et al. (2020). Given the similar performance of Mamba and MambaByte, we can further explore downstream capabilities.

Modeling longer contexts. From Figure 2.4, we note that Mamba and MambaByte show impressive extrapolation capabilities for sequences up to $64\times$ longer than the training length. We hypothesize that MambaByte shows slightly better length extrapolation than the subword Mamba because MambaByte models $4\times$ longer sequences at training despite both models processing the same effective number of bytes per training sequence.

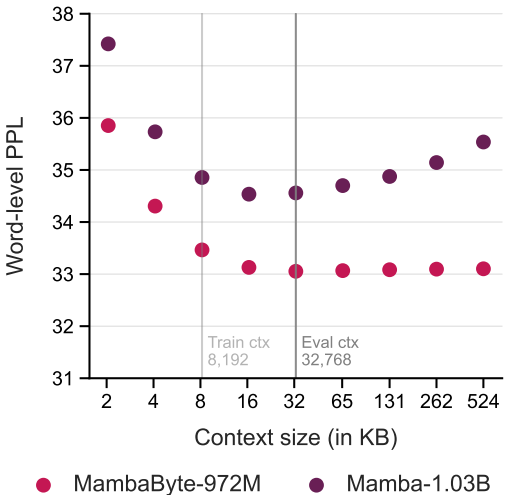


Figure 2.4: **Long context experiment.** Length extrapolation using a sliding window of $L_{\text{ctx}}/2$.

	Proba	PPL ↓	
		Mamba	MambaByte
Drop	0.05	+16.9	**+8.5**
	0.3	+213.2	**+31.7**
Repeat	0.05	+6.3	**+6.2**
	0.3	+28.4	**+26.6**
Antspeak		+58300.0	**+28.3**
Uppercase	0.05	+5.4	**+1.6**
	0.3	+18.3	**+5.5**
Random case		+20.8	**+7.7**
Swap	0.05	+29.0	**+9.3**
	0.3	+630.6	**+28.7**

Table 2.4: **Synthetic noise experiments.** Degradation of Mamba-1.03B and MambaByte-972M under varied noise settings.

Synthetic noise experiments. We employ the synthetic noise benchmark from Xue et al. (2022) to test model robustness; additional details about the noise settings are noted in Appendix A.8. We process the input text in the PG19 test set into chunks of 100 space-separated words and inject noise into every odd-indexed chunk while retaining the text in the even-indexed chunk unaltered. Table 2.4 shows the degradation of word-level PPL with noise injection, measured on even-indexed chunks. We observe that Mamba performance degrades significantly in the presence of noise compared to MambaByte across all noise conditions, indicating that tokenized vocabulary fundamentally limits subword models. This effect is pronounced in specific noise settings such as antspeak (every character is capitalized and padded with spaces) and character swapping (consecutive bytes are swapped). Our findings align with those observed by Xue et al. (2022) in that byte-level models are significantly more resilient to accidental and adversarial spelling errors than subword models.

Model	Bytes trained	Context	Test BPB ↓	Generation time (s) ↓
Transformer-350M	80B	1,024	1.064	132
MegaByte-1.3B+218M	80B	8,192	0.991	93
MegaByte-1.3B+218M	–	8,192	–	265
MambaByte-972M	75B*	8,192	**0.883**	**29**
w/ sliding window (2× bytes)			**0.863**	58
MambaByte-1.6B	–	8,192	–	36

Table 2.5: **Generation speed benchmarking.** Speed to generate 8,192 bytes; fields marked – are unknown. (Upper) The BPB on PG19 and generation time for the Transformer and MegaByte are taken from Yu et al. (2023). (Lower) MegaByte and MambaByte run on the same hardware; we use the available open-source MegaByte implementation here.

2.5.3 Generation efficiency

Autoregressive inference in Transformer models requires caching the entire context, which can significantly affect the generation speed. MambaByte does not suffer from this bottleneck as it maintains a single hidden state per layer that evolves with time, enabling constant time per generation step. Table 2.5 compares the text generation speeds of MambaByte-972M and MambaByte-1.6B with MegaByte-1.3B+350M on an A100 80GB PCIe GPU. While MegaByte significantly reduces the generation cost through patching, we observe MambaByte to be 2.6× faster in a parameter-matched setting due to its use of recurrent generation. Appendix A.9 includes more information about the generation process.

Generation via subword speculation. Table 2.6 shows the inference speedup using speculative decoding through subword drafting, averaged across 100 prompts generated using common phrases in the PG19 dataset. We use a Mamba-110M model as the drafter, and the subwords are generated using greedy decoding. We observe that through speculative subword drafting, MambaByte can

Model	Context	Relative speedup ↑	Log-odds ratio ↑
MambaByte-972M	8,192	1×	1.0
Mamba-1.03B	2,048	2.8×	0.10
MambaByte-972M w/ Mamba-110M speculation	8,192	**2.6×**	**0.89**

Table 2.6: **Generation speed with subword speculation.** Empirical results for speeding up inference from the MambaByte-972M model. Speed was measured in generating 8,192 bytes; the drafter drafts three subwords per iteration and the verifier accepts if the drafted bytes were in its top-3 candidates.

achieve a decoding speed nearing that of the subword Mamba. Furthermore, to assess the faithfulness of our speculative decoding approach, we use a greedy-decoded MambaByte-972M generation as the reference candidate for a given prompt. We report the ratio of the log-likelihood of generating the reference candidate to the log-likelihood of MambaByte generating the speculative-decoded sequence, which is averaged across all prompts. From Table 2.6, we observe our speculative decoding approach to be more faithful to MambaByte-972M than the subword Mamba-1.03B.

2.6 Further related work

Token-free language models. Tokenization has been fundamental to language modeling and vital in enhancing model efficiency and understanding. Several algorithms have been proposed to address tokenization issues, including sizeable vocabulary size and handling out-of-vocabulary tokens: Byte-Pair Encoding (Sennrich et al., 2015), WordPiece (Schuster and Nakajima, 2012; Devlin et al., 2018), and SentencePiece (Kudo and Richardson, 2018). The recent shift towards character (Tay et al., 2022; Ma et al., 2020; Mielke and Eisner, 2019) and byte-level

(Yu et al., 2023; Xue et al., 2022; Belouadi and Eger, 2022) modeling aims to achieve token-free preprocessing, thereby facilitating improved model adaptability and domain transferability in language modeling and multilingual processing.

Attention-free models. Attention-free models offer enhanced computational and memory efficiency and are increasingly adapted for several language processing tasks, including autoregressive language modeling. Models such as S4 (Gu et al., 2022a) and its subsequent variants (Gupta et al., 2022; Gu et al., 2022b) have demonstrated promising outcomes in subword-level language modeling. Gated SSM architectures such as GSS (Mehta et al., 2023) and BiGS Wang et al. (2023) incorporated a gating mechanism into SSMs for (bidirectional) language modeling. The recently introduced Mamba model (Gu and Dao, 2024) posits that the unchanging dynamics of these methods fail to incorporate input-specific context selection within the hidden state, which might be crucial for tasks like language modeling. Mamba has been shown to outperform Transformers across model sizes and at scale. Alternatively, several other sub-quadratic model architectures (Yang et al., 2024; De et al., 2024; Arora et al., 2023, 2024; Fu et al., 2024a) have also been proposed. Beyond language modeling, SSMs and Mamba have been applied in other modalities, including images (Yan et al., 2024), audio (Goel et al., 2022), and bioinformatics (Schiff et al., 2024).

Speculative decoding for fast inference. Speculative decoding (Spector and Re, 2023; Leviathan et al., 2023; Chen et al., 2023; Xia et al., 2023) has emerged as a promising approach to accelerate the inference of large language models, specifically Transformers. The core idea is to use a smaller draft model to speculatively generate candidate tokens, which the larger target model then verifies.

Leviathan et al. (2023); Chen et al. (2023) proposed a rejection sampling scheme to improve the inference quality. Spector and Re (2023) restructured the candidate tokens into a tree to enable more efficient verification. Additional approaches also investigated trained draft models (Bhendawade et al., 2024; Chen et al., 2023; Liu et al., 2023) and training-free draft models (He et al., 2023; Yang et al., 2023; Fu et al., 2024b). While previous methods employ drafter and verifier models with the same underlying tokenization scheme, this paper proposes using a smaller subword Mamba model as the speculative drafter and a larger MambaByte as the byte-level verifier.

2.7 Conclusion

In this chapter, we introduced MambaByte, a token-free SSM for modeling long byte-sequences. MambaByte outperforms other byte-level models over several datasets and shows competitive results with subword Transformers while being significantly robust to text corruptions, thus serving as a promising tokenization alternative. Due to their recurrent nature, SSMs enable significantly faster text generation to Transformer models. We further improve the generation efficiency through speculative decoding using subword drafting and show MambaByte to achieve to achieve a decoding efficiency similar to the subword Mamba, making byte models practical. Our findings establish the possibility of token-free language modeling in future large models.

CHAPTER 3

MECHANISTIC INSIGHTS INTO RECURRENT MEMORY FOR LONG-CONTEXT MODELING

[This chapter is based on ongoing joint work with Johannes Knittel, Hendrik Strobelt, and Alexander M. Rush.]

In Chapter 2, we investigated the extrapolation capabilities of MambaByte, demonstrating its ability to predict tokens in contexts at least $64\times$ longer than those seen during training. This highlights its exceptional capacity for long-context modeling, which is critical for tasks like repository-level code understanding (Bairi et al., 2024), autonomous agents (Weng, 2023), and long-history dialog modeling (Mazumder and Liu, 2024). In this chapter, we exclusively focus on understanding the internal mechanisms that enable such capabilities. Specifically, we aim to dissect the role of the recurrent memory in facilitating long-context retrieval.

3.1 Introduction

The ability to process and generate text based on rich contextual information is crucial for several reasons. Longer contexts provide models with more information to condition their outputs, resulting in more accurate, contextually relevant, and up-to-date responses. Moreover, long-context modeling enhances in-context learning by allowing the incorporation of more examples or instructions within the input (Chevalier et al., 2023; Agarwal et al., 2024; Lee et al., 2023). To explore these capabilities at scale, we study RecurrentGemma-9B, a high-performing linear recurrent model designed for long-sequence tasks (Botev et al., 2024). Our

investigation focuses on the passkey retrieval task (Mohtashami and Jaggi, 2023; Samuel, 2024), in which the model must identify and retrieve a passkey embedded within a long context. This task serves as a foundational benchmark for advanced long-context applications that interleave retrieval and reasoning in a multi-step fashion (Kuratov et al., 2024).

In this chapter, we are motivated by the specific question of what the recurrent memory encodes during successful or failed passkey retrievals. To this end, we experiment with k-sparse autoencoders (Makhzani and Frey, 2014; Gao et al., 2024) to analyze features within RecurrentGemma hidden states. Inspired by prior work demonstrating that sparse autoencoders can uncover causally relevant and interpretable features (Bricken et al., 2023; Cunningham et al., 2023; Marks et al., 2024; Lieberum et al., 2024), we aim to identify features that enable or guide recall beyond the local attention window. Understanding how certain features impact retrieval performance may reveal key limitations of recurrent models in handling long-context tasks.

Our experiments show promising results. Despite not explicitly training the sparse autoencoder to model recurrence, we identify several spanning features, with some extending over three times the length of the local attention window. These features seem to capture long-range dependencies within the hidden states. Furthermore, we pinpoint specific features that play a crucial role in successful passkey retrieval, providing insight into how the model's memory supports recall. The discovery of these "retrieval" features provides valuable insight into the limitations of recurrent models for long-context retrieval. Our findings could guide the design of future recurrent architectures by emphasizing the promotion of such features, potentially through architectural adjustments

or targeted training strategies that enhance the model's ability to retain and leverage long-range dependencies for more effective retrieval and recall tasks.

3.2 Background

3.2.1 RecurrentGemma

Architecture. RecurrentGemma-9B is a medium-scale hybrid model that combines local sliding-window attention of $2,048$ tokens with gated linear recurrence. This architecture effectively balances the strengths of both components: local attention excels at modeling the recent past, while the recurrent layers propagate information across longer sequences. In contrast to standard Transformers with global attention, where the KV cache scales with sequence length, RecurrentGemma maintains fixed state sizes.

Since we are only interested in the recurrent memory of RecurrentGemma, we provide a technical overview of the real-gated linear recurrent unit (RG-LRU), which serves as the token-mixing component of the model. For additional details on the model's specifics, we refer the reader to De et al. (2024). At timestep k, the recurrence in an RG-LRU unit for an input $x[k] \in \mathbb{R}^d$ and $h[k-1] \in \mathbb{R}^d$ is formulated as:

$$h[k] = a[k] \odot h[k-1] + \sqrt{1 - a[k]^2} \odot (i[k] \odot x[k]),$$
$$y[k] = h[k],$$

where the gates are defined as:

$$r[k] = \sigma(W_a x[k]); \quad i[k] = \sigma(W_x x[k]); \quad a[k] = \exp(-cr[k] \odot \text{softplus}(\Lambda)).$$

Here, c is a scalar-valued constant set to 8, σ denotes the sigmoid function, and $\Lambda \in \mathbb{R}^d$ is a learnable vector representing the eigenvalues of the state-transition matrix. The weight matrices W_a and W_x are block-diagonal, each containing n blocks of size $\mathbb{R}^{d/n \times d/n}$. The recurrence gate, $r[k]$, allows the model to effectively discard the input at timestep k, while preserving all information from the previous history, $h[k-1]$. Such gating allows the model to achieve super-exponential memory by reducing the influence of uninformative inputs.

Long-context retrieval abilities. De et al. (2024) demonstrate that hybrid models trained on synthetic copy tasks can achieve retrieval and copying capabilities well-beyond the local attention window. Since RecurrentGemma was trained on sequences of $8,192$ tokens—$4\times$ longer than the length of the local attention window—we expect long-context reasoning abilities to emerge naturally in the model, without requiring explicit fine-tuning on specialized (synthetic) tasks.

3.2.2 TopK sparse autoencoders

Recent studies have shown that a significant portion of language model activations can be expressed as sparse, linear combinations of vectors, where each vector represents a meaningful feature (Mikolov et al., 2013; Olah et al., 2020; Elhage et al., 2022; Gurnee et al., 2023; Nanda et al., 2023; Park et al., 2024). Sparse autoencoders have proven to be an effective unsupervised method for discovering these feature vectors, with prior research demonstrating their ability to uncover interpretable features (Bricken et al., 2023; Cunningham et al., 2023; Gao et al., 2024; Marks et al., 2024; Adly Templeton et al., 2024; Lieberum et al., 2024).

Building on the approach by Gao et al. (2024), we use a k-sparse autoencoder (Makhzani and Frey, 2014) to model hidden state activations. The TopK activation function retains only the k largest latent values, setting the rest to zero. This approach offers a Pareto improvement on the sparsity-reconstruction frontier compared to ReLU-based autoencoders that rely on an L_1 penalty to enforce sparsity.

For an input activation, x, the encoder and decoder reconstruct x as follows:

$$z = \mathrm{TopK}(W_{\text{enc}}(x - b_{\text{pre}}) + b_{\text{enc}}),$$
$$\hat{x} = W_{\text{dec}} + b_{\text{pre}},$$

where the encoder matrix $W_{\text{enc}} \in \mathbb{R}^{n \times d}$, latent bias $b_{\text{enc}} \in \mathbb{R}^n$, decoder weight matrix $W_{\text{dec}} \in \mathbb{R}^{d \times n}$, and pre bias $b_{\text{pre}} \in \mathbb{R}^d$. The columns of W_{dec}, denoted f_j for $j = 1, \ldots, n$, represent the dictionary of features into which the sparse autoencoder decomposes x.

3.2.3 Passkey retrieval task

To understand the long-context behavior of RecurrentGemma, we employ the passkey retrieval task, where the model is required to retrieve and output a specific passkey embedded within the input tokens. A six-digit passkey (ranging from $100,000$ to $999,999$) is inserted at varying depths, both within and outside the local attention window, across multiple trials. A generation is deemed correct if it matches the exact passkey, establishing a trivial baseline accuracy of $1/900,000$. Figure 3.1 illustrates an example, highlighting key regions of interest. (The specific prompts used for the task are provided in Appendix B.1.) To ensure that any correct answer is genuinely retrieved from the context, we confirm the

```
Some magic number is hidden in the following text.
Make sure to memorize it.
I will quiz you about the magic number afterwards.
Once upon a time ...
The magic number is 888888.
... live happily ever after.
Q: What is the magic number in the provided text?
A: The magic number in the provided text is
```

```
888888.
```

Figure 3.1: **Passkey retrieval task.** This task examines how the model retrieves a passkey embedded within a long context. The instruction tokens, highlighted in `blue`, guide the model to memorize the passkey. The passkey itself is revealed in the sentence highlighted in `yellow`. The question, within the local attention window, is shown in `green`, prompting the model to recall the passkey. Finally, the model's response, shown in `red`, displays the retrieved passkey.

passkey to be irrelevant to the given long context. This guarantees that, when a correct answer is generated, it is copied from the context rather than drawn from the model's internal knowledge.

Easy passkey task. We also introduce an adaptation of the standard passkey task, where the passkey is a single token from the model's vocabulary. For simplicity, we set this passkey to a single-digit magic number (from 0 to 9).[1] The trivial baseline accuracy for this task is $1/10$, suggesting that the model might simply guess the number to achieve a correct result. However, when considering aggregate performance for all digits, this probability drops to $(1/10)^{10}$, which is extremely low. That said, this is still an easier task compared to the previous one, as the model only needs to generate a single token.

[1]We also experimented with using a single magic *word* instead of a number. However, retrieval performance for magic words was significantly worse than for magic numbers, even when the magic word was within the local attention window. We do not explore this further, but speculate that this may be due to prompt sensitivity.

3.3 Methods

3.3.1 Establishing long-context retrieval abilities

We first evaluate the long-context retrieval capabilities of RecurrentGemma on the passkey task and the easy passkey task, focusing on the model's performance when the passkey is placed beyond the local attention window of $2,048$ tokens. In the easy passkey task, we further explore whether prior knowledge that the passkey is a single digit improves the model's retrieval accuracy. These experiments aim to identify the model's strengths and limitations, pinpointing specific scenarios of passkey depth and context length to guide subsequent analysis.

For both tasks, explicit instructions are provided, directing the model to memorize the passkey and indicating that it will be queried later. We do not evaluate scenarios where the passkey is introduced without such instructions, as recurrent models naturally prioritize storing information deemed relevant. Without explicit guidance to retain an otherwise irrelevant passkey, the model is likely to allocate memory to other contextual information.

3.3.2 Analyzing sparse autoencoder features

In cases where the model fails to retrieve the passkey, we aim to answer a key question: Is the magic number unavailable by the time it reaches the local attention window because the recurrent mechanism fails to retain that information? Or does the recurrent state effectively pass the information, but the local attention

mechanism struggles to utilize it, being distracted by other competing tokens (e.g., numbers) within the window?

To investigate this, we analyze features extracted by a k-sparse autoencoder trained on the recurrent hidden states. The goal is to identify the specific features that facilitate successful recall of the passkey beyond the local attention window.

Spanning features in recurrent memory. We adopt a generalized feature-centric approach, focusing on analyzing the top-1 feature associated with all the training tokens. We treat these features as binary indicators—either present or not—disregarding the activation values. This simplification allows us to concentrate on the occurrence and continuity of features rather than their activation values.

A significant aspect of our analysis is the identification of spanning features, i.e., features that remain active across multiple consecutive tokens. These are of particular interest because they may hold the key to capturing long-context dependencies. Specifically, we seek spanning features that originate before the local attention window and extend into it, as such features could play a critical role in bridging the gap between long-context information in the recurrent state and the model's local attention mechanism. While interpretability of these features remains an important consideration, our immediate focus is on uncovering whether such spanning features exist and understanding their potential connection to long-context recall.

It is important to emphasize that the autoencoder training process does not explicitly account for the sequential nature of data, treating each hidden state activation vector independently. If spanning features emerge, they likely reflect

the ability of recurrent hidden states to implicitly encode past context, suggesting that RecurrentGemma's recurrence mechanism inherently structures the hidden states to support such features, even without explicit constraints.

We also explore whether specific tokens promote the emergence of spanning features. For example, a token like "$", which marks the start of LaTeXmath code, could trigger a feature that remains active until the end of the code sequence. This exploration aligns with our passkey tasks, where we aim to identify features triggered by the "passkey" (the key information) that remain active until the needle is retrieved, or at least for a significant portion of the subsequent context. Since our goal is to establish evidence of such behavior, we manually select tokens for investigation. However, we note that this process can be easily automated to identify tokens that promote such features.

Identifying features for needle retrieval. Evidence of spanning features and tokens that promote such features suggests that the sparse autoencoder latents can capture aspects of recurrent memory capable of retaining and transmitting information across long contexts. However, this does not directly demonstrate that needle information is being retained or propagated across the context. To address this, we shift to a more prompt-centric analysis, revisiting the *easy* passkey task, which focuses on tracking features associated with a single-token passkey. This shift enables us to go beyond analyzing just the top feature, as our focus is no longer on global feature behavior across all contexts. This is crucial because the top-1 feature might be more closely tied to the specific properties of the token rather than to long-context memory. Consequently, for all experiments in this analysis, we expand our investigation to the top-128 features.

To keep the analysis manageable, we focus on the following features of interest:

- Features appearing in both the passkey sentence and the response: Intuitively, we expect these features to be associated with the passkey value, or that the passkey appears after the mention of "magic number".
- Features appearing in the passkey sentence and the first token within the local attention window: Analyzing these features helps verify whether the passkey information is successfully passed into the attention window. This analysis is made possible by the nature of the easy passkey task, which involves a single-token passkey. As a result, the sliding attention window remains stationary, allowing us to focus on the first token within the window.
- Features appearing in both the passkey sentence, the first in-attention-context token, and the response.

This tiered approach is crucial because the "circuit" responsible for retaining the needle information across long contexts may involve multiple interacting features. Simply considering the common features between the prompt, the needle, the attention context, and the response could result in a limited set of features that span the entire context, potentially overlooking the specific behaviors that drive successful retrieval. By breaking down the analysis into distinct steps, we can gain a deeper understanding of how different features contribute to successful long-context retrieval. We analyze these common features independently from the recall task to identify those that may signal the presence of the passkey, exhibit long-span behavior, or, ideally, selectively span to effectively transmit the needle information across long contexts.

Next, we specifically analyze the behavior of these features in successful vs. unsuccessful retrievals. Additionally, we look for any other features that exhibit distinct behavior in these two scenarios, potentially revealing additional factors that contribute to retrieval success or failure. To narrow the focus, we limit our investigation to features that commonly activate between the passkey sentence and the first in-attention-context token. This is based on a simplistic, though not entirely unreasonable, assumption that, once the recurrence has successfully passed the information into the local attention window, the recurrence has essentially succeeded.

3.4 Experimental setup

3.4.1 Passkey retrieval task

All passkey retrieval tasks follow a consistent template: instruction tokens, followed by filler context tokens, the passkey sentence (inserted at various depths), additional filler context tokens, and the final question, which is placed within the local attention window (as illustrated in Figure 3.1). We use Paul Graham essays as filler context, following the setup in (Hsieh et al., 2024).

For the passkey task, we perform greedy sampling of 20 tokens from RecurrentGemma, and for the easy passkey task, we sample 5 tokens. We evaluate RecurrentGemma on the passkey task across 50 random trials. For the easy passkey task, we conduct 10 trials, one for each digit in the passkey.

3.4.2 Sparse autoencoder

Dataset and modeling specifics. To enable hardware-efficient training, we adapt the fast Triton kernels for sparse matrix multiplications from the OpenAI sparse autoencoder open-source codebase[2]. We train a 131K-latent ($32\times$ the recurrent dimension) k-sparse autoencoder with $k = 32$ using layer-30 RG-LRU activations (before gating with the GeLU branch, see Figure 3.2) from RecurrentGemma-9B, generated using 339M tokens from The MiniPile dataset (Kaddour, 2023), a clean, deduplicated subset of The Pile corpus (Gao et al., 2020b). Note that this differs from most previous works (Engels et al., 2024; Gao et al., 2024; Adly Templeton et al., 2024; EleutherAI, 2024), which focus on autoencoders trained on residual stream activations. The choice of layer-30 is based on Gao et al. (2024, Appendix F.1), where the authors suggest that using a layer between the $3/4$-th and $5/6$-th depth captures many features without being specialized for next-token prediction. To enable greater parallelism, we train with a large batch size of $32,768$, keeping all other hyperparameters as specified in (Gao et al., 2024).

Training loss. The training loss follows the formulation in Gao et al. (2024), consisting of two components: the reconstruction error as mean-squared loss and auxiliary loss that models the reconstruction error using top-k_{aux} (typically top-512) dead latents. A latent is considered "dead" if it has not activated for the past 10M tokens. Let z' be the reconstruction using the top-k_{aux} dead latents,

[2] https://github.com/openai/sparse_autoencoder

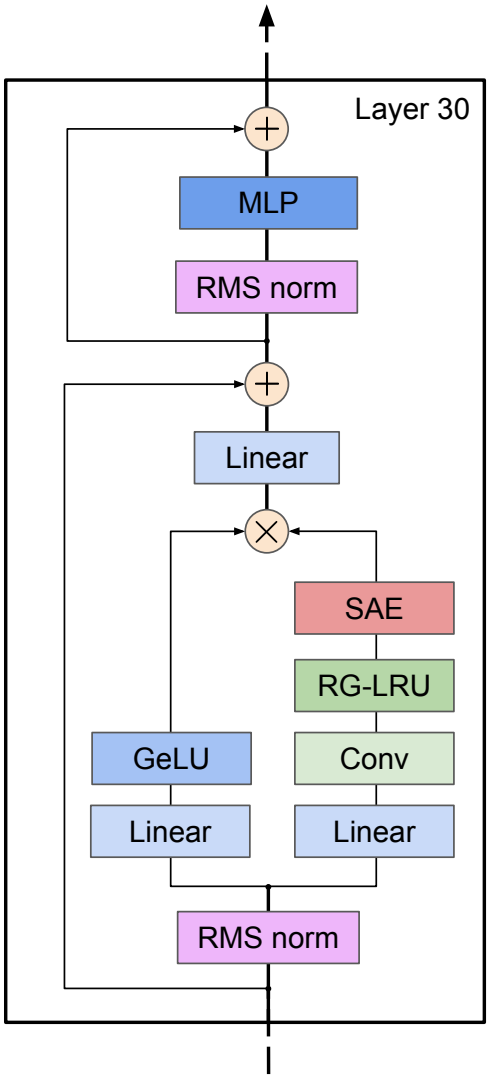


Figure 3.2: **Location of the sparse autoencoder.** The autoencoder is applied to the stream *before* gating with GeLU. This choice is motivated by our focus on analyzing the capabilities of the (gated) linear recurrence in facilitating long-context modeling.

then, the training loss is defined as:

$$\mathcal{L} = \underbrace{\left\|x - \hat{x}\right\|_2^2}_{\mathcal{L}_{\text{mse}}} + \alpha \underbrace{\left(\left\|(x - \hat{x}) - W_{\text{dec}} z'\right\|_2^2\right)}_{\mathcal{L}_{\text{aux}}},$$

for some constant α, typically $1/32$.

In line with Gao et al. (2024), we initialize W_{dec} as W_{enc}^T to help reduce the number of dead latents. Consequently, only about 2% of the latents (i.e., $2,688$

latents) were found to be dead in our trained autoencoder.

Infrastructure: Storing features. To facilitate scalable and generalizable feature analysis beyond manual inspection, we extracted the top-1 feature for all 339 million training tokens. However, this process revealed significant bottlenecks: storing features as compressed pickle files and performing linear scans were both slow and resource-intensive, making traditional methods impractical for such a large dataset. To overcome these challenges, we used DuckDB (Raasveldt and Mühleisen, 2019) to store all extracted features in a flat database structure with indexing on feature indices for fast querying. This approach reduced disk space usage and enabled efficient analyses with minimal RAM and memory requirements, allowing us to explore features at scale systematically.

3.5 Results

3.5.1 Long-context retrieval performance

Figure 3.3 illustrates the retrieval performance of RecurrentGemma-9B on the passkey task. The model performs well within the local attention window but shows a significant decline in retrieval success when the needle is placed beyond this window, even within the training length of $8,192$ tokens. We hypothesize that this limitation arises because the model lacked the need for robust retrieval capabilities during training. This could explain why toy models explicitly trained for copy tasks, such as those in De et al. (2024), perform better. Alternatively, it may stem from the limitations of the underlying linear recurrence, which

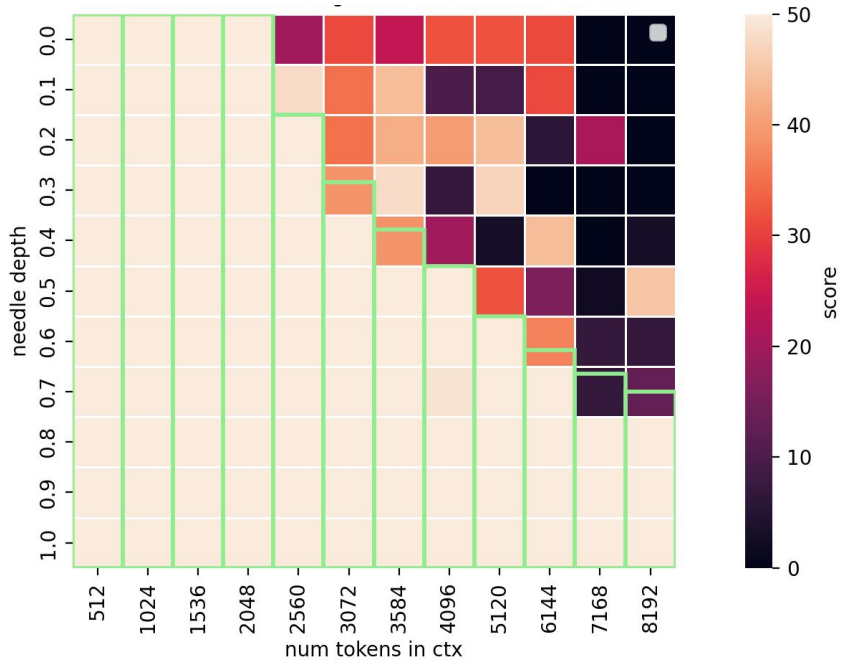
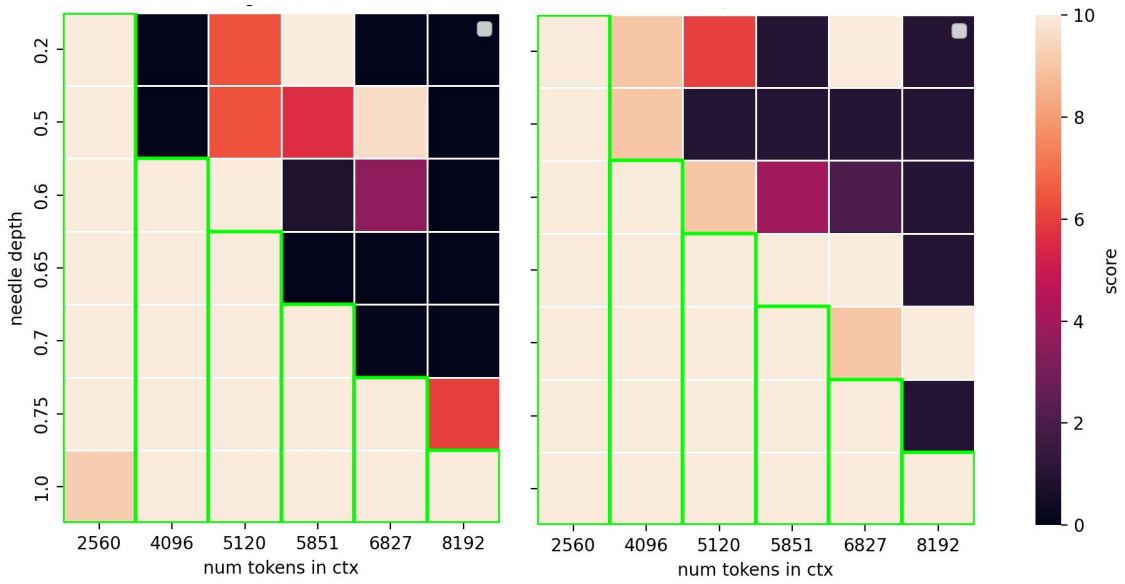


Figure 3.3: **Long-context passkey retrieval evaluation.** We evaluate the retrieval of six-digit passkeys (spanning more than one token) embedded at varying depths within the document across 50 randomized trials.
The portions of the heatmap highlighted within the green boxes represent instances where the passkey is located within the local attention window. Notably, unlike toy models (see De et al. (2024, Section 6.2)), RecurrentGemma's performance significantly degrades when retrieving passkeys located outside the local attention window. This degradation becomes more pronounced as the sequence length increases and the passkey is positioned farther from the local attention window.

struggles to retain the needle for really long contexts.

Figure 3.4 further investigates this behavior on the easy passkey task. As expected, the model retrieves the needle accurately when it is within the local attention window. However, even when the prompt explicitly mentions a single-digit magic number, it still fails to retrieve the needle effectively beyond the local window.



(a) Retrieval using the same prompt as the passkey task, without mentioning that the magic number is a single digit.

(b) Retrieval was performed using a modified passkey prompt, explicitly mentioning that the magic number is a single digit.

Figure 3.4: **Long-context easy passkey retrieval evaluation.** We evaluate the retrieval of a single-digit passkey (spanning exactly one token) across 10 trials, one for each digit. (Left.) When using the standard passkey prompt for this easier task, we observed retrieval errors where the model incorrectly generated multiple digits. (Right.) To simplify the task further, we explicitly note that the magic number is a single digit.

3.5.2 Features promoting long-context retrieval

Does the sparse autoencoder uncover spanning features? Our investigation revealed that approximately 5% of the total features ($6,673$ features) were spanning features, collectively capturing over 23M spans. Many of these features were interpretable: for example, `sae.32x/45941` detected the phrase " videos from our team", while `sae.32x/40432` consistently activated for "featured_button_text}}". Other features captured broader semantic patterns, such as `sae.32x/5103`, which identified measurements in recipe contexts, as shown in Figure 3.5. On average, span lengths were about 13.06 tokens. Notably, `sae.32x/97553` was the longest spanning feature, extending continuously

```
oven to 400 degrees.
2. In a large bowl combine dry ingredients. Add butter,
Directions
In a medium bowl, whisk together chili sauce, fish
Instructions:
1. In a small bowl, dissolve yeast in warm water
1 teaspoon salt
Directions
In a large bowl, cream together butter and sugar. Beat in
oven to 350F. In a medium bowl, combine lime and orange
Instructions
In a large saucepan over medium-high heat, warm the oil.
Heat 1 Tablespoon oil over in a large saute pan.
Add onions, red peppers,
In a large bowl, sift together flour, cornstarch, baking
```

Figure 3.5: **Feature that measures quantities in recipes.** We identify the tokens with the highest activation for the feature `sae.32x/5103`, then randomly sample and display a subset along with a few surrounding tokens for context. Observe that the feature activates for "large saucepan," but not for "medium-high heat."

over $8,170$ LaTeXtokens.

Next, we investigate specific tokens that promote spanning features, focusing on two particular tokens: "." and "$". For the "." token, we observed that the next token is often associated with feature `sae.32x/44363`, which can be interpreted as a start-of-sentence feature (see Figure 3.6.) When this feature is active, it indicates the beginning of a new sentence. However, we caution against making inferences about the start of a sentence based solely on the absence of this feature, as this could be misleading. Next, we examine the "$" token. We find that the two features most frequently associated with the "$" token are also associated with the token that follows it. Feature `sae.32x/97553` is associated with the "$" token in LaTeX code and spans the longest token span. The second most frequent feature, `sae.32x/24863`, is triggered when the "$" symbol is

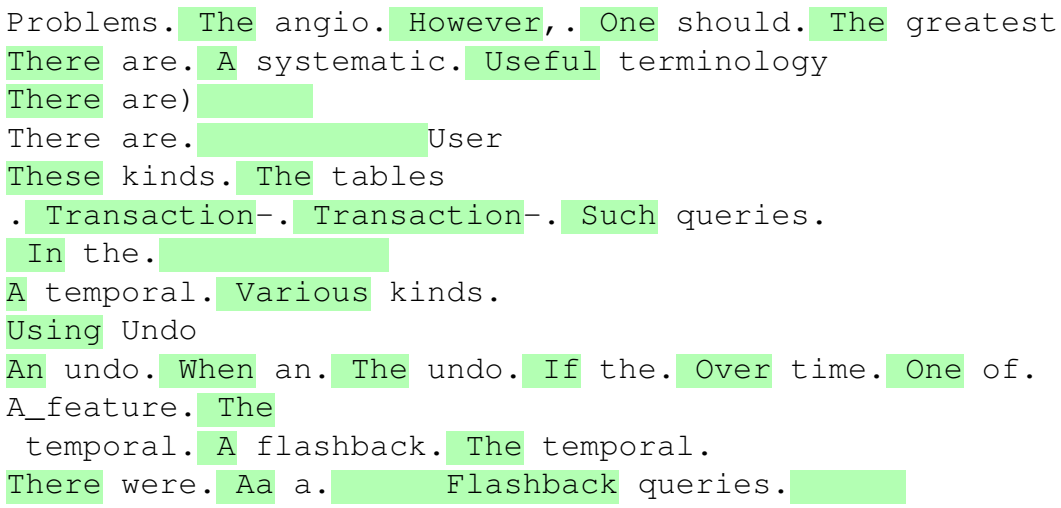


Figure 3.6: **Feature `sae.32x/44363` indicates the start of a sentence.** Note that this does not mean this feature is triggered every time a new sentence begins, but rather that when this feature is activated, it indicates the start of a sentence.

used in code, and remains active for a significantly long span. These observations suggest that the "$" token may promote spanning features, triggering them to remain active over an extended context.

From these analyses, it is evident that some of the features identified by the sparse autoencoder can span remarkably long context lengths, often extending well beyond the boundaries of the local attention window. Furthermore, specific tokens, such as "." and "$", appear to play a role in triggering feature spanning, activating features that persist over extended contexts.

Can we identify features responsible for needle retrieval? Now, we turn our attention to investigating specific features that promote or guide passkey recall. (An analysis comparing raw hidden memory activations in this context is presented in Appendix B.2.) Among the features analyzed, the following stand out as particularly significant:

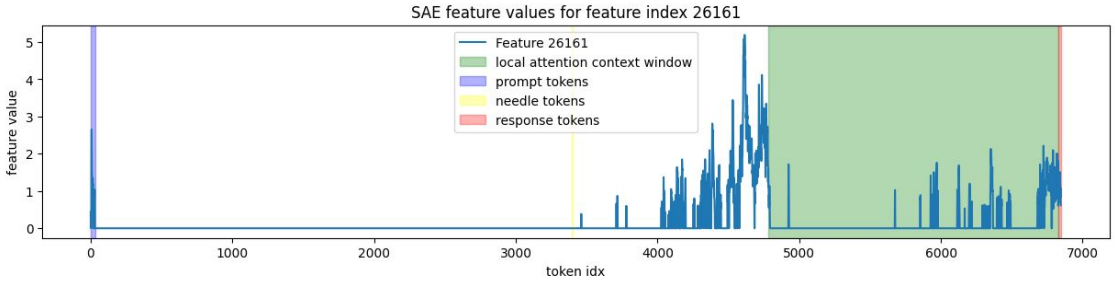
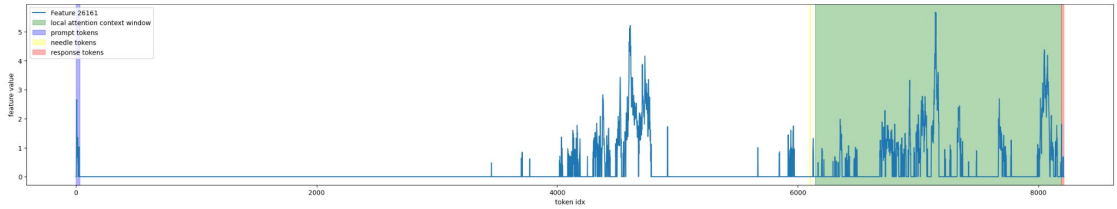
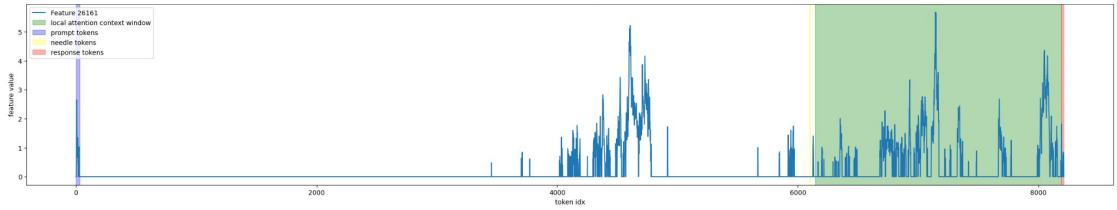


Figure 3.7: **Feature `sae.32x/26161` passes the needle to attention.** This feature has been identified as the key mechanism for holding and passing the needle to the local attention window. The feature values are visualized for a context length of $6,827$ tokens, with the needle (indicated in yellow) placed at a depth of 0.5. We find similar patterns across other context lengths and needle depths, suggesting a robust role for this feature in maintaining the needle's state. Prompt tokens, prepended to the context to instruct the model to memorize a specific magic number, are highlighted in blue. The local attention window is highlighted in green.

- Features `sae.32x/2052` and `sae.32x/120035`: These features activate at both the passkey position and when the magic number is later retrieved in the response, suggesting that they encode some representation of the magic number. However, their activation is not entirely exclusive to the needle. They occasionally spike for other tokens in the sequence, implying that these features might capture broader contextual or semantic information rather than being uniquely tied to the magic number.
- Feature `sae.32x/26161`: This feature emerges as one of the most critical for the retrieval task. It consistently activates for instruction tokens that ask the model to memorize a specific magic number. Additionally, it becomes increasingly active as tokens approach the local attention window boundary, and again when the magic number is retrieved. Figure 3.7 shows the activation pattern of this feature for an example with successful retrieval. Another feature, `sae.32x/24992`, exhibits similar behavior but appears to be less consistent and reliable compared to `sae.32x/26161`.



(a) The model retrieves the single-digit magic number correctly.

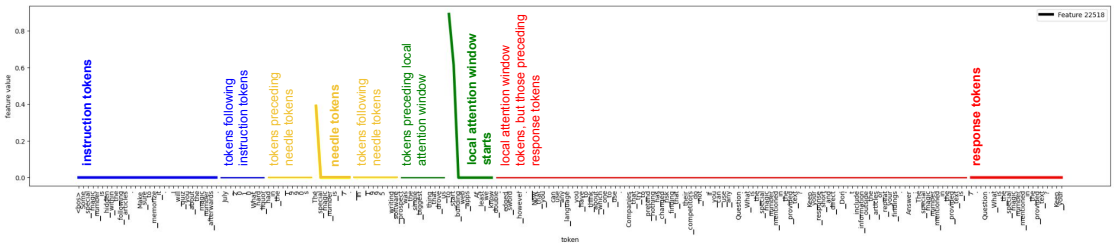


(b) The model fails to retrieve the single-digit magic number.

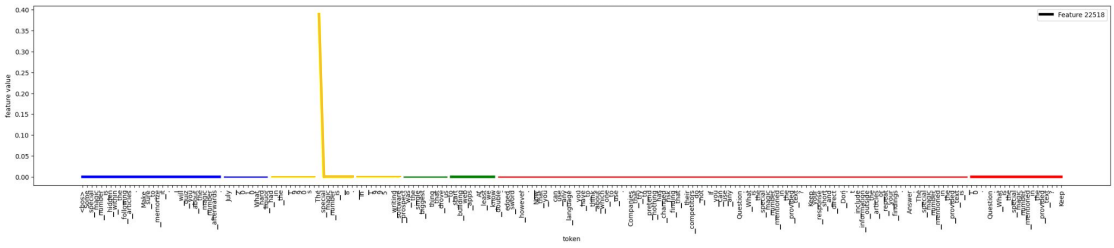
Figure 3.8: **Feature `sae.32x/26161` for successful vs. unsuccessful retrieval.** We plot the activation values of the feature over time to analyze activation differences. The prompt tokens are highlighted in blue, and the local attention window is marked in green. For both (3.8a) and (3.8b), the context length is set to $8,192$ tokens, with the needle placed at a depth of 0.75. The feature exhibits similar activation patterns for both successful and unsuccessful retrieval cases.

Next, we analyze the behavior of these features in scenarios of successful magic number retrieval compared to instances where the model fails to retrieve the correct value. However, contrary to our expectations, all the features identified before showed similar activation patterns for both successful and unsuccessful retrievals. This lack of differentiation suggests that the features identified while contributing to the retrieval task, may not be directly influencing the retrieval success. For example, in Figure 3.8, we illustrate the activations of feature `sae.32x/26161`, which we previously identified as being responsible for passing needle information to attention. Despite its consistent activation across both successful and unsuccessful retrievals, we observed no clear distinction in its behavior based on retrieval success.

Next, we examine features that consistently activate differently for successful vs. unsuccessful retrievals by focusing on those that commonly activate be-



(a) The model retrieves the single-digit magic number correctly.



(b) The model fails to retrieve the single-digit magic number.

Figure 3.9: **Feature `sae.32x/22518` for successful vs. unsuccessful retrieval.** We plot the activation values of the feature over time to analyze activation differences. The token types and their corresponding color coding are marked in (3.9a). For both (3.9a) and (3.9b), the context length is set to $8,192$ tokens, with the needle placed at a depth of 0.75. The feature exhibits a clear distinction in its activation patterns for successful vs. unsuccessful retrieval.

tween the passkey and the first in-attention-context token. Our analysis provides compelling evidence that feature `sae.32x/22518` is critical for promoting retrieval success. Figure 3.9 demonstrates that this feature is closely tied to passkey remembrance. Its activation at the first token within the local attention window serves as a key signal that the passkey information has been successfully transmitted. In contrast, for unsuccessful retrievals, `sae.32x/22518` reliably activates for the needle tokens but fails to activate at the first token within the local attention window. This behavior suggests that the needle information was not adequately carried forward into the attention window, leading to retrieval failure. This distinction highlights the role of `sae.32x/22518` in facilitating successful retrieval.

3.6 Further related work

We omit a discussion of alternate attention models here; for a comprehensive review of related work in that domain, please refer to Section 2.6.

Long-context retrieval benchmarks. Frontier Transformers have made significant strides in handling longer contexts (OpenAI et al., 2024; Anthropic, 2024; Georgiev et al., 2024). Noting that this work does not directly aim to improve the long-context capabilities of models but instead seeks to understand what is stored in the hidden memory of recurrent models to facilitate long-context retrieval, we find it more relevant to highlight related benchmarks rather than focusing on models designed for long-context tasks.

The Long Range Arena (Tay et al., 2020) is one of the earliest benchmarks used to evaluate models' ability to process long contexts across various modalities. More recent benchmarks, such as LongBench (Bai et al., 2024), L-Eval (An et al., 2023), and LooGLE (Li et al., 2024), include several downstream tasks explicitly designed to assess models' capabilities in understanding and generating lengthy contexts. In addition to assessing model performance on downstream tasks, benchmarks like the passkey retrieval task (Mohtashami and Jaggi, 2023) and the needle-in-a-haystack task (gkamradt, 2024) evaluate the ability to retrieve specific key information embedded in long filler contexts. For our study, we focus on variants of the passkey retrieval task rather than the needle-in-a-haystack test, as the latter requires an external model to assess the correctness of the retrieved information, which can be unreliable depending on the evaluation model.

More recently, Zhang et al. (2024a) proposed benchmarks extending long-

context evaluation beyond 100K tokens. Notably, Chen et al. (2024); Zhang et al. (2024a) identify that linear recurrent models suffer significant performance drops when the context length substantially exceeds their training length. In this work, we explore RecurrentGemma within its training length of $8,192$ tokens, leaving the exploration of longer contexts for future research.

Mechanistic interpretability. Mechanistic interpretability (Olah et al., 2020) seeks to reverse engineer and explain the inner workings of models, often through discovering and describing circuits (Wang et al., 2022; Hanna et al., 2023; Conmy et al., 2023; Marks et al., 2024), or by investigating whether certain behaviors can be traced to individual neurons or are represented in a more distributed fashion (Dalvi et al., 2019; Durrani et al., 2020; Bau et al., 2020; Elhage et al., 2022; Gurnee et al., 2023; Adly Templeton et al., 2024).

Recent studies show that sparse autoencoders can extract an overcomplete feature basis from language models (Yun et al., 2023; Bricken et al., 2023; Cunningham et al., 2023), with several sparse autoencoders trained on small open-source models (Marks, 2023; Bloom, 2024; Mossing et al., 2024). Rajamanoharan et al. (2024) and Taggart (2024) proposed using different activation functions to address activation shrinkage in ReLU sparse autoencoders. Building on this, Team et al. (2024b) trained and released the weights of GemmaScope: a comprehensive open suite of JumpReLU autoencoders (Rajamanoharan et al., 2024) trained on activations across all layers of Gemma models. Braun et al. (2024) trained sparse autoencoders on downstream KL divergence rather than reconstruction error. Gao et al. (2024) trained TopK sparse autoencoders on GPT-2 and GPT-4, and released scaling laws to facilitate optimal training. Most of these prior works focus on sparse autoencoders trained on residual streams in Transformers. To

the best of our knowledge, we are the first to analyze recurrent hidden memory using sparse autoencoders.

In addition to feature extraction, other interpretability approaches such as steering vectors (Li et al., 2023; Turner et al., 2023), probing (Belinkov, 2022), and causal tracing (Sharma et al., 2024) have also been explored.

3.7 Conclusion

In this chapter, we investigated the ability of RecurrentGemma to retrieve information embedded within long contexts. Our findings reveal that while the model performs well within the local attention window, its ability to retrieve information deteriorates significantly when the passkey resides beyond this boundary. By leveraging a k-sparse autoencoder, we identified specific features in the recurrent hidden states that promote successful needle retrieval. These features provide valuable insight into the internal dynamics of long-context modeling, suggesting a potential pathway for improving recurrent models.

Future work. While our discovery of retrieval-promoting features is promising, it relies heavily on manual inspection, which limits the robustness and generalizability of our findings. Future work should focus on developing stronger evaluation techniques to systematically assess the roles of these features. Two potential directions include:

- Causal interventions to introduce controlled modifications to the activation patterns of specific features to steer the model's behavior and directly test

their roles in retrieval tasks.

- Injecting noise to the recurrent hidden states or input sequences to analyze the behavior and robustness of retrieval features under perturbed conditions.

By advancing our understanding of feature behavior and their causal relationships to retrieval success, we can refine the design of recurrent models to enable more reliable and effective long-context processing.

CHAPTER 4

CONCLUSION

One key advantage of linear recurrent models over Transformers is their linear computational complexity with respect to sequence length, enabling faster inference for long sequences. This thesis examines the long-context capabilities of these models in two ways: by applying them to byte-level language modeling, which inherently involves $4\times$ longer sequences, and by investigating the features associated with the hidden memory of these recurrent models to uncover those that promote or guide associative recall.

We conclude this dissertation by discussing promising directions for future work inspired by these themes.

Enhancing recurrent model capacity. While linear recurrent models demonstrate impressive capabilities in modeling byte sequences, the situation shifts when considering long-context retrieval. In Chapter 3, we observed that although toy models with RG-LRU units trained on specific copy tasks perform well on in-context learning tasks involving copy and recall, RecurrentGemma's performance significantly degrades on the long-context passkey retrieval task when the passkey is moved outside the local attention window. These observations align with findings from Jelassi et al. (2024); Merrill et al. (2024); Chen et al. (2024); Zhang et al. (2024a). Despite having a constant-sized hidden memory—however large—linear recurrent models face inherent limitations: while they can process inputs of arbitrary length, there is an upper bound to the information their states can represent. Notably, Chen et al. (2024) identify this upper bound as a contributing factor to the inability to forget the earliest tokens, leading to inefficiencies in long-context tasks. They propose a few training-free strategies to

mitigate this limitation, albeit with some trade-offs in computational efficiency. Additionally, Cirone et al. (2024) emphasize the importance of introducing non-diagonal state transition matrix to unlock greater expressive power in state space models. These findings underscore the need recurrence architectures that enable dynamic scaling of memory capacity.

As an aside, while this work focuses on identifying features that promote or guide retrieval, an adjacent topic of interest could involve analyzing when instruction tokens are forgotten within the hidden state. Such forgetting could lead to the premature loss of passkey tokens, highlighting another critical area for improving long-context memory retention.

Understanding recurrence in distilled models. Recent advancements have highlighted the potential of distilling existing Transformers into linear recurrent models to achieve faster inference (Kasai et al., 2021; Zhang et al., 2024b; Mercat et al., 2024; Bick et al., 2024; Wang et al., 2024b; Zhang et al., 2024c). These approaches enable powerful language models to operate with linear-time complexity and constant-memory generation. Two promising research directions emerge in this context: (a) evaluating the performance of these distilled models in context retrieval tasks relative to their base Transformer architectures, and (b) investigating whether sparse autoencoders can uncover spanning features in these distilled models, indicating long-term information retention, similar to the features identified in this work, even when the autoencoders were not explicitly trained for this purpose.

Recurrence in instruction-tuned vs. standard variants. Our latest experiments revealed that the instruction-tuned variant, RecurrentGemma-9B-IT, demon-

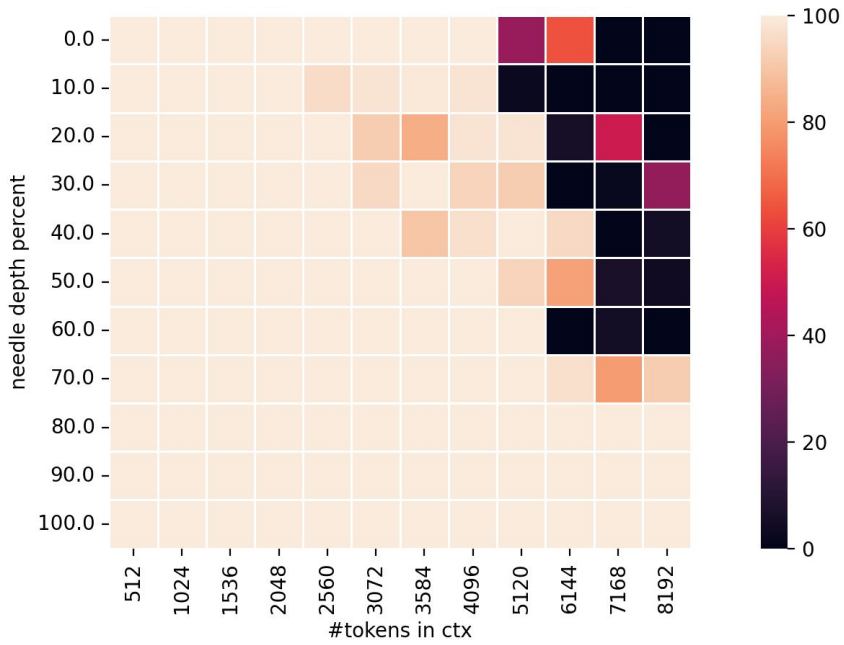


Figure 4.1: **Long-context passkey retrieval evaluation using RecurrentGemma-9B-IT.** We evaluate the retrieval performance of six-digit passkeys (spanning more than one token) across 100 random trials. Compared to the standard RecurrentGemma-9B model (see Figure 3.3), the instruction-tuned version demonstrates significantly improved retrieval effectiveness for passkeys beyond the local attention context.

strates significantly better performance in retrieval tasks compared to the standard RecurrentGemma-9B (see Figure 4.1). This enhancement is likely due to the framing of the retrieval task as an explicit instruction to locate a specific passkey within a long context, aligning closely with the capabilities fine-tuned during instruction training. Given RecurrentGemma-9B-IT's impressive performance, analyzing its activations through sparse autoencoders could yield valuable insights into the features that guide its long-context retrieval capabilities. Such an investigation might uncover whether instruction tuning alters the representation of information in the recurrent memory and, if so, which features are responsible for the observed improvements.

Beyond this, comparing the features extracted from RecurrentGemma-9B-IT

with those from the standard variant could help isolate the impact of instruction tuning. Such comparative analysis would also clarify whether instruction tuning inherently improves associative recall or simply better aligns the model's behavior with task-specific goals.

APPENDIX A

APPENDIX FOR CHAPTER 2

A.1 MambaByte training data specifics

	Total bytes	Total docs	Bytes/doc
PG19	11.74G	28,752	4,082,210
Stories	34.18G	948,247	36,045
Books	108.38G	196,640	551,179
ArXiv	60.27G	1,264,405	47,665
Code	677G	56,626,342	11,958

Table A.1: **Text dataset statistics.** The total bytes, total documents, and the mean document size (bytes per document) for each dataset.

We benchmark our results on various long-form text datasets. The PG19 dataset (Rae et al., 2020) is an extensive collection of full-length English books (written before 1919) from the Project Gutenberg online library. The PG19 dataset is ideal to test for long-distance context modeling (Gao et al., 2020b). The Stories dataset (Trinh and Le, 2018) is a subset of the CommonCrawl data used for commonsense reasoning and language modeling. The Books dataset (Gao et al., 2020b) is another collection of English books. The ArXiv dataset (Gao et al., 2020b) comprises technical publications in LaTeX from the arXiv online archive. Finally, the Code dataset (Gao et al., 2020b) is a large dataset of publicly available open-source code (under Apache, MIT, or BSD licenses). Dataset statistics are tabulated in Table A.1.

For the PG19 dataset, we employ the train, validation, and test data splits as indicated by Rae et al. (2020). For Stories, Books, ArXiv, and Code datasets, we randomly sample 40M consecutive bytes for testing and the rest to train MambaByte.

A.2 Compute-constrained modeling

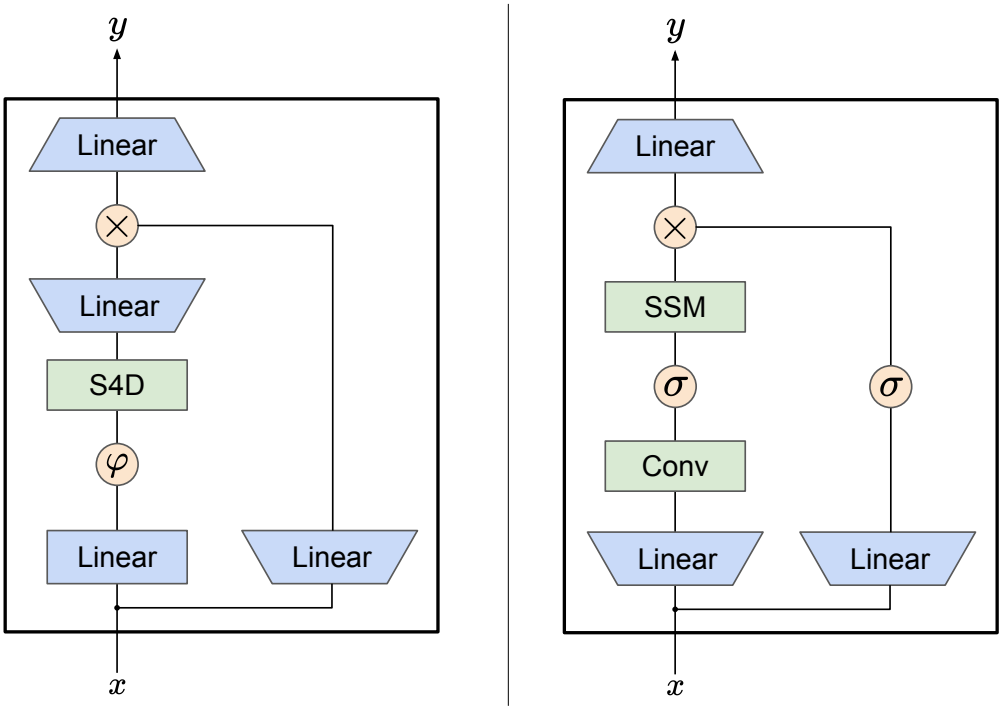


Figure A.1: **SSM model network architectures.** (Left) Gated-S4D block adapted from Mehta et al. (2023); (right) Mamba SSM block. (φ indicates GELU activation (Hendrycks and Gimpel, 2016), and σ indicates Swish activation (Ramachandran et al., 2017).)

As noted earlier, we evaluate and benchmark MambaByte in a compute-controlled setting. To this end, we estimate the FLOPs per byte incurred by various byte-level model architectures. We parameterize the architectures using hyperparameters n (n_g/n_l) number of (global/local) layers, dimension d (d_g/d_l) of the (global/local) residual stream, expansion factor e of linear layers, patch size p in MegaByte, state dimension n_{state} in SSMs, 1D convolution kernel size k, and low-rank projection dimension r in Mamba. We also include L_{ctx} bytes in the input context. Detailed component-wise compute counts for the forward pass are included in Table A.2.

For the medium-scale language modeling experiments (Table 1, §5 of Yu et al.

Model	Component	FLOPs per byte
Transformer (Vaswani et al., 2017)	Multi-head attention	$2n(4d^2 + 2L_{\text{ctx}}d)$
	Pointwise feed-forward	$2n(2ed^2)$
MegaByte[1] (Yu et al., 2023)	Embedding projection	$2d_g^2$
	Global transformer model	$2n_g(4d_g^2 + 2d_gL_{\text{ctx}}/p + 2ed_g^2)/p$
	Global-to-local projection	$2d_gd_l$
	Local transformer model	$2n_l(4d_l^2 + 2pd_l + 2ed_l^2)$
Gated-S4D (Figure A.1, left)	Linear projections	$2n(3ed^2 + d^2)$
	Kernel via Vandermonde $v(\overline{\text{A}})$	$n(\alpha_v ed(n_{\text{state}} + L_{\text{ctx}})\log_2^2(n_{\text{state}} + L_{\text{ctx}})/L_{\text{ctx}})$
	S4D SSM with convolution	$n(\alpha_{\text{fft}}\log(L_{\text{ctx}})ed + ed)$
	Element-wise gating	ned
MambaByte (Figure A.1, right)	Linear projections	$2n(3ed^2)$
	Pre-SSM 1D convolution	$2nked$
	$\Delta, \text{B}, \text{C}$ from input x	$2n(2edr + 2edn_{\text{state}})$
	Discretization, pre-scan: $\overline{\text{A}}$, $\overline{\text{B}}x$	$n(3edn_{\text{state}})$
	Recurrence w/ parallel scan	$n(edn_{\text{state}})$
	Output: $y = \overline{\text{C}}h + \overline{\text{D}}x$	$2nedn_{\text{state}} + ned$
	Element-wise gating	ned

Table A.2: **Compute (forward pass) estimates for various byte-level language models.** Embedding, de-embedding, and sub-leading terms such as biases, nonlinearities, and layer norms are omitted. (α_* indicates an implementation-specific constant scaling term.)

(2023)), Yu et al. (2023) employ the MegaByte-758M+262M model, with a context length of $8,192$ and patch size of 8, trained for 80B bytes. As shown in Figure A.2, MambaByte-353M ($n = 53$, $d = 1,024$, $e = 2$) and MegaByte-758M+262M use the same total compute in FLOPs; hence, we employ the MambaByte-353M to benchmark against MegaByte-758M+262M in Table 2.2 of §2.5.

For the PG19 scaling experiment (Table 2, §5 and Appendix D.3 of Yu et al. (2023)), Yu et al. (2023) use MegaByte-1.3B+350M (context length of $8,192$ and patch size of 8) trained for 400B bytes to benchmark the observed word-level perplexity against several state-of-the-art subword models. Owing to our hardware limitations, we train MambaByte-972M ($n = 48$, $d = 1,792$, $e = 2$) and control for the total compute used (see Figure A.2 to view the associated computational costs). All the model sizes and associated hyperparameters employed in this

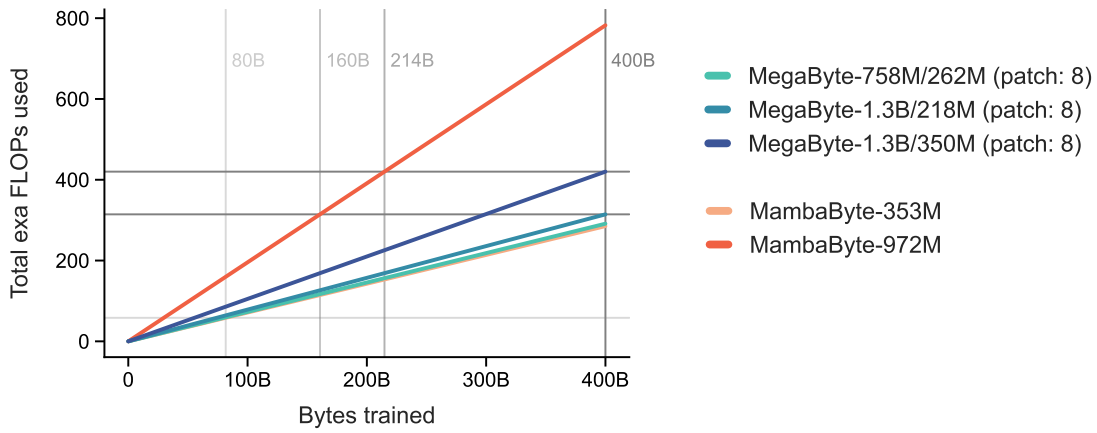


Figure A.2: **Computational cost for different model architectures at different scales.** All models use a context length of $8,192$, and MegaByte architectures use a patch size of 8.

work are tabulated in Table A.3.

Model	Parameters	Hyperparameters				
		n (n_g/n_l)	d (d_g/d_l)	e	L_{ctx}	Others
Transformer	320M (Yu et al., 2023)	22	1,024	4	1,024	heads: –
	350M (Yu et al., 2023)	24	1,024	4	1,024	heads: 16
	361M	28	1,024	4	8,192	heads: 16
PerceiverAR	248M (Yu et al., 2023)	17	1,024	4	8,192	latents: 1,024
MegaByte	193M+177M[1]	14/14	1,024/1,024	4	8,192	$p = 4, 8$; heads: 16/16
	758M+262M (Yu et al., 2023)	14/18	2,048/1,024	4	8,192	$p = 8$; heads: 16/16
	1.3B+218M (Yu et al., 2023)	24/15	2,048/1,024	4	8,192	$p = 8$; heads: 32/–
	1.3B+350M (Yu et al., 2023)	24/24	2,048/1,024	4	8,192	$p = 8$; heads: 32/16
Gated-S4D	368M	26	1,024	4	8,192	$n_{\text{state}} = 64$
MambaByte	353M	53	1,024	2	8,192	$k = 4; n_{\text{state}} = 16; r = 64$
	972M	48	1,792	2	8,192	$k = 4; n_{\text{state}} = 16; r = 112$
	1.6B	48	2,304	2	8,192	$k = 4; n_{\text{state}} = 16; r = 144$

Table A.3: **Model hyperparameters.** We report the model size and associated hyperparameters for all the models employed in this study. (Accompanying citation indicates the work from which the associated configuration is noted; fields marked as – are unknown.)

[1]We used the open-source implementation: https://github.com/lucidrains/MEGABYTE-pytorch.

A.3 Training recipes used in MambaByte

All the models in this study were trained using an AdamW optimizer with $\beta = (0.9, 0.95)$. We used a linear learning rate warm-up (for the first 500 steps) followed by cosine annealing. Keeping consistent with MegaByte training (Yu et al., 2023), we used a batch size of 48 across all our experiments. Additionally, we do not use dropout with any of our models.

For the experiments in Figure 2.1, we conducted a hyperparameter search using peak learning rates of 0.0002, 0.0006, and 0.0008 and clipped the gradient norm to 1.0 for all the models. The best-observed performance curve for each model is reported in Figure 2.1. Furthermore, we use an improved Transformer recipe that uses RMSNorm instead of LayerNorm, rotary positional encodings (Su et al., 2021), and linear terms without bias (same as Yu et al. (2023)).

In our medium-scale experiments shown in Table 2.2, we set the peak learning rate to 0.0004 and clipped the gradient norm to 0.1. We trained the MambaByte-353M for a total of 80K steps, equivalent to $80,000 \times 48 \times 8,192 \approx 30$B bytes.

In the large-scale experiment on PG19, we use a similar setting to that in the medium-scale experiments: the peak learning rate is set to 0.0004, and the gradient norm is clipped to 0.1. The MambaByte-972M is trained for 380K steps, equivalent to $380,000 \times 48 \times 8,192 \approx 150$B bytes.

A.4 Parallel scan for linear recurrences

The parallelization of scan operations has been well explored Ladner and Fischer (1980); Lakshmivarahan and Dhall (1994); Blelloch (1990), with several scientific computing libraries containing optimized routines. In this section, we detail the use of a parallel scan for linear recurrence in §2.2 as a specific application of a more generalized setting in §1.4 of Blelloch (1990).

To compute a linear recurrence $h[k] = \overline{A}h[k-1] + \overline{B}x[k]$ for a sequence of length L, let us define L initial elements $e_{1:L}$, such that:

$$e_k = (A_k, b_k) := \left(\overline{A}, \overline{B}x[k]\right), \tag{A.1}$$

where $A_k, \overline{A} \in \mathbb{R}^{n\times n}$, $\overline{B} \in \mathbb{R}^{n\times 1}$, and $x[k], b_k \in \mathbb{R}^n$. These initial L elements are precomputed before the scan.

Now, the binary associative operator $\bullet$ to use on the linear recurrence is defined as:

$$e_j \bullet e_k \equiv (A_k A_j, A_k b_j + b_k). \tag{A.2}$$

Associativity of the operator. Note that the binary operator shown above in (A.2) is associative:

$$\begin{aligned}
e_i \bullet (e_j \bullet e_k) &= e_i \bullet (A_k A_j, A_k b_j + b_k) \\
&= ((A_k A_j) A_i, A_k A_j b_i + (A_k b_j + b_k)) \\
&= (A_k (A_j A_i), A_k (A_j b_i + b_j) + b_k) \\
&= (A_j A_i, A_j b_i + b_j) \bullet e_k \\
&= (e_i \bullet e_j) \bullet e_k.
\end{aligned}$$

Illustration of linear recurrence using the operator. Using the binary associative operator in (A.2), we can see how the linear recurrence $h[k] = \overline{A}h[k-1] + \overline{B}x[k]$ can be computed. By setting $h[0] = 0$, we have:

$$
\begin{aligned}
h[1] &= \overline{B}x[1]; \\
h[2] &= \overline{AB}x[1] + \overline{B}x[2]; \\
h[3] &= \overline{A}^2\overline{B}x[1] + \overline{AB}x[2] + \overline{B}x[3]; \\
&\vdots \\
h[k] &= \overline{A}^{k-1}\overline{B}x[1] + \overline{A}^{k-2}\overline{B}x[2] + \cdots + \overline{AB}x[k-2] + \overline{B}x[k-1].
\end{aligned}
$$

Algorithm 1 Blelloch work-efficient parallel (inclusive) scan on a PRAM (for an array of L elements, using a binary operator $\bullet$).

procedure REDUCE($[e_1, e_2, \ldots, e_L]$, $\bullet$)
  for $d = 1, \ldots, \log_2(L)$ **do**
    for $i = 1, \ldots, L$ in increments of 2^d **in parallel do**
      $k \leftarrow i + 2^d - 1$; $j \leftarrow i + 2^{d-1} - 1$
      $e_k \leftarrow e_j \bullet e_k$

procedure DOWNSWEEP($[e_1, e_2, \ldots, e_L]$, $\bullet$)
  $e_L \leftarrow (\mathrm{I}, 0)$
  for $d = \log_2(L), \ldots, 1$ **do**
    for $i = 1, \ldots, L$ in increments of 2^d **in parallel do**
      $k \leftarrow i + 2^d - 1$; $j \leftarrow i + 2^{d-1} - 1$
      temp $\leftarrow e_j$
      $e_j \leftarrow e_k$
      $e_k \leftarrow e_k \bullet$ temp

procedure SCAN($[e_1, e_2, \ldots, e_L]$, $\bullet$)
  REDUCE($[e_1, e_2, \ldots, e_L]$, $\bullet$) ▷ $L-1$ $\bullet$ computes.
  all_reduced_out $\leftarrow e_L$ ▷ Last element of the inclusive scan.
  UPSWEEP($[e_1, e_2, \ldots, e_L]$, $\bullet$) ▷ $L-1$ $\bullet$ computes, $L-1$ swaps.
  return $[e_2, \ldots, e_L,$ all_reduced_out$]$ ▷ Shift one to the left, append all-reduced output.

Blelloch work-efficient parallel scan Blelloch (1990) for a given array of L elements $e_{1:L}$ using the binary operator in (A.2) is shown in Algorithm 1. We

initialize $e_{1:L}$ using (A.1) and run a parallel scan in Algorithm 1. For illustrative purposes, let $L = 4$; now we run the reduce phase as follows:

$$
\begin{aligned}
&e_1 = (\overline{A}, \overline{B}x[1]).\\
&e_2 \leftarrow e_1 \bullet e_2 = (\overline{A}, \overline{B}x[1]) \bullet (\overline{A}, \overline{B}x[2]) = (\overline{A}^2, \overline{AB}x[1] + \overline{B}x[2]).\\
&e_3 = (\overline{A}, \overline{B}x[3]).\\
&e_4 \leftarrow e_2 \bullet (e_3 \bullet e_4)\\
&\quad = e_2 \bullet ((\overline{A}, \overline{B}x[3]) \bullet (\overline{A}, \overline{B}x[4]))\\
&\quad = (\overline{A}^2, \overline{AB}x[1] + \overline{B}x[2]) \bullet (\overline{A}^2, \overline{AB}x[3] + \overline{B}x[4])\\
&\quad = (\overline{A}^4, \overline{A}^3\overline{B}x[1] + \overline{A}^2\overline{B}x[2] + \overline{AB}x[3] + \overline{B}x[4]).
\end{aligned}
$$

Next, we store e_4 above (as all-reduced output) for later, then set $e_4 = (\mathrm{I}, 0)$ and run the down sweep phase as follows:

$$
\begin{aligned}
e_2 \leftarrow (\mathrm{I}, 0); e_4 \leftarrow (\mathrm{I}, 0) \bullet (\overline{A}^2, \overline{AB}x[1] + \overline{B}x[2]) &= (\overline{A}^2, \overline{AB}x[1] + \overline{B}x[2]).\\
e_1 \leftarrow (\mathrm{I}, 0); e_2 \leftarrow (\mathrm{I}, 0) \bullet (\overline{A}, \overline{B}x[1]) &= (\overline{A}, \overline{B}x[1]).\\
e_3 \leftarrow (\overline{A}^2, \overline{AB}x[1] + \overline{B}x[2]); e_4 \leftarrow (\overline{A}^2, \overline{AB}x[1] + \overline{B}x[2]) &\bullet (\overline{A}, \overline{B}x[3])\\
&= (\overline{A}^3, \overline{A}^2\overline{B}x[1] + \overline{AB}x[2] + \overline{B}x[3]).
\end{aligned}
$$

To generate an inclusive scan, we shift the resulting array one to the left and insert the all-reduced output to the end. As a result, we have the output elements of the parallel scan, $r_{1:4}$, as:

$$
\begin{aligned}
&r_1 = e_2 = (\overline{A}, \overline{B}x[1]) = (\overline{A}, h[1]);\\
&r_2 = e_3 = (\overline{A}^2, \overline{AB}x[1] + \overline{B}x[2]) = (\overline{A}^2, h[2]);\\
&r_3 = e_4 = (\overline{A}^3, \overline{A}^2\overline{B}x[1] + \overline{AB}x[2] + \overline{B}x[3]) = (\overline{A}^3, h[3]);\\
&r_4 = (\overline{A}^4, \overline{A}^3\overline{B}x[1] + \overline{A}^2\overline{B}x[2] + \overline{AB}x[3] + \overline{B}x[4]) = (\overline{A}^4, h[4]).
\end{aligned}
$$

Observe that the second entry b_k of $r_k = (A_k, b_k)$ corresponds to the desired $h[k]$.

Efficiency of a parallel scan. From Algorithm 1, we note that the scan operation can be performed in $\mathcal{O}(T_\bullet \log_2(L))$ time using $L/2$ processors, where $T_\bullet$ is the cost of computing $e_j \bullet e_k$. $T_\bullet$ depends on the structure of $\overline{A}$; for a general $\overline{A}$, $T_\bullet$ is dominated by the cost of matrix-matrix multiplication, $\mathcal{O}(n^3)$. Hence, the overall cost of a parallel scan for an L-length sequence and a general $\overline{A}$ is $\mathcal{O}(n^3 \log_2(L))$.

Furthermore, for $F_\bullet$ FLOPs per $\bullet$ compute, we note the parallel scan to incur $\mathcal{O}(F_\bullet L)$ FLOPs, making the work-efficient parallel scan theoretically efficient for a tractable $F_\bullet$. For a general $\overline{A}$, Algorithm 1 results in $\mathcal{O}(n^3 L)$ FLOPs.

A.5 Discretization and selection in Mamba

Discretization has deep connections to continuous-time systems, which allows for desirable properties such as model normalization (Orvieto et al., 2023; Gu et al., 2023) and resolution invariance (Nguyen et al., 2022). In this section, we review how zero-order hold discretization of Mamba selective SSM can be viewed as a generalization of the gating mechanism in recurrent networks. An illustration of the Mamba SSM discretization and illustration is depicted in Figure A.3.

Zero-order hold discretization. For a given input $x(t) \in \mathbb{R}$, system matrix $A \in \mathbb{R}^{n \times n}$, and input matrix $B \in \mathbb{R}^{n \times 1}$, the differential equation for a continuous-

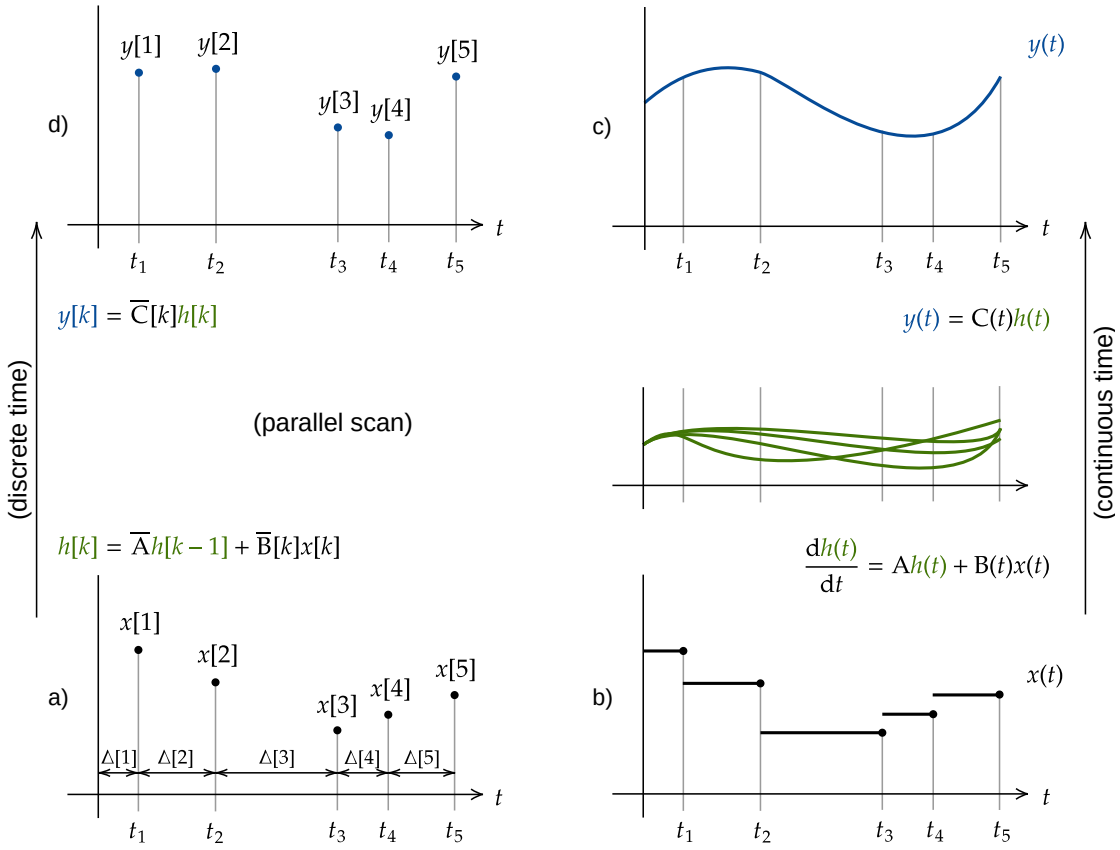


Figure A.3: **Illustration of the Mamba SSM.** (a) The discrete-time input $x[k]$, along with input-selective $\Delta[k]$. (b) The continuous-time signal $x(t)$. (c) Mathematically, the SSM transforms the continuous-time $x(t)$ through an n-dimensional hidden state (here, $n = 4$) using parameters A and $\mathrm{B}(t)$, which is then mapped to the output $y(t)$ using $\mathrm{C}(t)$. (d) Practically, we compute $y[k]$ using a discrete-time parallel scan at the steps defined by $\Delta[k]$ and discrete-time matrices $\overline{\mathrm{A}}[k]$, $\overline{\mathrm{B}}[k]$, and $\overline{\mathrm{C}}[k]$. At inference, we run the recurrence directly.

time LTI SSM, as noted in §2.2, can be written as:

$$\begin{aligned}
\exp(-At)\left(\frac{\mathrm{d}h(t)}{\mathrm{d}t} - Ah(t)\right) &= \exp(-At)Bx(t) \\
\Rightarrow \exp(-At)\frac{\mathrm{d}h(t)}{\mathrm{d}t} - \exp(-At)Ah(t) &= \exp(-At)Bx(t) \\
\Rightarrow \frac{\mathrm{d}}{\mathrm{d}t}\exp(-At)h(t) &= \exp(-At)Bx(t).
\end{aligned}$$

By integrating from 0 to s, we get the following integral form:

$$\exp(-As)h(s) - h(0) = \int_0^s \exp(-At)Bx(t)\mathrm{d}t$$
$$\Rightarrow h(s) = \exp(As)\left(h(0) + \int_0^s \exp(-At)Bx(t)\mathrm{d}t\right). \tag{A.3}$$

To discretize the above continuous-time LTI SSM, we sample the system at equispaced intervals with sampling frequency Δ: $x[k] = x(k\Delta)$ and assume a zero-order hold, i.e., $x(t)$ is constant between samples: $x(k\Delta + \xi) = x(k\Delta) = x[k]$ for any $\xi \in [0, \Delta)$. Now, by letting $s = (k+1)\Delta$, (A.3) can be simplified as:

$$\begin{aligned}
h[k+1] &= \exp(A(k+1)\Delta)\left(h(0) + \int_0^{(k+1)\Delta} \exp(-At)Bx(t)\mathrm{d}t\right) \\
&= \exp(A(k+1)\Delta)\Bigg(h(0) + \int_0^{k\Delta} \exp(-At)Bx(t)\mathrm{d}t + \\
&\qquad \int_{k\Delta}^{(k+1)\Delta} \exp(-At)Bx(t)\mathrm{d}t\Bigg) \\
&= \exp(A\Delta)\left(\exp(Ak\Delta)\left(h(0) + \int_0^{k\Delta} \exp(-At)Bx(t)\mathrm{d}t\right)\right) + \\
&\qquad \exp(A(k+1)\Delta)\int_{k\Delta}^{(k+1)\Delta} \exp(-At)Bx(t)\mathrm{d}t \\
&= \exp(A\Delta)h[k] + \exp(A(k+1)\Delta)\int_{k\Delta}^{(k+1)\Delta} \exp(-At)Bx(t)\mathrm{d}t.
\end{aligned}$$

Now, from the zero-order hold assumption, we realize that $x(t) \in [k\Delta, (k+1)\Delta)$ is constant; hence,

$$\begin{aligned}
h[k+1] &= \exp(A\Delta)h[k] + \exp(A(k+1)\Delta)\left(\int_{k\Delta}^{(k+1)\Delta} \exp(-At)\mathrm{d}t\right)Bx[k] \\
&= \exp(A\Delta)h[k] + \left(\int_{k\Delta}^{(k+1)\Delta} \exp(A((k+1)\Delta - t))\mathrm{d}t\right)Bx[k].
\end{aligned}$$

Finally, we simplify the above integral by variable substitution. By using

$v = (k+1)\Delta - t$ and $\mathrm{d}v = -\mathrm{d}t$, we have:

$$\begin{aligned} h[k+1] &= \exp(A\Delta)h[k] - \left(\int_{\Delta}^{0} \exp(Av)\mathrm{d}v\right) Bx[k] \\ &= \exp(A\Delta)h[k] - \left(A^{-1}\exp(Av)\Big|_{\Delta}^{0}\right) Bx[k] \\ &= \underbrace{\exp(A\Delta)}_{\overline{A}} h[k] + \underbrace{A^{-1}(\exp(A\Delta) - \mathrm{I})B}_{\overline{B}} x[k]. \end{aligned}$$

Hence, the matrices of the associated discrete SSM are:[2]

$$\overline{A} = \exp(A\Delta); \quad \overline{B} = A^{-1}(\exp(A\Delta) - \mathrm{I})B; \quad \overline{C} = C; \quad \overline{D} = D.$$

Selection mechanics and gating in recurrent networks. Gu and Dao (2024) note that a selective SSM can be realized as a gated recurrence by setting $\Delta = \mathrm{softplus}(z(x)) = \mathrm{softplus}(W_{\Delta}(W_R x))$ (as indicated in (2.3) of §2.2). By letting $\mathrm{A} = -1$, $\mathrm{B} = 1$, and $n = 1$, the authors observe:

$$\begin{aligned} \overline{\mathrm{A}} &= \exp(\mathrm{A}\,\Delta) \\ &= \exp(-\log(1+\exp(z(x)))) \\ &= \frac{1}{1+\exp(z(x))} \\ &= \sigma(-z(x)) \\ &= 1 - \sigma(z(x)). \end{aligned} \qquad \begin{aligned} \overline{\mathrm{B}} &= \mathrm{A}^{-1}(\exp(\mathrm{A}\,\Delta) - \mathrm{I})\,\mathrm{B} \\ &= \mathrm{I} - \exp(\mathrm{A}\,\Delta) \\ &= \sigma(z(x)). \end{aligned}$$

Using $\overline{\mathrm{A}}$ and $\overline{\mathrm{B}}$ from above in the discrete recurrence (2.2), the selective SSM takes the form of a 1D gated recurrence:

$$h[k] = (1 - \sigma(z(x)))\,h[k-1] + \sigma(z(x))x[k]. \tag{A.4}$$

[2] In Mamba (Gu and Dao, 2024), B is discretized through a simplified Euler (as opposed to zero-order hold) discretization from empirical observations of A being more important than B, and the performance does not change significantly with simplification on B.

It is interesting to note from (A.4) that $\lim_{\Delta \to \infty} h[k] = x[k]$ and $\lim_{\Delta \to 0} h[k] = h[k-1]$: a large Δ ($\Delta \to \infty$) denotes the evolution of the system to focus only on the current input and forgetting the state. In contrast, a small Δ ($\Delta \to 0$) represents a transient input being ignored.

Selectivity of A, B, and C matrices. Gu and Dao (2024) argue that since the system matrix A only affects the model through Δ, i.e., $\overline{\mathrm{A}} = \exp(\mathrm{A}\,\Delta)$. Hence, the selectivity in Δ is sufficient to ensure selectivity in A.

While the selectivity in Δ enables selectivity in the input matrix B, Gu and Dao (2024) hypothesize that making B and C selective (in addition to Δ) would allow for more fine-grained control based on the content $x[k]$ and evolving context $h[k]$.

A.6 Evaluation metrics for byte-level models

Subword-based language models (Vaswani et al., 2017; Hawthorne et al., 2022; Hutchins et al., 2022) report their performance in word or subword-level PPL, while byte-level language models (Xue et al., 2022; Yu et al., 2023) report theirs in BPB. To facilitate meaningful comparisons, we report performance in BPB when benchmarking against byte-level models and word-level PPL when comparing to token-level models.[3] This section details the conversion from BPB and subword-level PPL to word-level PPL.

Irrespective of the underlying segmentation, the amount of information $I(D)$

[3]Unless stated otherwise, we use PPL to report word-level (not subword-level) perplexity.

	L_B	L_S	L_W	L_B/L_W	L_S/L_W
Train	$11,677,824,216$	$2,914,582,573$	$1,973,048,393$	5.92	1.48
Validation	$17,733,002$	$4,357,506$	$3,007,061$	5.90	1.45
Test	$41,289,101$	$10,282,006$	$6,965,511$	5.93	1.48

Table A.4: **PG19 dataset statistics.** Split-wise UTF-8 encoded byte L_B, SentencePiece-tokenized subword L_S, and space-separated word L_W counts in the PG19 dataset. (The byte count includes the newline character.) We also indicate the associated bytes per word L_B/L_W and subwords per word L_S/L_W.

in a given dataset D is constant. Simply put,

$$I(D) = L_W \text{ bits per word} = L_S \text{ bits per subword} = L_B \text{ bits per byte} \tag{A.5a}$$

$$\triangleq \frac{-\ln(D; \text{model})}{\ln(2)}, \tag{A.5b}$$

where L_W, L_S, and L_B are the length of the dataset in words, subwords, and bytes, respectively. From (A.5), we observe:

$$\text{BPB} = \frac{-\ln(D; \text{model})/L_B}{\ln(2)} = \frac{\ell_{\text{byte}}}{\ln(2)},$$

where ℓ_{byte} is the observed byte-level negative log-likelihood loss (computed using $\ln$). From (A.5), we also note the following conversion from BPB to word-level PPL:

$$\frac{-\ln(D; \text{model})/L_W}{\ln(2)} = \frac{L_B}{L_W}\,\text{BPB} = \frac{L_B}{L_W}\frac{\ell_{\text{byte}}}{\ln(2)}$$

$$\Rightarrow \text{PPL} = \exp\left(\frac{L_B}{L_W}\ell_{\text{byte}}\right) = \exp\left(\frac{L_B}{L_W}\ln(2)\,\text{BPB}\right).$$

Similarly, we can compute word-level PPL from the observed subword-level negative log-likelihood loss ℓ_{subword} as:

$$\text{PPL} = \exp\left(\frac{L_S}{L_W}\ell_{\text{subword}}\right).$$

For the PG19 dataset, we train MambaByte-972M to minimize BPB over the training data and report word-level PPL on the test data. In our medium-scale

benchmarking experiments, for (subword) Mamba-1.03B, we trained a 32K-subword vocabulary using the SubwordTextEncoder from the `tfds` package in TensorFlow, the same as Rae et al. (2020). Split-wise values of L_B/L_W and L_S/L_W for the PG19 dataset are tabulated in Table A.4.

A.7 Speculative decoding through subword drafting

Algorithm 2 above outlines our speculative decoding approach: the smaller subword draft model drafts m subwords at a time, which are then verified at a byte-level by a larger MambaByte model.

A.8 Synthetic noise settings for evaluating byte-level models

To confirm the robustness of MambaByte to input text corruptions, we employ the following synthetic noise settings adapted from Xue et al. (2022):

1) *Drop*: Bytes are dropped with a pre-set probability.
2) *Repetition*: Bytes are repeated one to three times (with equal likelihood).
3) *Antspeak*: Every character is capitalized and padded with spaces.
4) *Uppercase*: Characters are converted to uppercase with a pre-set probability.
5) *Random case*: Every character is set to a random case.

In addition to these five settings, we include the *character swap* setting, where consecutive bytes are swapped with some probability.

Algorithm 2 Speculative decoding iteration with subword drafter and byte-level verifier and corrector. (We use $\tilde{b}$ to indicate a drafted byte, while $\hat{b}$ denotes a corrected byte.)

Inputs: M_{subword}, M_{byte}, prefix subwords $s_{1:t}$, previous M_{subword} hidden state $\hbar_{\text{prev}}$, previous M_{byte} hidden state h_{prev}, draft block size m, verify model tolerance n.

▷ Sample m draft subwords $\tilde{s}_j$s from M_{subword} autoregressively and record the hidden states $\hbar_j$s at each timestep. ◁

$\hbar_0 \leftarrow \hbar_{\text{prev}}$

for $j = 1, \ldots, m$ **do**

  $q_j(x), \hbar_j \leftarrow M_{\text{subword}}(s_{1:t} + \tilde{s}_{1:j-1}, \hbar_{j-1})$

  $\tilde{s}_j \sim q_j(x)$

$b_{1:t'}, \tilde{b}_{1:n} \leftarrow \text{bytes}(s_{1:t}), \text{bytes}(\tilde{s}_{1:m})$ ▷ Get bytes for both prefix and drafted subwords.

▷ Run M_{byte} in parallel to verify the drafted bytes, while recording the associated hidden states h_is. ◁

$(p_1(x), h_1), \ldots, (p_n(x), h_n) \leftarrow$

$M_{\text{byte}}(b_{1:t'}, h_{\text{prev}}), M_{\text{byte}}(b_{1:t'} + \tilde{b}_1), \ldots, M_{\text{byte}}(b_{1:t'} + \tilde{b}_{1:n-1})$

▷ Find the bifurcation position c such that $\tilde{b}_{1:c}$ drafted bytes all fall in top-β candidates of M_{byte}, while $\tilde{b}_{c+1}$ does not. ◁

$c \leftarrow \min\left(\{i \mid 1 \le i \le n, \text{rank}_{p_i}(\tilde{b}_i) > \beta\} \cup \{n\}\right)$

$c' \leftarrow \min\left(\{j \mid 1 \le j \le m, \text{cumsum}(\text{len}(\tilde{s}_{1:m}))[j] > c\} \cup \{m\}\right)$ ▷ Find associated subword bifurcation position.

▷ Starting from $\tilde{b}_c$ (and using h_c), sample corrected bytes $\hat{b}_i$s from M_{byte} autoregressively until a boundary byte (e.g., space) is obtained. ◁

$\hat{b}_c \leftarrow \tilde{b}_c; k \leftarrow 0$

while $\hat{b}_{c+k}$ is not a boundary byte **do**

  $k \leftarrow k + 1$

  $p_{c+k}(x), h_{c+k} \leftarrow M_{\text{byte}}(\hat{b}_{c+k-1}, h_{c+k-1})$

  $\hat{b}_{c+k} \sim p(x)$

return generated bytes $\tilde{b}_{1:c} + \hat{b}_{c+1:c+k}$, M_{byte} last hidden state h_{c+k}, M_{subword} hidden state $\hbar_{c'}$ as cache to restart from.

A.9 PG19 generation samples generated by MambaByte

This section includes a few sample generations from the MambaByte-972M trained on the PG19 dataset. We use Nucleus sampling with $p = 0.98$ (Holtzman et al., 2020) and generate continuations for a total of $8,192$ bytes (including the

given context prefix). Furthermore, we chose the same test set prefixes used in Appendix F of Rae et al. (2020). We observe that the model is able to continue the dialogue in the style of the prefix and effectively recall the character names over hundreds of bytes.

Baby Mine by Margaret Mayo

Context (487 bytes):

> CHAPTER I
>
> Even in college Alfred Hardy was a young man of fixed ideas and high ideals and proud of it.
>
> His friend, Jimmy Jinks, had few ideas and no ideals, and was glad of it, and before half of their first college term had passed, Jimmy had ridded himself of all such worries as making up his own mind or directing his own morals. Alfred did all these things so much better, argued Jimmy, furthermore, Alfred LIKED to do them–Jimmy owed it to his friend to give him that pleasure.

MambaByte-972M:

> It was not that Alfred really considered his friend an ideal husband, but Jimmy admitted that it looked differently. And, too, it was all there was to life.

And on went Alfred, through long years of honest comradeship and unblushing promises and bubbling kindness. In his heart he had long since forgotten Jimmy and his friendship, since he realized the superiority of Jimmy's ideal, if of any one's.

It was to be expected that Prince William would fall a victim. Perhaps he was a modified betting man, and then had thrown over Nature and his team-mates, as a victim to ridicule or a follower of hazardous ways.
It was then that Prince William went back into society, perhaps sought an interview with Miss La Roche, and learned that he had been the victim of romance death, and a suicide, and had been endowed with an absurd roman
cross. Whatever it was, he saw in it the myth of Alfred's heroic deeds or the myth of a yellow dog in the market plate. It sobered him gradually.
He became nearly a convert, and it was Alfred who, in bidding Miss Rossiter
good-by, said that he would never forget her, and never would be friendly with his friend's family. It was not till now that Miss La Ross realized that a really good life results in an excellent impulse, but the sequel proved the truth of Miss La Ross's predictions. One night, a year before Prince William's death, when she was wearing a diamond star in her hair, and admiring it, she heard Jimmy's front door open in the hall, and, turning round to look through the open doorway, in came Jimmy, and after that where Jimmy went Alfred would sometimes go to the house next door, and they went over this now, and ever since Miss La Ross's spirit had been there, and Jimmy had often insisted that a marriage should be

accomplished in some ancient house–"And we will found a family of our own," said Jimmy, "a for some distant house in this great Belgium."

The days went on and it was a glorious spring morning. Early in March a thick autumn rain had fallen on the house in New York and the leafless trees looked weird and strange, and the wind roared like a crested wave of fire. In the afternoon the thunder roared and soon an occasional gray drizzle eddied about the wet streets and the city glow and tint began to merge into a world of pink and red and amber and purple and sunset colors.

The low ceiling pillars had covered their treasures of gold and gems, and the great stained-glass windows in dreamily gentle, half-lights, had arranged themselves in red and gold stripes of magic tints. Even though the delicate color scheme had had its disappointments once, it had certainly affected one hour in this room as well as ten months later and longer stays at least would have done.

To-day a soft South wind had drifted through the open door, and a sudden east wind had driven open the French windows of Miss La Ross's bedroom, and it rained in pure flames between the ceiling and boarded floor. Alfred's room was fragrant with his presence.

"... A little singing bird
That, living in a cage, demands a friend
Whose cool-blooded lure can warm the heart with love

To a fluttering, wounded thing.

As in a pathetic garden, so in the hall room.

It was cold, to-day. Already the rooms seemed overheated. The curtains were already half drawn.

She shivered.

"Mid-winter, to-day," thought Alfred, watching the sweep of Ann's white shoulder and patting her thin cotton frock. "Seven there were of days. And seven is yet untold gone. Fine, fine day, by Christ! Come out of this old soot, and we'll fly... Away. God rest his soul from hell, if ever such a devil crawled this broad, raw earth.... Where are you, Ann?"

Ann waited and trembled, she knew not why, for a sharp voice was asking suddenly for the check book in her hand.

"Get me change enough to pay for lunch for Jimmy," Alfred chided.

Before the one empty chair on the hall table and under the curtains lay a crashing pile of ready money. "And the window shades are closed," added Alfred.

"It won't shut out the rain," smiled Ann.

"But he won't care," protested Ann.

Alfred laid a strong withdrawing hand on the fair golden hair for a moment.

"It's all right," he coaxed. "Without a cent behind them to-day we can put in four thousand and close the bottom against a falling price like this." He was holding up the window sill six inches.

While he stood she whispered:

"I'm only lucky to save the day."

"He helps you without a reward," Alfred said.

"He's kind... and darned bad."

Ann noted dangerous things that afternoon.

"You could sing and play?" she asked.

"No, no!" insisted Alfred. "I CAN'T play and sing. The room is cold. It's warm within."

Alfred was changing clothes when he had that lucky escape, and Alfred momentarily forgot his debt. Ann laid the bill she had placed on the

table, and when she had gone Alfred had not even looked at it, and it was the act she saw in that frame of mind, remembering it, that made her put it back again.

Now Alfred was thoroughly cold and temperamental, and when he probed an obligation that he had just been trying to shift on the other fellow, he was more easily reminded. When Jimmy, cold and hungry, had wormed his way into his room that day at dinner, and been halted at his close chair by the soup stove, the young man's gaze had fixed furiously to the platter of gold and had immediately started on the other food with an intensity of expression that had awakened Jimmy's appreciation of the hot day of purposes and had aroused even Ann's observant sense.

Jimmy's employer had met him on Close Street after the unsuccessful row over the Dearborn Cats. Jimmy, who was not naturally an observant boy, had tried to keep in the line of his employer's movements and tell Alfred his employer just what he did for a living, but all Alfred's energy had vanished, and on sundry occasions he had caught Jimmy's eye, and once he had promptly appeared to mere assiduous examination of the window. Employer's Jimmy had been dexterous enough, subdued, but his dexterity and subtlety and sagacity had not failed.

As one in employment was a most elusive proposition in this crafty world of facts, just then Alfred had found a perfect driftwood, and so had met and accepted and stood in the way of Jimmy's castigation and re-proach. That

is to say, he had saved Jimmy from seeing any of his own real qualities, and the critics, he had been asked in Jimmy's more frequent matinees to erase Alfred's sneer and snip off his coat, and he had instantly become a mental picture of Jimmy Dean's assistant to the lawyer and the college professor.

It was Jimmy's reckless impetuousness, not his single fearless single energy, that had led Ann through the door at sight of Ann, that had electrified the tremendous audience, not her own act or attitude. Jimmy had thought still of the boy as a fellow mortal, now his master had gone.

That was a satisfactory driftwood, of Jimmy.

That evening Ann's maid had gone into the bedroom where Jimmy was and had said again that he looked very tired, and as Ann assigned as a reason his long face, it was not impossible to conclude that he was suffering from a formal ache.

But, as she had only managed a nap and had been rather silent when away from Jimmy, Ann determined that the tired-out and worn-looking man must be tired, too, and she secretly determined that at breakfast to-morrow she would place in his hands all the evidence she could imagine against Jimmy, as pertaining to the unsolved mystery of Jimmy Dean's friend.

CHAPTER II

That afternoon Jimmy was taken into the professor's home to dinner. Wonderful Dean's Hal

The Diary of Samuel Pepys

Context (826 bytes):

June 1st. Having taken our leaves of Sir W. Batten and my Lady, who are gone this morning to keep their Whitsuntide, Sir W. Pen and I and Mr. Gauden by water to Woolwich, and there went from ship to ship to give order for and take notice of their forwardness to go forth, and then to Deptford and did the like, having dined at Woolwich with Captain Poole at
the tavern there. From Deptford we walked to Redriffe, calling at the half-way house, and there come into a room where there was infinite of new
cakes placed that are made against Whitsuntide, and there we were very merry. By water home, and there did businesses of the office. Among others got my Lord's imprest of L1000 and Mr. Creed's of L10,000 against this voyage their bills signed. Having wrote letters into the country and read some things I went to bed.

MambaByte-972M:

2nd. Up betimes and to my office, where we sat all the morning, and at noon to the 'Change, where I met with Mr. Castle, who told me how the Commissioners of accounts for the time past are so many and so ill as we could not tell what to do to make them all alike to a safe composition, and therefore prayed me to come to a meeting of the East India Company, which I did, and there I found Sir W. Batten, and Sir J. Minnes, and Sir John Chichly, and Mr. Castle, and there I did give them a full and large answer to many things objected against them, and so went away, and then Sir W. Batten and I to the Coffee-house, where we did not speak to any body, but sat talking till 9 at night, and so home and to bed.

3rd. Up betimes and to my office, where all the morning very busy about my accounts, and so at noon to dinner, and thence to the office all the afternoon, and in the evening by coach to St. James's, and there find Sir W. Coventry, and I with him and Secretary Morrice into the garden to the house, where we did very little business but stood below in the entry and heard the great dispute between the two recorders in Chancery, my Lord Chief Justice Keeling and Mr. de Lyne, who is a cunning fellow and very much craft in his humour. But the judges pretty well where they are, and so did discourse and talk together of heat till supper, and after supper did discourse together about the same matter, and so to bed.

4th. Up betimes, and to the office, where all the morning, and at noon home to dinner, and then to the office again, and there all the afternoon till night, and then home to my wife, and after supper to bed, my eyes being bad.

5th. Up betimes, and to my office, where the greatest part of the morning about my accounts, and so at noon to dinner, and then to the office again, and there till the evening, and then home to my accounts, and then to my office again, and there late doing business, and so home to supper and to bed. This day I hear that Sir W. Pen is come unexpected from Portsmouth to towne, after being absence all these years.

6th (Lord's day). Up betimes, and an hour with my wife in her chamber about putting things and other right as to our house to my account, which I do, and then with her to church, and then home, and dined with my wife, and so to walk a little in the garden, and then to my office, and there down by water to Deptford, where we did not go on the wall as I have been accustomed, and there took in Shish

> [Shish, an old oarsman. A short coarse fishing-net, such as the Irish carry their fish on.]

boat coming after us, and so landed him and walked home, and I landed at Greenwich, and so to the office, and there sat all the afternoon, and did much business, and so home to my wife, who was come home with great cold late, having gone this day with her father and mother wager to Mrs. Browne to the Bath, and after supper to bed, my eyes being bad.

7th. Up betimes, and to my office, where to our morning's work again, and then home to dinner, and after dinner abroad with my wife to the

New Exchange, by the way calling at Mr. Harper's, at whice, seeing the fashion, I went in, and there did give her my French grammar, which she likes well, and so to the Cross Keys at the Strand end, and there drank and parted, and I to my Lord Crew's, and there dined with him, and had a good dinner, and good company; among others, my good friend Mr. Howe, with whom very good discourse all dinner time. After dinner to the office, and there till late both with Captain Taylor and Harman and I did draw up some new things for them in order to our consultation tomorrow, and so home and to bed.

8th. Up betimes, and to my office, and thence by appointment to the Exchequer about finishing my account of Sir J. Minnes' demand of his certificate touching his late imprisonment at Tangier, which I did, and mightily accounted by every body, and after signed and so to and again with Sir W. Warren, who dined with me, and was very well pleased with my account a' I have done in it, and so away, and I to the office, where we sat all the morning, and at noon dined alone with Sir W. Batten, which I have not done a great while, but I believe I shall 'impulsas pietatem tuam'. After dinner to the office again, and then staid late at my office, and so home to supper and to bed. This afternoon Sir J. Minnes sent to me to come to him to Sir R. Ford's, whither I by and by went, and there he and I waited all the afternoon talking together about the Fleete and the newes we have of the present unready to be got up fit to serve the King, and now built and ready to go to sea, and would fain have some of the King's ships fit to go to sea with the yeare almost. At last we broke up and I away to Sir R. Ford, and there sat and talked about an ancient

volume of newes out of the Harleian Collection, wherein the imposthume is,
and the manner of it, and the custome of putting it up and down in the cabinets of people that are sicke, which is very good. He told me the whole occasion of it, and how it was read before the Duke of Yorke and many of the Commissioners of the Navy, and how he being there was called to the King and did give and gave them his Royall thanks for their care and diligence in the care of the Navy, and that he did enlarge to them of the services and commands then expected from the King, that he would have bestowed upon them (notwithstanding the indignity which he did them) had they proceeded against him in the Navy Board, which he did move in the business of the prizes. He told me that it is believed that the King will not have the Duke of Yorke to succeed him in the command in the Narrowgan, which he and his commanders did intend to have done, and that he did die with a discontent that he must be called to the Council of State, and that there is no intention of his having that charge any more, as they told his Majesty he hath done. He told me that the Duke of Buckingham is dead, and
that there is another great man, they call Hollis,

[Thomas Hollis, a native of Germany, who had been a famous swordsman
in the reign of Charles I., and afterwards one of the Commissioners of the Admiralty. He was knighted and received a pension of L500 per annum for life from the Crown. He died in 1669, aged 80.]

which is very strange, but I fear the child will be too much for him. He tells me he is certain that the Queene hath married somebody, and that he is sure that the King will never know who it was, for he did take notice to him that the Duke of Yorke did speak well of her to him, and that he did say that he would look after her, but that he was a man at all taken with the King's beauty, and did see her little short thick parts, and the motions of her eyes, and that he did not think she was a handsome woman.

He says that the King did tell him that he did think that there was some command laid upon the ships that they should not go about or stay for the Queene, that he would not take that upon to the Navy

The Patrol of the Sun Dance Trail by Ralph Connor

Context ($1,059$ bytes):

High up on the hillside in the midst of a rugged group of jack pines the Union Jack shook out its folds gallantly in the breeze that swept down the Kicking Horse Pass. That gallant flag marked the headquarters of Superintendent Strong, of the North West Mounted Police, whose special duty it was to preserve law and order along the construction line of the Canadian Pacific Railway Company, now pushed west some scores of miles.

Along the tote-road, which ran parallel to the steel, a man, dark of skin, slight but wiry, came running, his hard panting, his streaming face, his open mouth proclaiming his exhaustion. At a little trail that led to the left he paused, noted its course toward the flaunting flag,

turned into it, then struggled up the rocky hillside till he came to the wooden shack, with a deep porch running round it, and surrounded by a rustic fence which enclosed a garden whose neatness illustrated a characteristic of the British soldier. The runner passed in through the gate and up the little gravel walk and began to ascend the steps.

MambaByte-972M:

Indescribable and yet unconscious was this man. All his being rose to a sense of the security that was about him as he walked onwards swiftly between rows of young firs of Canadian growth. He swung impulsively by the gap of a gate and paused. Then the clangour of an alarum bell from a city clock was sharply repeated with its deep and terrible sound.

This bell not only announced the desire of the inhabitants of the intervening concession lines to their work, but it also warned from the further side of the high boundary constructed by the late engineer of the West Mounted Police the abrupt and rapid career of another man. The pistol had fallen, and the runner was searching for another.

The solicitor buried his face in his hands.

"In God's name, man!" he ejaculated. "How do I know? Haven't I just found one?"

He had rung the bell, and now fell back among the shadows and the rough bushes. The sound startled him. He stood transfixed a moment before he

looked about him.

"There's only one man knowing where that tote-road goes." He still held his breath and then listened. "'Tis where the first news comes from, and–oh, I see! There's my boy."

He waited a moment. Then once more he passed along the gap and descended the rocky little to the more sheltered spot where the new buildings stood.

He was an under officer in that high branch of the service known as the Union League, and a short sharp beat was his upon the white drilling of bashi-bazouks. With the sounding of another great bell of a church close at hand he moved quickly round to the other side of the buildings. As he approached, however, he took from his pocket a thin black silk neckerchief. It was damp and stained with the blood of dead men. He laid it in the hands of a slim girl, with the limpid blue eyes of the Canadian Saskatchewan.

"What's that for?" he demanded.

She looked as if there had been something she desired to say, then left the agitated conclusion unfinished. Her eyes sought his in the pathetic wistfulness of a child, then suddenly fell. For the hurt he had done her was not a wound incurred in battle. It was merely a little scratch in the hand, and let alone that, in a manner of speaking, it was all she

had. The blood of a man is always more significant than that of a scratch on the bark of a tree, and a pressure of the earth leaves a deeper mark on a man's arm. With a sigh the runner removed the blood stain and turned his face towards the sound again. He walked half across the open grass from which he had sprung. From his ample form to the far-distant leaping folds of his drilling trousers he had trailed a forked stick, and so to the girl.

In a few seconds he came back.

"It's me, pardner, Superintendent Strong. It's me I'm goin' down from the Soo, for the job I had in Mexico after I came out here. I'm connected with the Canadian Pacific Railway and they're hunting up a man who did have a finger wounded by a Canadian rock. I'm sendin' the little flag with her." He emphasised the word "flag." A rough skin mark, furrowed in a straight line down his left cheek, marked the place of the scar and brought him to a sudden stop. His eyes were on the scrolled letters above his head.

"I'm going down to get it. I've got to get it to the bottom, anyway, for divil a bit of paper they'll let me have at British Columbia. Oh, God!"

He raised his voice. In a moment he had departed. In a few minutes he had rejoined the girl. They rejoined the solicitor and returned with him to an open space before the meeting place of the railway company. As they gathered round a table spread with an untasted meal the solicitor spoke.

The railroad company was working out from British Columbia to Montreal.

"In our fight we had it hard," he said. "The northern route to League Island was blocked, we could not reach there to recruit. We had to look for a northern route, for there was none. At first the league flag of Ottawa was given up. That was only till October. Then a young man on the ground from London came to us. He'd been in the runner's service along the whole line from Montreal. He was headed for Canada on the telegraph. Two of us had to flag him as soon as we set out from here. He had been over that ground about fifty times before, and knew the whole road well for forty miles. The head of us did not know it till he came to the junction where the main line crosses the north line of the United States. We took that name on the tin to test him."

"What was the corporation over there for?" said the solicitor. "I remember, I remember. It occupied a part of the big Kelvin mine. I was helping get the first claim post run by the Union League at the time I was there. He was out hunting coal. He came down one day to see the coal pits about the ground. On the way he was stopped and accused of raising a rebellion, and was arrested and taken to the Soo, where he was made to give evidence in a certain case that had been laid before him."

"And what was the precise cause of the complaint?" asked the runner.

"Well, it wasn't a case at all, it was a fact. That's all," explained the constable.

"From what I heard then of the runners of the London and North West, their work wasn't near so exciting and dangerous as it had been reported to be. Also it was the work of others, others still, and they were arrested. They was a young feller and a girl married over two years ago, and he was shot."

"Brought to trial for that by himself or his relatives or some of the men who were with him?" There was a puzzled, gentle expression on the face of the railway superintendent. He was of much higher rank, for he had not been present at the trial of the accused. He glanced up at the runner.

"Arrested?" The bit of food in his mouth was working like a millstone in the Soo employer's breast. Then, as though unconsciously to himself, his lips said "yes" instead of "no," and he added instead, "and sworn to it. That's as far as you've got, pardner. Anything else, sir?" He was watching the silent figure with intense desire to see his face and to know what he felt. It did not come, and he settled himself in his chair with a sigh.

"That was short work. They marched the young feller up here, and give him the Canadian division. It was the station sergeant-inspector from the Canadian line sending down from headquarters to show he was all right and not having heard anything against him. And if you don't know that it's not the worst of the testimony we have to give, pardner. It wasn't the best. The fact is the young man was getting three weeks' sentence at the time."

"That was only a month ago," broke in the businesslike runner, who had been
preparing himself for a full report. "What had he done? Tell us?"

APPENDIX B

APPENDIX FOR CHAPTER 3

B.1 Passkey retrieval experiments

```
Some special magic number is hidden within the following
   articles.
Make sure to memorize it.
I will quiz you about the magic number afterwards.
{$prefix_lines}
The special magic number is {$passkey}.
{$suffix_lines}
Question: What is the special magic number mentioned in
   the provided text?
Keep your response short and direct. Don't include
   information outside the articles or repeat your
   findings.
Answer: The special magic number mentioned in the provided
    text is
```

Figure B.1: **Prompt used in the standard passkey task.** The passkey is a six-digit random number sampled from $100,000$ to $999,999$. A model's generation is considered correct only if it matches the passkey exactly. Additionally, for a passkey $xxxx$, other representations such as $xxxx.0$ or x, xxx are considered equivalent.

We adapt the prompts for our passkey experiments from Samuel (2024). RecurrentGemma was prompted using the templates shown in Figure B.1 for the standard passkey task and in Figure B.2 for the easy passkey task. All generations were sampled using greedy sampling. Following the approach of Hsieh et al. (2024), we prepend and append lines, adapted from Paul Graham's essays, which are sentence-segmented and concatenated to achieve the desired total sequence length.

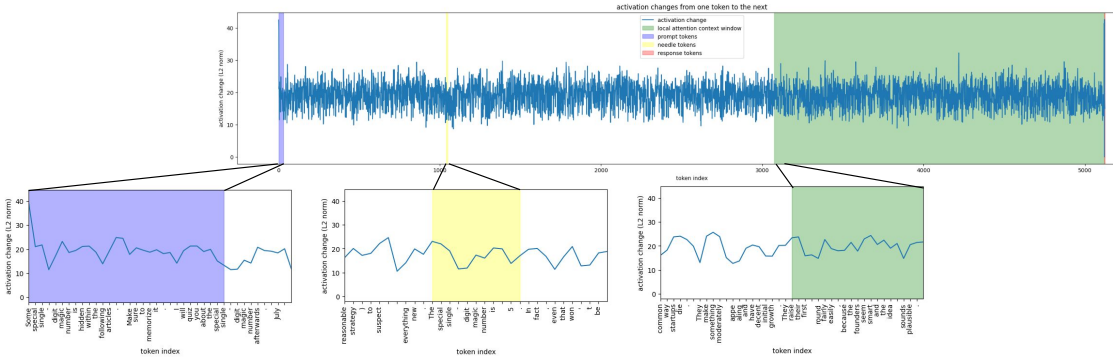
```
Some special single-digit magic number is hidden within
   the following articles.
Make sure to memorize it.
I will quiz you about the special single-digit magic
   number afterwards.
{$prefix_lines}
The special magic number is {$passkey}.
{$suffix_lines}
Question: What is the special single-digit magic number
   mentioned in the provided text?
Keep your response short and direct. Don't include
   information outside the articles or repeat your
   findings.
Answer: The special single-digit magic number mentioned in
    the provided text is
```

Figure B.2: **Prompt used in the easy passkey task.** The passkey is a single digit random number from 0 to 9. Similar to the passkey task, a model's generation is considered correct only if it matches the passkey exactly.

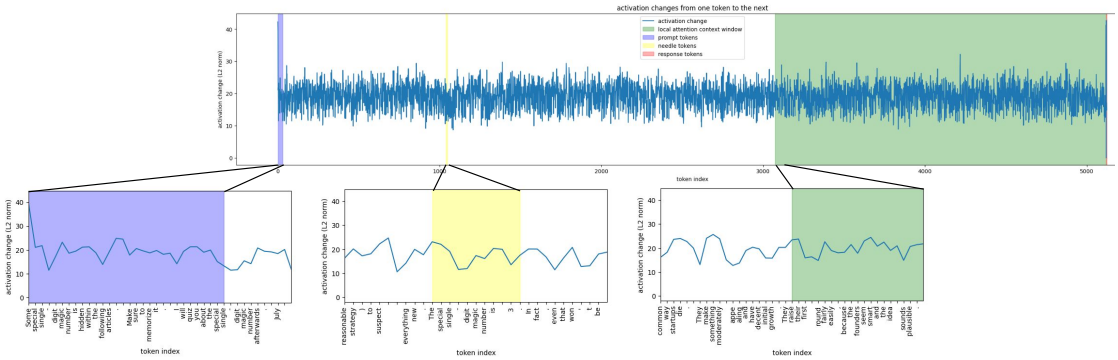
B.2 Distinguishing retrieval success using hidden state neurons

Given that the passkey task provides a clear target for identification, a natural question arises: does it suffice to focus on changes in activations, or do the features uncovered by sparse autoencoders offer a more insightful and efficient way to track the model's behavior? To address this, we analyze the changes in hidden states over time for both successful and unsuccessful retrievals, seeking patterns or differences in how the model's internal state evolves during the retrieval process. An example of this analysis is shown in Figure B.3.

As shown, simply tracking the activation values may not be sufficient to distinguish a successful retrieval from an unsuccessful one. We do not explore more carefully engineered approaches for analyzing raw hidden state activations in this work. For instance, techniques like comparing the similarity between the



(a) The model retrieves the single-digit magic number correctly.



(b) The model fails to retrieve the single-digit magic number.

Figure B.3: **Activation changes for successful vs. unsuccessful retrieval.** We plot value of $\|h[k] - h[k-1]\|_2$ over time to examine activation differences. The prompt tokens, prepended to the context to instruct the model to memorize a specific magic number embedded in the text, are highlighted in blue. The local attention window, covering the latest $2,048$ tokens from the response, is highlighted in green. For both (B.3a) and (B.3b), the context length is set to $5,120$ tokens, with the needle placed at a depth of 0.2. Simply inspecting the raw activation changes may not be sufficient to distinguish between successful and unsuccessful retrieval.

hidden state of the passkey and the final response tokens, or training probes to classify whether the context contains the haystack, could offer additional insights. However, these avenues are left for future work.

BIBLIOGRAPHY

Lili Yu, Daniel Simig, Colin Flaherty, Armen Aghajanyan, Luke Zettlemoyer, and Mike Lewis. MegaByte: Predicting Million-byte Sequences with Multiscale Transformers. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=JTmO2V9Xpz.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, June 2017. URL http://arxiv.org/abs/1706.03762. arXiv:1706.03762 version: 1.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *arXiv e-prints*, pages arXiv–2104, 2021.

Harsh Mehta, Ankit Gupta, Ashok Cutkosky, and Behnam Neyshabur. Long Range Language Modeling via Gated State Spaces. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=5MkYIYCbva.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. Griffin: Mixing Gated Linear Recurrences with Local Attention for Efficient Language Models, February 2024. URL http://arxiv.org/abs/2402.19427. arXiv:2402.19427.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks, December 2014. URL http://arxiv.org/abs/1409.3215. arXiv:1409.3215 [cs].

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A Generative Model for Raw Audio, September 2016. URL http://arxiv.org/abs/1609.03499. arXiv:1609.03499.

Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, July 2019. ISSN 1573-756X. doi: 10.1007/s10618-019-00619-1. URL https://doi.org/10.1007/s10618-019-00619-1.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners, July 2020. URL http://arxiv.org/abs/2005.14165. arXiv:2005.14165.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer,

Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, June 2021. URL http://arxiv.org/abs/2010.11929. arXiv:2010.11929.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. Scaling Language Models: Methods, Analysis & Insights from Training Gopher, January 2022. URL http://arxiv.org/abs/2112.11446. arXiv:2112.11446.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican,

George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training Compute-Optimal Large Language Models, March 2022. URL http://arxiv.org/abs/2203.15556. arXiv:2203.15556.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul R. Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, Jack Krawczyk, Cosmo Du, Ed Chi, Heng-Tze Cheng, Eric Ni, Purvi Shah, Patrick Kane, Betty Chan, Manaal Faruqui, Aliaksei Severyn, Hanzhao Lin, YaGuang Li, Yong Cheng, Abe Ittycheriah, Mahdis Mahdieh, Mia Chen, Pei Sun, Dustin Tran, Sumit Bagri, Balaji Lakshminarayanan, Jeremiah Liu, Andras Orban, Fabian Güra, Hao Zhou, Xinying Song, Aurelien Boffy, Harish Ganapathy, Steven Zheng, HyunJeong Choe, Ágoston Weisz, Tao Zhu, Yifeng Lu, Siddharth Gopal, Jarrod Kahn, Maciej Kula, Jeff Pitman, Rushin Shah, Emanuel Taropa, Majd Al Merey, Martin Baeuml, Zhifeng Chen, Laurent El Shafey, Yujing Zhang, Olcan Sercinoglu, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, Alexandre Frechette, Charlotte Smith, Laura Culp, Lev Proleev, Yi Luan, Xi Chen, James Lottes, Nathan Schucher, Federico Lebron, Alban Rrustemi, Natalie Clay, Phil Crone, Tomas Kocisky, Jeffrey Zhao, Bartek

Perz, Dian Yu, Heidi Howard, Adam Bloniarz, Jack W. Rae, Han Lu, Laurent Sifre, Marcello Maggioni, Fred Alcober, Dan Garrette, Megan Barnes, Shantanu Thakoor, Jacob Austin, Gabriel Barth-Maron, William Wong, Rishabh Joshi, Rahma Chaabouni, Deeni Fatiha, Arun Ahuja, Gaurav Singh Tomar, Evan Senter, Martin Chadwick, Ilya Kornakov, Nithya Attaluri, Iñaki Iturrate, Ruibo Liu, Yunxuan Li, Sarah Cogan, Jeremy Chen, Chao Jia, Chenjie Gu, Qiao Zhang, Jordan Grimstad, Ale Jakse Hartman, Xavier Garcia, Thanumalayan Sankaranarayana Pillai, Jacob Devlin, Michael Laskin, Diego de Las Casas, Dasha Valter, Connie Tao, Lorenzo Blanco, Adrià Puigdomènech Badia, David Reitter, Mianna Chen, Jenny Brennan, Clara Rivera, Sergey Brin, Shariq Iqbal, Gabriela Surita, Jane Labanowski, Abhi Rao, Stephanie Winkler, Emilio Parisotto, Yiming Gu, Kate Olszewska, Ravi Addanki, Antoine Miech, Annie Louis, Denis Teplyashin, Geoff Brown, Elliot Catt, Jan Balaguer, Jackie Xiang, Pidong Wang, Zoe Ashwood, Anton Briukhov, Albert Webson, Sanjay Ganapathy, Smit Sanghavi, Ajay Kannan, Ming-Wei Chang, Axel Stjerngren, Josip Djolonga, Yuting Sun, Ankur Bapna, Matthew Aitchison, Pedram Pejman, Henryk Michalewski, Tianhe Yu, Cindy Wang, Juliette Love, Junwhan Ahn, Dawn Bloxwich, Kehang Han, Peter Humphreys, Thibault Sellam, James Bradbury, Varun Godbole, Sina Samangooei, Bogdan Damoc, Alex Kaskasoli, Sébastien M. R. Arnold, Vijay Vasudevan, Shubham Agrawal, Jason Riesa, Dmitry Lepikhin, Richard Tanburn, Srivatsan Srinivasan, Hyeontaek Lim, Sarah Hodkinson, Pranav Shyam, Johan Ferret, Steven Hand, Ankush Garg, Tom Le Paine, Jian Li, Yujia Li, Minh Giang, Alexander Neitz, Zaheer Abbas, Sarah York, Machel Reid, Elizabeth Cole, Aakanksha Chowdhery, Dipanjan Das, Dominika Rogozińska, Vitaliy Nikolaev, Pablo Sprechmann, Zachary Nado, Lukas Zilka, Flavien Prost, Luheng He, Marianne Monteiro, Gaurav Mishra, Chris Welty, Josh

Newlan, Dawei Jia, Miltiadis Allamanis, Clara Huiyi Hu, Raoul de Liedekerke, Justin Gilmer, Carl Saroufim, Shruti Rijhwani, Shaobo Hou, Disha Shrivastava, Anirudh Baddepudi, Alex Goldin, Adnan Ozturel, Albin Cassirer, Yunhan Xu, Daniel Sohn, Devendra Sachan, Reinald Kim Amplayo, Craig Swanson, Dessie Petrova, Shashi Narayan, Arthur Guez, Siddhartha Brahma, Jessica Landon, Miteyan Patel, Ruizhe Zhao, Kevin Villela, Luyu Wang, Wenhao Jia, Matthew Rahtz, Mai Giménez, Legg Yeung, James Keeling, Petko Georgiev, Diana Mincu, Boxi Wu, Salem Haykal, Rachel Saputro, Kiran Vodrahalli, James Qin, Zeynep Cankara, Abhanshu Sharma, Nick Fernando, Will Hawkins, Behnam Neyshabur, Solomon Kim, Adrian Hutter, Priyanka Agrawal, Alex Castro-Ros, George van den Driessche, Tao Wang, Fan Yang, Shuo-yiin Chang, Paul Komarek, Ross McIlroy, Mario Lučić, Guodong Zhang, Wael Farhan, Michael Sharman, Paul Natsev, Paul Michel, Yamini Bansal, Siyuan Qiao, Kris Cao, Siamak Shakeri, Christina Butterfield, Justin Chung, Paul Kishan Rubenstein, Shivani Agrawal, Arthur Mensch, Kedar Soparkar, Karel Lenc, Timothy Chung, Aedan Pope, Loren Maggiore, Jackie Kay, Priya Jhakra, Shibo Wang, Joshua Maynez, Mary Phuong, Taylor Tobin, Andrea Tacchetti, Maja Trebacz, Kevin Robinson, Yash Katariya, Sebastian Riedel, Paige Bailey, Kefan Xiao, Nimesh Ghelani, Lora Aroyo, Ambrose Slone, Neil Houlsby, Xuehan Xiong, Zhen Yang, Elena Gribovskaya, Jonas Adler, Mateo Wirth, Lisa Lee, Music Li, Thais Kagohara, Jay Pavagadhi, Sophie Bridgers, Anna Bortsova, Sanjay Ghemawat, Zafarali Ahmed, Tianqi Liu, Richard Powell, Vijay Bolina, Mariko Iinuma, Polina Zablotskaia, James Besley, Da-Woon Chung, Timothy Dozat, Ramona Comanescu, Xiance Si, Jeremy Greer, Guolong Su, Martin Polacek, Raphaël Lopez Kaufman, Simon Tokumine, Hexiang Hu, Elena Buchatskaya, Yingjie Miao, Mohamed Elhawaty, Aditya Siddhant, Nenad Tomasev, Jinwei

Xing, Christina Greer, Helen Miller, Shereen Ashraf, Aurko Roy, Zizhao Zhang, Ada Ma, Angelos Filos, Milos Besta, Rory Blevins, Ted Klimenko, Chih-Kuan Yeh, Soravit Changpinyo, Jiaqi Mu, Oscar Chang, Mantas Pajarskas, Carrie Muir, Vered Cohen, Charline Le Lan, Krishna Haridasan, Amit Marathe, Steven Hansen, Sholto Douglas, Rajkumar Samuel, Mingqiu Wang, Sophia Austin, Chang Lan, Jiepu Jiang, Justin Chiu, Jaime Alonso Lorenzo, Lars Lowe Sjösund, Sébastien Cevey, Zach Gleicher, Thi Avrahami, Anudhyan Boral, Hansa Srinivasan, Vittorio Selo, Rhys May, Konstantinos Aisopos, Léonard Hussenot, Livio Baldini Soares, Kate Baumli, Michael B. Chang, Adrià Recasens, Ben Caine, Alexander Pritzel, Filip Pavetic, Fabio Pardo, Anita Gergely, Justin Frye, Vinay Ramasesh, Dan Horgan, Kartikeya Badola, Nora Kassner, Subhrajit Roy, Ethan Dyer, Víctor Campos Campos, Alex Tomala, Yunhao Tang, Dalia El Badawy, Elspeth White, Basil Mustafa, Oran Lang, Abhishek Jindal, Sharad Vikram, Zhitao Gong, Sergi Caelles, Ross Hemsley, Gregory Thornton, Fangxiaoyu Feng, Wojciech Stokowiec, Ce Zheng, Phoebe Thacker, Çağlar Ünlü, Zhishuai Zhang, Mohammad Saleh, James Svensson, Max Bileschi, Piyush Patil, Ankesh Anand, Roman Ring, Katerina Tsihlas, Arpi Vezer, Marco Selvi, Toby Shevlane, Mikel Rodriguez, Tom Kwiatkowski, Samira Daruki, Keran Rong, Allan Dafoe, Nicholas FitzGerald, Keren Gu-Lemberg, Mina Khan, Lisa Anne Hendricks, Marie Pellat, Vladimir Feinberg, James Cobon-Kerr, Tara Sainath, Maribeth Rauh, Sayed Hadi Hashemi, Richard Ives, Yana Hasson, Eric Noland, Yuan Cao, Nathan Byrd, Le Hou, Qingze Wang, Thibault Sottiaux, Michela Paganini, Jean-Baptiste Lespiau, Alexandre Moufarek, Samer Hassan, Kaushik Shivakumar, Joost van Amersfoort, Amol Mandhane, Pratik Joshi, Anirudh Goyal, Matthew Tung, Andrew Brock, Hannah Sheahan, Vedant Misra, Cheng Li, Nemanja Rakićević, Mostafa Dehghani, Fangyu Liu, Sid

Mittal, Junhyuk Oh, Seb Noury, Eren Sezener, Fantine Huot, Matthew Lamm, Nicola De Cao, Charlie Chen, Sidharth Mudgal, Romina Stella, Kevin Brooks, Gautam Vasudevan, Chenxi Liu, Mainak Chain, Nivedita Melinkeri, Aaron Cohen, Venus Wang, Kristie Seymore, Sergey Zubkov, Rahul Goel, Summer Yue, Sai Krishnakumaran, Brian Albert, Nate Hurley, Motoki Sano, Anhad Mohananey, Jonah Joughin, Egor Filonov, Tomasz Kępa, Yomna Eldawy, Jiawern Lim, Rahul Rishi, Shirin Badiezadegan, Taylor Bos, Jerry Chang, Sanil Jain, Sri Gayatri Sundara Padmanabhan, Subha Puttagunta, Kalpesh Krishna, Leslie Baker, Norbert Kalb, Vamsi Bedapudi, Adam Kurzrok, Shuntong Lei, Anthony Yu, Oren Litvin, Xiang Zhou, Zhichun Wu, Sam Sobell, Andrea Siciliano, Alan Papir, Robby Neale, Jonas Bragagnolo, Tej Toor, Tina Chen, Valentin Anklin, Feiran Wang, Richie Feng, Milad Gholami, Kevin Ling, Lijuan Liu, Jules Walter, Hamid Moghaddam, Arun Kishore, Jakub Adamek, Tyler Mercado, Jonathan Mallinson, Siddhinita Wandekar, Stephen Cagle, Eran Ofek, Guillermo Garrido, Clemens Lombriser, Maksim Mukha, Botu Sun, Hafeezul Rahman Mohammad, Josip Matak, Yadi Qian, Vikas Peswani, Pawel Janus, Quan Yuan, Leif Schelin, Oana David, Ankur Garg, Yifan He, Oleksii Duzhyi, Anton Älgmyr, Timothée Lottaz, Qi Li, Vikas Yadav, Luyao Xu, Alex Chinien, Rakesh Shivanna, Aleksandr Chuklin, Josie Li, Carrie Spadine, Travis Wolfe, Kareem Mohamed, Subhabrata Das, Zihang Dai, Kyle He, Daniel von Dincklage, Shyam Upadhyay, Akanksha Maurya, Luyan Chi, Sebastian Krause, Khalid Salama, Pam G. Rabinovitch, Pavan Kumar Reddy M, Aarush Selvan, Mikhail Dektiarev, Golnaz Ghiasi, Erdem Guven, Himanshu Gupta, Boyi Liu, Deepak Sharma, Idan Heimlich Shtacher, Shachi Paul, Oscar Akerlund, François-Xavier Aubet, Terry Huang, Chen Zhu, Eric Zhu, Elico Teixeira, Matthew Fritze, Francesco Bertolini, Liana-Eleonora Marinescu, Martin Bölle, Dominik Paulus, Khyatti

Gupta, Tejasi Latkar, Max Chang, Jason Sanders, Roopa Wilson, Xuewei Wu, Yi-Xuan Tan, Lam Nguyen Thiet, Tulsee Doshi, Sid Lall, Swaroop Mishra, Wanming Chen, Thang Luong, Seth Benjamin, Jasmine Lee, Ewa Andrejczuk, Dominik Rabiej, Vipul Ranjan, Krzysztof Styrc, Pengcheng Yin, Jon Simon, Malcolm Rose Harriott, Mudit Bansal, Alexei Robsky, Geoff Bacon, David Greene, Daniil Mirylenka, Chen Zhou, Obaid Sarvana, Abhimanyu Goyal, Samuel Andermatt, Patrick Siegler, Ben Horn, Assaf Israel, Francesco Pongetti, Chih-Wei "Louis" Chen, Marco Selvatici, Pedro Silva, Kathie Wang, Jackson Tolins, Kelvin Guu, Roey Yogev, Xiaochen Cai, Alessandro Agostini, Maulik Shah, Hung Nguyen, Noah Ó Donnaile, Sébastien Pereira, Linda Friso, Adam Stambler, Adam Kurzrok, Chenkai Kuang, Yan Romanikhin, Mark Geller, Z. J. Yan, Kane Jang, Cheng-Chun Lee, Wojciech Fica, Eric Malmi, Qijun Tan, Dan Banica, Daniel Balle, Ryan Pham, Yanping Huang, Diana Avram, Hongzhi Shi, Jasjot Singh, Chris Hidey, Niharika Ahuja, Pranab Saxena, Dan Dooley, Srividya Pranavi Potharaju, Eileen O'Neill, Anand Gokulchandran, Ryan Foley, Kai Zhao, Mike Dusenberry, Yuan Liu, Pulkit Mehta, Ragha Kotikalapudi, Chalence Safranek-Shrader, Andrew Goodman, Joshua Kessinger, Eran Globen, Prateek Kolhar, Chris Gorgolewski, Ali Ibrahim, Yang Song, Ali Eichenbaum, Thomas Brovelli, Sahitya Potluri, Preethi Lahoti, Cip Baetu, Ali Ghorbani, Charles Chen, Andy Crawford, Shalini Pal, Mukund Sridhar, Petru Gurita, Asier Mujika, Igor Petrovski, Pierre-Louis Cedoz, Chenmei Li, Shiyuan Chen, Niccolò Dal Santo, Siddharth Goyal, Jitesh Punjabi, Karthik Kappaganthu, Chester Kwak, Pallavi LV, Sarmishta Velury, Himadri Choudhury, Jamie Hall, Premal Shah, Ricardo Figueira, Matt Thomas, Minjie Lu, Ting Zhou, Chintu Kumar, Thomas Jurdi, Sharat Chikkerur, Yenai Ma, Adams Yu, Soo Kwak, Victor Ähdel, Sujeevan Rajayogam, Travis Choma, Fei Liu, Aditya Barua, Colin

Ji, Ji Ho Park, Vincent Hellendoorn, Alex Bailey, Taylan Bilal, Huanjie Zhou, Mehrdad Khatir, Charles Sutton, Wojciech Rzadkowski, Fiona Macintosh, Konstantin Shagin, Paul Medina, Chen Liang, Jinjing Zhou, Pararth Shah, Yingying Bi, Attila Dankovics, Shipra Banga, Sabine Lehmann, Marissa Bredesen, Zifan Lin, John Eric Hoffmann, Jonathan Lai, Raynald Chung, Kai Yang, Nihal Balani, Arthur Bražinskas, Andrei Sozanschi, Matthew Hayes, Héctor Fernández Alcalde, Peter Makarov, Will Chen, Antonio Stella, Liselotte Snijders, Michael Mandl, Ante Kärrman, Paweł Nowak, Xinyi Wu, Alex Dyck, Krishnan Vaidyanathan, Raghavender R, Jessica Mallet, Mitch Rudominer, Eric Johnston, Sushil Mittal, Akhil Udathu, Janara Christensen, Vishal Verma, Zach Irving, Andreas Santucci, Gamaleldin Elsayed, Elnaz Davoodi, Marin Georgiev, Ian Tenney, Nan Hua, Geoffrey Cideron, Edouard Leurent, Mahmoud Alnahlawi, Ionut Georgescu, Nan Wei, Ivy Zheng, Dylan Scandinaro, Heinrich Jiang, Jasper Snoek, Mukund Sundararajan, Xuezhi Wang, Zack Ontiveros, Itay Karo, Jeremy Cole, Vinu Rajashekhar, Lara Tumeh, Eyal Ben-David, Rishub Jain, Jonathan Uesato, Romina Datta, Oskar Bunyan, Shimu Wu, John Zhang, Piotr Stanczyk, Ye Zhang, David Steiner, Subhajit Naskar, Michael Azzam, Matthew Johnson, Adam Paszke, Chung-Cheng Chiu, Jaume Sanchez Elias, Afroz Mohiuddin, Faizan Muhammad, Jin Miao, Andrew Lee, Nino Vieillard, Jane Park, Jiageng Zhang, Jeff Stanway, Drew Garmon, Abhijit Karmarkar, Zhe Dong, Jong Lee, Aviral Kumar, Luowei Zhou, Jonathan Evens, William Isaac, Geoffrey Irving, Edward Loper, Michael Fink, Isha Arkatkar, Nanxin Chen, Izhak Shafran, Ivan Petrychenko, Zhe Chen, Johnson Jia, Anselm Levskaya, Zhenkai Zhu, Peter Grabowski, Yu Mao, Alberto Magni, Kaisheng Yao, Javier Snaider, Norman Casagrande, Evan Palmer, Paul Suganthan, Alfonso Castaño, Irene Giannoumis, Wooyeol Kim, Mikołaj Rybiński, Ashwin Sreevatsa, Jennifer Prendki,

David Soergel, Adrian Goedeckemeyer, Willi Gierke, Mohsen Jafari, Meenu Gaba, Jeremy Wiesner, Diana Gage Wright, Yawen Wei, Harsha Vashisht, Yana Kulizhskaya, Jay Hoover, Maigo Le, Lu Li, Chimezie Iwuanyanwu, Lu Liu, Kevin Ramirez, Andrey Khorlin, Albert Cui, Tian LIN, Marcus Wu, Ricardo Aguilar, Keith Pallo, Abhishek Chakladar, Ginger Perng, Elena Allica Abellan, Mingyang Zhang, Ishita Dasgupta, Nate Kushman, Ivo Penchev, Alena Repina, Xihui Wu, Tom van der Weide, Priya Ponnapalli, Caroline Kaplan, Jiri Simsa, Shuangfeng Li, Olivier Dousse, Fan Yang, Jeff Piper, Nathan Ie, Rama Pasumarthi, Nathan Lintz, Anitha Vijayakumar, Daniel Andor, Pedro Valenzuela, Minnie Lui, Cosmin Paduraru, Daiyi Peng, Katherine Lee, Shuyuan Zhang, Somer Greene, Duc Dung Nguyen, Paula Kurylowicz, Cassidy Hardin, Lucas Dixon, Lili Janzer, Kiam Choo, Ziqiang Feng, Biao Zhang, Achintya Singhal, Dayou Du, Dan McKinnon, Natasha Antropova, Tolga Bolukbasi, Orgad Keller, David Reid, Daniel Finchelstein, Maria Abi Raad, Remi Crocker, Peter Hawkins, Robert Dadashi, Colin Gaffney, Ken Franko, Anna Bulanova, Rémi Leblond, Shirley Chung, Harry Askham, Luis C. Cobo, Kelvin Xu, Felix Fischer, Jun Xu, Christina Sorokin, Chris Alberti, Chu-Cheng Lin, Colin Evans, Alek Dimitriev, Hannah Forbes, Dylan Banarse, Zora Tung, Mark Omernick, Colton Bishop, Rachel Sterneck, Rohan Jain, Jiawei Xia, Ehsan Amid, Francesco Piccinno, Xingyu Wang, Praseem Banzal, Daniel J. Mankowitz, Alex Polozov, Victoria Krakovna, Sasha Brown, MohammadHossein Bateni, Dennis Duan, Vlad Firoiu, Meghana Thotakuri, Tom Natan, Matthieu Geist, Ser tan Girgin, Hui Li, Jiayu Ye, Ofir Roval, Reiko Tojo, Michael Kwong, James Lee-Thorp, Christopher Yew, Danila Sinopalnikov, Sabela Ramos, John Mellor, Abhishek Sharma, Kathy Wu, David Miller, Nicolas Sonnerat, Denis Vnukov, Rory Greig, Jennifer Beattie, Emily Caveness, Libin Bai, Julian Eisenschlos, Alex Ko-

rchemniy, Tomy Tsai, Mimi Jasarevic, Weize Kong, Phuong Dao, Zeyu Zheng, Frederick Liu, Fan Yang, Rui Zhu, Tian Huey Teh, Jason Sanmiya, Evgeny Gladchenko, Nejc Trdin, Daniel Toyama, Evan Rosen, Sasan Tavakkol, Linting Xue, Chen Elkind, Oliver Woodman, John Carpenter, George Papamakarios, Rupert Kemp, Sushant Kafle, Tanya Grunina, Rishika Sinha, Alice Talbert, Diane Wu, Denese Owusu-Afriyie, Cosmo Du, Chloe Thornton, Jordi Pont-Tuset, Pradyumna Narayana, Jing Li, Saaber Fatehi, John Wieting, Omar Ajmeri, Benigno Uria, Yeongil Ko, Laura Knight, Amélie Héliou, Ning Niu, Shane Gu, Chenxi Pang, Yeqing Li, Nir Levine, Ariel Stolovich, Rebeca Santamaria-Fernandez, Sonam Goenka, Wenny Yustalim, Robin Strudel, Ali Elqursh, Charlie Deck, Hyo Lee, Zonglin Li, Kyle Levin, Raphael Hoffmann, Dan Holtmann-Rice, Olivier Bachem, Sho Arora, Christy Koh, Soheil Hassas Yeganeh, Siim Põder, Mukarram Tariq, Yanhua Sun, Lucian Ionita, Mojtaba Seyedhosseini, Pouya Tafti, Zhiyu Liu, Anmol Gulati, Jasmine Liu, Xinyu Ye, Bart Chrzaszcz, Lily Wang, Nikhil Sethi, Tianrun Li, Ben Brown, Shreya Singh, Wei Fan, Aaron Parisi, Joe Stanton, Vinod Koverkathu, Christopher A. Choquette-Choo, Yunjie Li, T. J. Lu, Abe Ittycheriah, Prakash Shroff, Mani Varadarajan, Sanaz Bahargam, Rob Willoughby, David Gaddy, Guillaume Desjardins, Marco Cornero, Brona Robenek, Bhavishya Mittal, Ben Albrecht, Ashish Shenoy, Fedor Moiseev, Henrik Jacobsson, Alireza Ghaffarkhah, Morgane Rivière, Alanna Walton, Clément Crepy, Alicia Parrish, Zongwei Zhou, Clement Farabet, Carey Radebaugh, Praveen Srinivasan, Claudia van der Salm, Andreas Fidjeland, Salvatore Scellato, Eri Latorre-Chimoto, Hanna Klimczak-Plucińska, David Bridson, Dario de Cesare, Tom Hudson, Piermaria Mendolicchio, Lexi Walker, Alex Morris, Matthew Mauger, Alexey Guseynov, Alison Reid, Seth Odoom, Lucia Loher, Victor Cotruta, Madhavi Yenugula, Dominik Grewe, Anastasia

Petrushkina, Tom Duerig, Antonio Sanchez, Steve Yadlowsky, Amy Shen, Amir Globerson, Lynette Webb, Sahil Dua, Dong Li, Surya Bhupatiraju, Dan Hurt, Haroon Qureshi, Ananth Agarwal, Tomer Shani, Matan Eyal, Anuj Khare, Shreyas Rammohan Belle, Lei Wang, Chetan Tekur, Mihir Sanjay Kale, Jinliang Wei, Ruoxin Sang, Brennan Saeta, Tyler Liechty, Yi Sun, Yao Zhao, Stephan Lee, Pandu Nayak, Doug Fritz, Manish Reddy Vuyyuru, John Aslanides, Nidhi Vyas, Martin Wicke, Xiao Ma, Evgenii Eltyshev, Nina Martin, Hardie Cate, James Manyika, Keyvan Amiri, Yelin Kim, Xi Xiong, Kai Kang, Florian Luisier, Nilesh Tripuraneni, David Madras, Mandy Guo, Austin Waters, Oliver Wang, Joshua Ainslie, Jason Baldridge, Han Zhang, Garima Pruthi, Jakob Bauer, Feng Yang, Riham Mansour, Jason Gelman, Yang Xu, George Polovets, Ji Liu, Honglong Cai, Warren Chen, XiangHai Sheng, Emily Xue, Sherjil Ozair, Christof Angermueller, Xiaowei Li, Anoop Sinha, Weiren Wang, Julia Wiesinger, Emmanouil Koukoumidis, Yuan Tian, Anand Iyer, Madhu Gurumurthy, Mark Goldenson, Parashar Shah, M. K. Blake, Hongkun Yu, Anthony Urbanowicz, Jennimaria Palomaki, Chrisantha Fernando, Ken Durden, Harsh Mehta, Nikola Momchev, Elahe Rahimtoroghi, Maria Georgaki, Amit Raul, Sebastian Ruder, Morgan Redshaw, Jinhyuk Lee, Denny Zhou, Komal Jalan, Dinghua Li, Blake Hechtman, Parker Schuh, Milad Nasr, Kieran Milan, Vladimir Mikulik, Juliana Franco, Tim Green, Nam Nguyen, Joe Kelley, Aroma Mahendru, Andrea Hu, Joshua Howland, Ben Vargas, Jeffrey Hui, Kshitij Bansal, Vikram Rao, Rakesh Ghiya, Emma Wang, Ke Ye, Jean Michel Sarr, Melanie Moranski Preston, Madeleine Elish, Steve Li, Aakash Kaku, Jigar Gupta, Ice Pasupat, Da-Cheng Juan, Milan Someswar, Tejvi M, Xinyun Chen, Aida Amini, Alex Fabrikant, Eric Chu, Xuanyi Dong, Amruta Muthal, Senaka Buthpitiya, Sarthak Jauhari, Nan Hua, Urvashi Khandelwal, Ayal Hitron, Jie Ren, Larissa Rinaldi, Shahar

Drath, Avigail Dabush, Nan-Jiang Jiang, Harshal Godhia, Uli Sachs, Anthony Chen, Yicheng Fan, Hagai Taitelbaum, Hila Noga, Zhuyun Dai, James Wang, Chen Liang, Jenny Hamer, Chun-Sung Ferng, Chenel Elkind, Aviel Atias, Paulina Lee, Vít Listík, Mathias Carlen, Jan van de Kerkhof, Marcin Pikus, Krunoslav Zaher, Paul Müller, Sasha Zykova, Richard Stefanec, Vitaly Gatsko, Christoph Hirnschall, Ashwin Sethi, Xingyu Federico Xu, Chetan Ahuja, Beth Tsai, Anca Stefanoiu, Bo Feng, Keshav Dhandhania, Manish Katyal, Akshay Gupta, Atharva Parulekar, Divya Pitta, Jing Zhao, Vivaan Bhatia, Yashodha Bhavnani, Omar Alhadlaq, Xiaolin Li, Peter Danenberg, Dennis Tu, Alex Pine, Vera Filippova, Abhipso Ghosh, Ben Limonchik, Bhargava Urala, Chaitanya Krishna Lanka, Derik Clive, Yi Sun, Edward Li, Hao Wu, Kevin Hongtongsak, Ianna Li, Kalind Thakkar, Kuanysh Omarov, Kushal Majmundar, Michael Alverson, Michael Kucharski, Mohak Patel, Mudit Jain, Maksim Zabelin, Paolo Pelagatti, Rohan Kohli, Saurabh Kumar, Joseph Kim, Swetha Sankar, Vineet Shah, Lakshmi Ramachandruni, Xiangkai Zeng, Ben Bariach, Laura Weidinger, Tu Vu, Alek Andreev, Antoine He, Kevin Hui, Sheleem Kashem, Amar Subramanya, Sissie Hsiao, Demis Hassabis, Koray Kavukcuoglu, Adam Sadovsky, Quoc Le, Trevor Strohman, Yonghui Wu, Slav Petrov, Jeffrey Dean, and Oriol Vinyals. Gemini: A Family of Highly Capable Multimodal Models, June 2024a. URL http://arxiv.org/abs/2312.11805. arXiv:2312.11805.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian

Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open Foundation and Fine-Tuned Chat Models, July 2023. URL http://arxiv.org/abs/2307.09288. arXiv:2307.09288.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway,

Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju-yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving Open Language Models at a Practical Size, Octo-

ber 2024b. URL http://arxiv.org/abs/2408.00118. arXiv:2408.00118.

Chameleon Team. Chameleon: Mixed-Modal Early-Fusion Foundation Models, May 2024. URL http://arxiv.org/abs/2405.09818. arXiv:2405.09818.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo

Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein,

Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy

Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook

Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vítor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. The Llama 3 Herd of Models, August 2024. URL http://arxiv.org/abs/2407.21783. arXiv:2407.21783.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan

Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario

Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, C. J. Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. GPT-4 Technical Report, March 2024. URL http://arxiv.org/abs/2303.08774. arXiv:2303.08774.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of Artificial General Intelligence: Early experiments with GPT-4, April 2023. URL http://arxiv.org/abs/2303.12712. arXiv:2303.12712.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10674–10685, New Orleans, LA, USA, June 2022. IEEE. ISBN 978-1-

66546-946-3. doi: 10.1109/CVPR52688.2022.01042. URL https://ieeexplore.ieee.org/document/9878449/.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, August 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03819-2. URL https://www.nature.com/articles/s41586-021-03819-2. Publisher: Nature Publishing Group.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently Modeling Long Sequences with Structured State Spaces, August 2022a. URL http://arxiv.org/abs/2111.00396. arXiv:2111.00396.

Junxiong Wang, Jing Nathan Yan, Albert Gu, and Alexander M. Rush. Pretraining Without Attention, May 2023. URL http://arxiv.org/abs/2212.10544. arXiv:2212.10544.

Jimmy T. H. Smith, Andrew Warrington, and Scott Linderman. Simplified State Space Layers for Sequence Modeling. In *The Eleventh International Conference*

on Learning Representations, 2023. URL https://openreview.net/forum?id=Ai8Hw3AXqks.

Daniel Y Fu, Tri Dao, Khaled Kamal Saab, Armin W Thomas, Atri Rudra, and Christopher Re. Hungry Hungry Hippos: Towards Language Modeling with State Space Models. In *The Eleventh International Conference on Learning Representations*, 2022.

Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Re. Hyena Hierarchy: Towards Larger Convolutional Language Models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 28043–28078. PMLR, July 2023. URL https://proceedings.mlr.press/v202/poli23a.html. ISSN: 2640-3498.

Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces, May 2024. URL http://arxiv.org/abs/2312.00752. arXiv:2312.00752.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality, May 2024. URL http://arxiv.org/abs/2405.21060. arXiv:2405.21060.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, Xuzheng He, Haowen Hou, Jiaju Lin, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Bolun Wang, Johan S. Wind, Stanislaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, Peng Zhou, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. RWKV:

Reinventing RNNs for the Transformer Era, December 2023. URL http://arxiv.org/abs/2305.13048. arXiv:2305.13048.

Antonio Orvieto, Samuel L. Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting Recurrent Neural Networks for Long Sequences. In *Proceedings of the 40th International Conference on Machine Learning*, pages 26670–26698. PMLR, July 2023. URL https://proceedings.mlr.press/v202/orvieto23a.html. ISSN: 2640-3498.

Aleksandar Botev, Soham De, Samuel L. Smith, Anushan Fernando, George-Cristian Muraru, Ruba Haroun, Leonard Berrada, Razvan Pascanu, Pier Giuseppe Sessa, Robert Dadashi, Léonard Hussenot, Johan Ferret, Sertan Girgin, Olivier Bachem, Alek Andreev, Kathleen Kenealy, Thomas Mesnard, Cassidy Hardin, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Armand Joulin, Noah Fiedel, Evan Senter, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, David Budden, Arnaud Doucet, Sharad Vikram, Adam Paszke, Trevor Gale, Sebastian Borgeaud, Charlie Chen, Andy Brock, Antonia Paterson, Jenny Brennan, Meg Risdal, Raj Gundluru, Nesh Devanathan, Paul Mooney, Nilay Chauhan, Phil Culliton, Luiz Gustavo Martins, Elisa Bandy, David Huntsperger, Glenn Cameron, Arthur Zucker, Tris Warkentin, Ludovic Peran, Minh Giang, Zoubin Ghahramani, Clément Farabet, Koray Kavukcuoglu, Demis Hassabis, Raia Hadsell, Yee Whye Teh, and Nando de Frietas. RecurrentGemma: Moving Past Transformers for Efficient Open Language Models, August 2024. URL http://arxiv.org/abs/2404.07839. arXiv:2404.07839.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksan-

dra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xLSTM: Extended Long Short-Term Memory, May 2024. URL http://arxiv.org/abs/2405.04517. arXiv:2405.04517.

Leo Feng, Frederick Tung, Mohamed Osama Ahmed, Yoshua Bengio, and Hossein Hajimirsadegh. Were RNNs All We Needed?, October 2024. URL http://arxiv.org/abs/2410.01201. arXiv:2410.01201.

Bo Peng, Daniel Goldstein, Quentin Anthony, Alon Albalak, Eric Alcaide, Stella Biderman, Eugene Cheah, Xingjian Du, Teddy Ferdinan, Haowen Hou, Przemysław Kazienko, Kranthi Kiran GV, Jan Kocoń, Bartłomiej Koptyra, Satyapriya Krishna, Ronald McClelland Jr, Jiaju Lin, Niklas Muennighoff, Fares Obeid, Atsushi Saito, Guangyu Song, Haoqin Tu, Cahya Wirawan, Stanisław Woźniak, Ruichong Zhang, Bingchen Zhao, Qihang Zhao, Peng Zhou, Jian Zhu, and Rui-Jie Zhu. Eagle and Finch: RWKV with Matrix-Valued States and Dynamic Recurrence, September 2024. URL http://arxiv.org/abs/2404.05892. arXiv:2404.05892.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention, August 2020. URL http://arxiv.org/abs/2006.16236. arXiv:2006.16236.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. Gated Linear Attention Transformers with Hardware-Efficient Training, August 2024. URL http://arxiv.org/abs/2312.06635. arXiv:2312.06635.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. HGRN2: Gated Linear RNNs with State Expansion, August 2024. URL http://arxiv.org/abs/2404.07904. arXiv:2404.07904.

Tobias Katsch. GateLoop: Fully Data-Controlled Linear Recurrence for Sequence Modeling, January 2024. URL http://arxiv.org/abs/2311.01927. arXiv:2311.01927.

Nicola Muca Cirone, Antonio Orvieto, Benjamin Walker, Cristopher Salvi, and Terry Lyons. Theoretical Foundations of Deep Selective State-Space Models, November 2024. URL http://arxiv.org/abs/2402.19047. arXiv:2402.19047.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://ieeexplore.ieee.org/abstract/document/6795963. Conference Name: Neural Computation.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, September 2014. URL http://arxiv.org/abs/1406.1078. arXiv:1406.1078.

Eric Martin and Chris Cundy. Parallelizing Linear Recurrent Neural Nets Over Sequence Length, February 2018. URL http://arxiv.org/abs/1709.04057. arXiv:1709.04057.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and Eran Malach. Repeat After Me: Transformers are Better than State Space Models at Copying, June 2024. URL http://arxiv.org/abs/2402.01032. arXiv:2402.01032.

William Merrill, Jackson Petty, and Ashish Sabharwal. The Illusion of State in

State-Space Models, June 2024. URL http://arxiv.org/abs/2404.08819. arXiv:2404.08819.

Junxiong Wang, Tushaar Gangavarapu, Jing Nathan Yan, and Alexander M. Rush. MambaByte: Token-free Selective State Space Model, August 2024a. URL http://arxiv.org/abs/2401.13660. arXiv:2401.13660.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A Neural Probabilistic Language Model. *Advances in neural information processing systems*, 13, 2000.

Mike Schuster and Kaisuke Nakajima. Japanese and Korean Voice Search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. *arXiv preprint arXiv:1508.07909*, 2015.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, and others. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv preprint arXiv:1609.08144*, 2016.

Changhan Wang, Kyunghyun Cho, and Jiatao Gu. Neural Machine Translation with Byte-Level Subwords. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 9154–9160, 2020.

Yingqiang Gao, Nikola I Nikolov, Yuhuang Hu, and Richard HR Hahnloser. Character-Level Translation with Self-attention. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1591–1604, 2020a.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306, 2022. Publisher: MIT Press One Broadway, 12th Floor, Cambridge, Massachusetts 02142, USA ....

Dokook Choe, Rami Al-Rfou, Mandy Guo, Heeyoung Lee, and Noah Constant. Bridging the gap for tokenizer-free language models. *arXiv preprint arXiv:1908.10322*, 2019.

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-Level Language Modeling with Deeper Self-Attention. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3159–3166, July 2019. doi: 10.1609/aaai.v33i01.33013159. URL https://ojs.aaai.org/index.php/AAAI/article/view/4182.

Jonathan H Clark, Dan Garrette, Iulia Turc, and John Wieting. Canine: Pre-training an Efficient Tokenization-Free Encoder for Language Representation. *Transactions of the Association for Computational Linguistics*, 10:73–91, 2022. Publisher: MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info ....

Yi Tay, Vinh Q. Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. Charformer: Fast Character Transformers via Gradient-based Subword Tokenization, February 2022. URL http://arxiv.org/abs/2106.12672. arXiv:2106.12672.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, and

others. OPT: Open Pre-trained Transformer Language Models. *arXiv preprint arXiv:2205.01068*, 2022.

Zihang Dai, Guokun Lai, Yiming Yang, and Quoc Le. Funnel-Transformer: Filtering out Sequential Redundancy for Efficient Language Processing. *Advances in neural information processing systems*, 33:4271–4282, 2020.

Piotr Nawrot, Jan Chorowski, Adrian Lancucki, and Edoardo Maria Ponti. Efficient Transformers with Dynamic Token Pooling. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6403–6417, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.353. URL https://aclanthology.org/2023.acl-long.353.

Ankit Gupta, Albert Gu, and Jonathan Berant. Diagonal State Spaces are as Effective as Structured State Spaces. *Advances in Neural Information Processing Systems*, 35:22982–22994, 2022.

Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the Parameterization and Initialization of Diagonal State Space Models. *Advances in Neural Information Processing Systems*, 35:35971–35983, 2022b.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast Inference from Transformers via Speculative Decoding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19274–19286. PMLR, July 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Lau-

rent Sifre, and John Jumper. Accelerating Large Language Model Decoding with Speculative Sampling, 2023. _eprint: 2302.01318.

Heming Xia, Tao Ge, Peiyi Wang, Si-Qing Chen, Furu Wei, and Zhifang Sui. Speculative Decoding: Exploiting Speculative Execution for Accelerating Seq2seq Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3909–3925, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.257. URL https://aclanthology.org/2023.findings-emnlp.257.

Guy E Blelloch. Prefix Sums and Their Applications. (CMU-CS-90-190), November 1990. URL https://www.cs.cmu.edu/~guyb/papers/Ble93.pdf. Publisher: School of Computer Science, Carnegie Mellon University.

Szymon Tworkowski, Konrad Staniszewski, Mikołaj Pacek, Yuhuai Wu, Henryk Michalewski, and Piotr Miłoś. Focused transformer: Contrastive training for context scaling. *Advances in Neural Information Processing Systems*, 36, 2024.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive Transformers for Long-Range Sequence Modelling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SylKikSYDH.

Trieu H. Trinh and Quoc V. Le. A Simple Method for Commonsense Reasoning. *arXiv preprint arXiv:1806.02847*, 2018.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *arXiv preprint arXiv:2101.00027*, 2020b.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient Content-Based Sparse Attention with Routing Transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68, 2021. doi: 10.1162/tacl_a_00353. URL https://aclanthology.org/2021.tacl-1.4. Place: Cambridge, MA Publisher: MIT Press.

Curtis Hawthorne, Andrew Jaegle, Cătălina Cangea, Sebastian Borgeaud, Charlie Nash, Mateusz Malinowski, Sander Dieleman, Oriol Vinyals, Matthew Botvinick, Ian Simon, Hannah Sheahan, Neil Zeghidour, Jean-Baptiste Alayrac, Joao Carreira, and Jesse Engel. General-purpose, long-context autoregressive modeling with Perceiver AR. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8535–8558. PMLR, July 2022. URL https://proceedings.mlr.press/v162/hawthorne22a.html.

DeLesley Hutchins, Imanol Schlag, Yuhuai Wu, Ethan Dyer, and Behnam Neyshabur. Block-Recurrent Transformers. *Advances in Neural Information Processing Systems*, 35:33248–33261, 2022.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://aclanthology.org/D18-2012.

Wentao Ma, Yiming Cui, Chenglei Si, Ting Liu, Shijin Wang, and Guoping Hu. CharBERT: character-aware pre-trained language model. *arXiv preprint arXiv:2011.01513*, 2020.

Sebastian J Mielke and Jason Eisner. Spell once, summon anywhere: A two-level open-vocabulary language model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6843–6850, 2019. Issue: 01.

Jonas Belouadi and Steffen Eger. Bygpt5: End-to-end style-conditioned poetry generation with token-free language models. *arXiv preprint arXiv:2212.10474*, 2022.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Ré. Zoology: Measuring and improving recall in efficient language models. *arXiv preprint arXiv:2312.04927*, 2023.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, Dylan Zinsley, James Zou, Atri Rudra, and Christopher Ré. Simple linear attention language models balance the recall-throughput tradeoff. *arXiv preprint arXiv:2402.18668*, 2024.

Dan Fu, Simran Arora, Jessica Grogan, Isys Johnson, Evan Sabri Eyuboglu, Armin Thomas, Benjamin Spector, Michael Poli, Atri Rudra, and Christopher Ré. Monarch mixer: A simple sub-quadratic gemm-based architecture. *Advances in Neural Information Processing Systems*, 36, 2024a.

Jing Nathan Yan, Jiatao Gu, and Alexander M Rush. Diffusion models without attention. *CVPR 2024*, 2024.

Karan Goel, Albert Gu, Chris Donahue, and Christopher Ré. It's raw! audio generation with state-space models. In *International Conference on Machine Learning*, pages 7616–7633. PMLR, 2022.

Yair Schiff, Chia-Hsiang Kao, Aaron Gokaslan, Tri Dao, Albert Gu, and Volodymyr Kuleshov. Caduceus: Bi-directional equivariant long-range dna sequence modeling. *arXiv preprint arXiv:2403.03234*, 2024.

Benjamin Spector and Chris Re. Accelerating llm inference with staged speculative decoding. *arXiv preprint arXiv:2308.04623*, 2023.

Nikhil Bhendawade, Irina Belousova, Qichen Fu, Henry Mason, Mohammad Rastegari, and Mahyar Najibi. Speculative Streaming: Fast LLM Inference without Auxiliary Models. *arXiv preprint arXiv:2402.11131*, 2024.

Xiaoxuan Liu, Lanxiang Hu, Peter Bailis, Ion Stoica, Zhijie Deng, Alvin Cheung, and Hao Zhang. Online speculative decoding. *arXiv preprint arXiv:2310.07177*, 2023.

Zhenyu He, Zexuan Zhong, Tianle Cai, Jason D Lee, and Di He. Rest: Retrieval-based speculative decoding. *arXiv preprint arXiv:2311.08252*, 2023.

Nan Yang, Tao Ge, Liang Wang, Binxing Jiao, Daxin Jiang, Linjun Yang, Rangan Majumder, and Furu Wei. Inference with reference: Lossless acceleration of large language models. *arXiv preprint arXiv:2304.04487*, 2023.

Yichao Fu, Peter Bailis, Ion Stoica, and Hao Zhang. Break the sequential dependency of llm inference using lookahead decoding. *arXiv preprint arXiv:2402.02057*, 2024b.

Ramakrishna Bairi, Atharv Sonwane, Aditya Kanade, Vageesh D. C., Arun Iyer, Suresh Parthasarathy, Sriram Rajamani, B. Ashok, and Shashank Shet. CodePlan: Repository-Level Coding using LLMs and Planning. *Proc. ACM Softw. Eng.*, 1(FSE):31:675–31:698, July 2024. doi: 10.1145/3643757. URL https://dl.acm.org/doi/10.1145/3643757.

Lilian Weng. LLM Powered Autonomous Agents, June 2023. URL https://lilianweng.github.io/posts/2023-06-23-agent/. Section: posts.

Sahisnu Mazumder and Bing Liu. *Lifelong and Continual Learning Dialogue Systems*. Synthesis Lectures on Human Language Technologies. Springer International Publishing, Cham, 2024. ISBN 978-3-031-48188-8 978-3-031-48189-5. doi: 10.1007/978-3-031-48189-5. URL https://link.springer.com/10.1007/978-3-031-48189-5.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting Language Models to Compress Contexts, November 2023. URL http://arxiv.org/abs/2305.14788. arXiv:2305.14788.

Rishabh Agarwal, Avi Singh, Lei M. Zhang, Bernd Bohnet, Luis Rosias, Stephanie Chan, Biao Zhang, Ankesh Anand, Zaheer Abbas, Azade Nova, John D. Co-Reyes, Eric Chu, Feryal Behbahani, Aleksandra Faust, and Hugo Larochelle. Many-Shot In-Context Learning, October 2024. URL http://arxiv.org/abs/2404.11018. arXiv:2404.11018.

Jonathan Lee, Annie Xie, Aldo Pacchiano, Yash Chandak, Chelsea Finn, Ofir Nachum, and Emma Brunskill. Supervised Pretraining Can Learn In-Context Reinforcement Learning. *Advances in Neural Information Processing Systems*, 36: 43057–43083, December 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8644b61a9bc87bf7844750a015feb600-Abstract-Conference.html.

Amirkeivan Mohtashami and Martin Jaggi. Landmark Attention: Random-Access Infinite Context Length for Transformers, November 2023. URL http://arxiv.org/abs/2305.16300. arXiv:2305.16300.

David Samuel. BERTs are Generative In-Context Learners, October 2024. URL http://arxiv.org/abs/2406.04823. arXiv:2406.04823.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. In Search of Needles in a 11M Haystack: Recurrent Memory Finds What LLMs Miss, February 2024. URL http://arxiv.org/abs/2402.10790. arXiv:2402.10790.

Alireza Makhzani and Brendan Frey. k-Sparse Autoencoders, March 2014. URL http://arxiv.org/abs/1312.5663. arXiv:1312.5663.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders, June 2024. URL http://arxiv.org/abs/2406.04093. arXiv:2406.04093.

Trenton Bricken, Jonathan Marcus, Siddharth Mishra-Sharma, Meg Tong, Ethan Perez, Mrinank Sharma, Kelley Rivoire, Thomas Henighan, and Adam Jermyn. Using Dictionary Learning Features as Classifiers, 2023. URL https://transformer-circuits.pub/2024/features-as-classifiers/index.html.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse Autoencoders Find Highly Interpretable Features in Language Models, October 2023. URL http://arxiv.org/abs/2309.08600. arXiv:2309.08600.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse Feature Circuits: Discovering and Editing Interpretable

Causal Graphs in Language Models, March 2024. URL http://arxiv.org/abs/2403.19647. arXiv:2403.19647.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma Scope: Open Sparse Autoencoders Everywhere All At Once on Gemma 2, August 2024. URL http://arxiv.org/abs/2408.05147. arXiv:2408.05147.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space, September 2013. URL http://arxiv.org/abs/1301.3781. arXiv:1301.3781.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom In: An Introduction to Circuits. *Distill*, 5(3):e00024.001, March 2020. ISSN 2476-0757. doi: 10.23915/distill.00024.001. URL https://distill.pub/2020/circuits/zoom-in.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy Models of Superposition, September 2022. URL http://arxiv.org/abs/2209.10652. arXiv:2209.10652.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding Neurons in a Haystack: Case Studies with Sparse Probing, June 2023. URL http://arxiv.org/abs/2305.01610. arXiv:2305.01610.

Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent Linear Representa-

tions in World Models of Self-Supervised Sequence Models, September 2023. URL http://arxiv.org/abs/2309.00941. arXiv:2309.00941.

Kiho Park, Yo Joong Choe, and Victor Veitch. The Linear Representation Hypothesis and the Geometry of Large Language Models, July 2024. URL http://arxiv.org/abs/2311.03658. arXiv:2311.03658.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet, May 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/.

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?, August 2024. URL http://arxiv.org/abs/2404.06654. arXiv:2404.06654.

Jean Kaddour. The MiniPile Challenge for Data-Efficient Language Models, April 2023. URL http://arxiv.org/abs/2304.08442. arXiv:2304.08442.

Joshua Engels, Isaac Liao, Eric J. Michaud, Wes Gurnee, and Max Tegmark. Not All Language Model Features Are Linear, May 2024. URL http://arxiv.org/abs/2405.14860. arXiv:2405.14860 version: 1.

EleutherAI. EleutherAI/sae, November 2024. URL https://github.com/EleutherAI/sae. original-date: 2024-05-24T06:22:26Z.

Mark Raasveldt and Hannes Mühleisen. DuckDB: an Embeddable Analytical Database. In *Proceedings of the 2019 International Conference on Management of Data*, SIGMOD '19, pages 1981–1984, New York, NY, USA, June 2019. Association for Computing Machinery. ISBN 978-1-4503-5643-5. doi: 10.1145/3299869.3320212. URL https://dl.acm.org/doi/10.1145/3299869.3320212.

Anthropic. Introducing the next generation of Claude, March 2024. URL https://www.anthropic.com/news/claude-3-family.

Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, Soroosh Mariooryad, Yifan Ding, Xinyang Geng, Fred Alcober, Roy Frostig, Mark Omernick, Lexi Walker, Cosmin Paduraru, Christina Sorokin, Andrea Tacchetti, Colin Gaffney, Samira Daruki, Olcan Sercinoglu, Zach Gleicher, Juliette Love, Paul Voigtlaender, Rohan Jain, Gabriela Surita, Kareem Mohamed, Rory Blevins, Junwhan Ahn, Tao Zhu, Kornraphop Kawintiranon, Orhan Firat, Yiming Gu, Yujing Zhang, Matthew Rahtz, Manaal Faruqui, Natalie Clay, Justin Gilmer, J. D. Co-Reyes, Ivo Penchev, Rui Zhu, Nobuyuki Morioka, Kevin Hui, Krishna Haridasan, Victor Campos, Mahdis Mahdieh, Mandy Guo, Samer Hassan, Kevin Kilgour, Arpi Vezer, Heng-Tze Cheng, Raoul de Liedekerke, Siddharth Goyal, Paul Barham, D. J. Strouse, Seb Noury, Jonas Adler, Mukund Sundararajan, Sharad Vikram, Dmitry Lepikhin, Michela Paganini, Xavier Garcia, Fan Yang, Dasha Valter, Maja Trebacz, Kiran Vodrahalli, Chulayuth Asawaroengchai, Roman Ring, Norbert Kalb, Livio Baldini Soares, Siddhartha Brahma, David

Steiner, Tianhe Yu, Fabian Mentzer, Antoine He, Lucas Gonzalez, Bibo Xu, Raphael Lopez Kaufman, Laurent El Shafey, Junhyuk Oh, Tom Hennigan, George van den Driessche, Seth Odoom, Mario Lucic, Becca Roelofs, Sid Lall, Amit Marathe, Betty Chan, Santiago Ontanon, Luheng He, Denis Teplyashin, Jonathan Lai, Phil Crone, Bogdan Damoc, Lewis Ho, Sebastian Riedel, Karel Lenc, Chih-Kuan Yeh, Aakanksha Chowdhery, Yang Xu, Mehran Kazemi, Ehsan Amid, Anastasia Petrushkina, Kevin Swersky, Ali Khodaei, Gowoon Chen, Chris Larkin, Mario Pinto, Geng Yan, Adria Puigdomenech Badia, Piyush Patil, Steven Hansen, Dave Orr, Sebastien M. R. Arnold, Jordan Grimstad, Andrew Dai, Sholto Douglas, Rishika Sinha, Vikas Yadav, Xi Chen, Elena Gribovskaya, Jacob Austin, Jeffrey Zhao, Kaushal Patel, Paul Komarek, Sophia Austin, Sebastian Borgeaud, Linda Friso, Abhimanyu Goyal, Ben Caine, Kris Cao, Da-Woon Chung, Matthew Lamm, Gabe Barth-Maron, Thais Kagohara, Kate Olszewska, Mia Chen, Kaushik Shivakumar, Rishabh Agarwal, Harshal Godhia, Ravi Rajwar, Javier Snaider, Xerxes Dotiwalla, Yuan Liu, Aditya Barua, Victor Ungureanu, Yuan Zhang, Bat-Orgil Batsaikhan, Mateo Wirth, James Qin, Ivo Danihelka, Tulsee Doshi, Martin Chadwick, Jilin Chen, Sanil Jain, Quoc Le, Arjun Kar, Madhu Gurumurthy, Cheng Li, Ruoxin Sang, Fangyu Liu, Lampros Lamprou, Rich Munoz, Nathan Lintz, Harsh Mehta, Heidi Howard, Malcolm Reynolds, Lora Aroyo, Quan Wang, Lorenzo Blanco, Albin Cassirer, Jordan Griffith, Dipanjan Das, Stephan Lee, Jakub Sygnowski, Zach Fisher, James Besley, Richard Powell, Zafarali Ahmed, Dominik Paulus, David Reitter, Zalan Borsos, Rishabh Joshi, Aedan Pope, Steven Hand, Vittorio Selo, Vihan Jain, Nikhil Sethi, Megha Goel, Takaki Makino, Rhys May, Zhen Yang, Johan Schalkwyk, Christina Butterfield, Anja Hauth, Alex Goldin, Will Hawkins, Evan Senter, Sergey Brin, Oliver Woodman, Marvin Ritter, Eric Noland, Minh Giang,

Vijay Bolina, Lisa Lee, Tim Blyth, Ian Mackinnon, Machel Reid, Obaid Sarvana, David Silver, Alexander Chen, Lily Wang, Loren Maggiore, Oscar Chang, Nithya Attaluri, Gregory Thornton, Chung-Cheng Chiu, Oskar Bunyan, Nir Levine, Timothy Chung, Evgenii Eltyshev, Xiance Si, Timothy Lillicrap, Demetra Brady, Vaibhav Aggarwal, Boxi Wu, Yuanzhong Xu, Ross McIlroy, Kartikeya Badola, Paramjit Sandhu, Erica Moreira, Wojciech Stokowiec, Ross Hemsley, Dong Li, Alex Tudor, Pranav Shyam, Elahe Rahimtoroghi, Salem Haykal, Pablo Sprechmann, Xiang Zhou, Diana Mincu, Yujia Li, Ravi Addanki, Kalpesh Krishna, Xiao Wu, Alexandre Frechette, Matan Eyal, Allan Dafoe, Dave Lacey, Jay Whang, Thi Avrahami, Ye Zhang, Emanuel Taropa, Hanzhao Lin, Daniel Toyama, Eliza Rutherford, Motoki Sano, HyunJeong Choe, Alex Tomala, Chalence Safranek-Shrader, Nora Kassner, Mantas Pajarskas, Matt Harvey, Sean Sechrist, Meire Fortunato, Christina Lyu, Gamaleldin Elsayed, Chenkai Kuang, James Lottes, Eric Chu, Chao Jia, Chih-Wei Chen, Peter Humphreys, Kate Baumli, Connie Tao, Rajkumar Samuel, Cicero Nogueira dos Santos, Anders Andreassen, Nemanja Rakićević, Dominik Grewe, Aviral Kumar, Stephanie Winkler, Jonathan Caton, Andrew Brock, Sid Dalmia, Hannah Sheahan, Iain Barr, Yingjie Miao, Paul Natsev, Jacob Devlin, Feryal Behbahani, Flavien Prost, Yanhua Sun, Artiom Myaskovsky, Thanumalayan Sankaranarayana Pillai, Dan Hurt, Angeliki Lazaridou, Xi Xiong, Ce Zheng, Fabio Pardo, Xiaowei Li, Dan Horgan, Joe Stanton, Moran Ambar, Fei Xia, Alejandro Lince, Mingqiu Wang, Basil Mustafa, Albert Webson, Hyo Lee, Rohan Anil, Martin Wicke, Timothy Dozat, Abhishek Sinha, Enrique Piqueras, Elahe Dabir, Shyam Upadhyay, Anudhyan Boral, Lisa Anne Hendricks, Corey Fry, Josip Djolonga, Yi Su, Jake Walker, Jane Labanowski, Ronny Huang, Vedant Misra, Jeremy Chen, R. J. Skerry-Ryan, Avi Singh, Shruti Rijhwani, Dian Yu, Alex Castro-Ros, Beer

Changpinyo, Romina Datta, Sumit Bagri, Arnar Mar Hrafnkelsson, Marcello Maggioni, Daniel Zheng, Yury Sulsky, Shaobo Hou, Tom Le Paine, Antoine Yang, Jason Riesa, Dominika Rogozinska, Dror Marcus, Dalia El Badawy, Qiao Zhang, Luyu Wang, Helen Miller, Jeremy Greer, Lars Lowe Sjos, Azade Nova, Heiga Zen, Rahma Chaabouni, Mihaela Rosca, Jiepu Jiang, Charlie Chen, Ruibo Liu, Tara Sainath, Maxim Krikun, Alex Polozov, Jean-Baptiste Lespiau, Josh Newlan, Zeyncep Cankara, Soo Kwak, Yunhan Xu, Phil Chen, Andy Coenen, Clemens Meyer, Katerina Tsihlas, Ada Ma, Juraj Gottweis, Jinwei Xing, Chenjie Gu, Jin Miao, Christian Frank, Zeynep Cankara, Sanjay Ganapathy, Ishita Dasgupta, Steph Hughes-Fitt, Heng Chen, David Reid, Keran Rong, Hongmin Fan, Joost van Amersfoort, Vincent Zhuang, Aaron Cohen, Shixiang Shane Gu, Anhad Mohananey, Anastasija Ilic, Taylor Tobin, John Wieting, Anna Bortsova, Phoebe Thacker, Emma Wang, Emily Caveness, Justin Chiu, Eren Sezener, Alex Kaskasoli, Steven Baker, Katie Millican, Mohamed Elhawaty, Kostas Aisopos, Carl Lebsack, Nathan Byrd, Hanjun Dai, Wenhao Jia, Matthew Wiethoff, Elnaz Davoodi, Albert Weston, Lakshman Yagati, Arun Ahuja, Isabel Gao, Golan Pundak, Susan Zhang, Michael Azzam, Khe Chai Sim, Sergi Caelles, James Keeling, Abhanshu Sharma, Andy Swing, YaGuang Li, Chenxi Liu, Carrie Grimes Bostock, Yamini Bansal, Zachary Nado, Ankesh Anand, Josh Lipschultz, Abhijit Karmarkar, Lev Proleev, Abe Ittycheriah, Soheil Hassas Yeganeh, George Polovets, Aleksandra Faust, Jiao Sun, Alban Rrustemi, Pen Li, Rakesh Shivanna, Jeremiah Liu, Chris Welty, Federico Lebron, Anirudh Baddepudi, Sebastian Krause, Emilio Parisotto, Radu Soricut, Zheng Xu, Dawn Bloxwich, Melvin Johnson, Behnam Neyshabur, Justin Mao-Jones, Renshen Wang, Vinay Ramasesh, Zaheer Abbas, Arthur Guez, Constant Segal, Duc Dung Nguyen, James Svensson, Le Hou, Sarah York, Kieran Milan, So-

phie Bridgers, Wiktor Gworek, Marco Tagliasacchi, James Lee-Thorp, Michael Chang, Alexey Guseynov, Ale Jakse Hartman, Michael Kwong, Ruizhe Zhao, Sheleem Kashem, Elizabeth Cole, Antoine Miech, Richard Tanburn, Mary Phuong, Filip Pavetic, Sebastien Cevey, Ramona Comanescu, Richard Ives, Sherry Yang, Cosmo Du, Bo Li, Zizhao Zhang, Mariko Iinuma, Clara Huiyi Hu, Aurko Roy, Shaan Bijwadia, Zhenkai Zhu, Danilo Martins, Rachel Saputro, Anita Gergely, Steven Zheng, Dawei Jia, Ioannis Antonoglou, Adam Sadovsky, Shane Gu, Yingying Bi, Alek Andreev, Sina Samangooei, Mina Khan, Tomas Kocisky, Angelos Filos, Chintu Kumar, Colton Bishop, Adams Yu, Sarah Hodkinson, Sid Mittal, Premal Shah, Alexandre Moufarek, Yong Cheng, Adam Bloniarz, Jaehoon Lee, Pedram Pejman, Paul Michel, Stephen Spencer, Vladimir Feinberg, Xuehan Xiong, Nikolay Savinov, Charlotte Smith, Siamak Shakeri, Dustin Tran, Mary Chesus, Bernd Bohnet, George Tucker, Tamara von Glehn, Carrie Muir, Yiran Mao, Hideto Kazawa, Ambrose Slone, Kedar Soparkar, Disha Shrivastava, James Cobon-Kerr, Michael Sharman, Jay Pavagadhi, Carlos Araya, Karolis Misiunas, Nimesh Ghelani, Michael Laskin, David Barker, Qiujia Li, Anton Briukhov, Neil Houlsby, Mia Glaese, Balaji Lakshminarayanan, Nathan Schucher, Yunhao Tang, Eli Collins, Hyeontaek Lim, Fangxiaoyu Feng, Adria Recasens, Guangda Lai, Alberto Magni, Nicola De Cao, Aditya Siddhant, Zoe Ashwood, Jordi Orbay, Mostafa Dehghani, Jenny Brennan, Yifan He, Kelvin Xu, Yang Gao, Carl Saroufim, James Molloy, Xinyi Wu, Seb Arnold, Solomon Chang, Julian Schrittwieser, Elena Buchatskaya, Soroush Radpour, Martin Polacek, Skye Giordano, Ankur Bapna, Simon Tokumine, Vincent Hellendoorn, Thibault Sottiaux, Sarah Cogan, Aliaksei Severyn, Mohammad Saleh, Shantanu Thakoor, Laurent Shefey, Siyuan Qiao, Meenu Gaba, Shuo-yiin Chang, Craig Swanson, Biao Zhang, Benjamin Lee, Paul Kis-

han Rubenstein, Gan Song, Tom Kwiatkowski, Anna Koop, Ajay Kannan, David Kao, Parker Schuh, Axel Stjerngren, Golnaz Ghiasi, Gena Gibson, Luke Vilnis, Ye Yuan, Felipe Tiengo Ferreira, Aishwarya Kamath, Ted Klimenko, Ken Franko, Kefan Xiao, Indro Bhattacharya, Miteyan Patel, Rui Wang, Alex Morris, Robin Strudel, Vivek Sharma, Peter Choy, Sayed Hadi Hashemi, Jessica Landon, Mara Finkelstein, Priya Jhakra, Justin Frye, Megan Barnes, Matthew Mauger, Dennis Daun, Khuslen Baatarsukh, Matthew Tung, Wael Farhan, Henryk Michalewski, Fabio Viola, Felix de Chaumont Quitry, Charline Le Lan, Tom Hudson, Qingze Wang, Felix Fischer, Ivy Zheng, Elspeth White, Anca Dragan, Jean-baptiste Alayrac, Eric Ni, Alexander Pritzel, Adam Iwanicki, Michael Isard, Anna Bulanova, Lukas Zilka, Ethan Dyer, Devendra Sachan, Srivatsan Srinivasan, Hannah Muckenhirn, Honglong Cai, Amol Mandhane, Mukarram Tariq, Jack W. Rae, Gary Wang, Kareem Ayoub, Nicholas FitzGerald, Yao Zhao, Woohyun Han, Chris Alberti, Dan Garrette, Kashyap Krishnakumar, Mai Gimenez, Anselm Levskaya, Daniel Sohn, Josip Matak, Inaki Iturrate, Michael B. Chang, Jackie Xiang, Yuan Cao, Nishant Ranka, Geoff Brown, Adrian Hutter, Vahab Mirrokni, Nanxin Chen, Kaisheng Yao, Zoltan Egyed, Francois Galilee, Tyler Liechty, Praveen Kallakuri, Evan Palmer, Sanjay Ghemawat, Jasmine Liu, David Tao, Chloe Thornton, Tim Green, Mimi Jasarevic, Sharon Lin, Victor Cotruta, Yi-Xuan Tan, Noah Fiedel, Hongkun Yu, Ed Chi, Alexander Neitz, Jens Heitkaemper, Anu Sinha, Denny Zhou, Yi Sun, Charbel Kaed, Brice Hulse, Swaroop Mishra, Maria Georgaki, Sneha Kudugunta, Clement Farabet, Izhak Shafran, Daniel Vlasic, Anton Tsitsulin, Rajagopal Ananthanarayanan, Alen Carin, Guolong Su, Pei Sun, Shashank V, Gabriel Carvajal, Josef Broder, Iulia Comsa, Alena Repina, William Wong, Warren Weilun Chen, Peter Hawkins, Egor Filonov, Lucia Loher, Christoph

Hirnschall, Weiyi Wang, Jingchen Ye, Andrea Burns, Hardie Cate, Diana Gage Wright, Federico Piccinini, Lei Zhang, Chu-Cheng Lin, Ionel Gog, Yana Kulizhskaya, Ashwin Sreevatsa, Shuang Song, Luis C. Cobo, Anand Iyer, Chetan Tekur, Guillermo Garrido, Zhuyun Xiao, Rupert Kemp, Huaixiu Steven Zheng, Hui Li, Ananth Agarwal, Christel Ngani, Kati Goshvadi, Rebeca Santamaria-Fernandez, Wojciech Fica, Xinyun Chen, Chris Gorgolewski, Sean Sun, Roopal Garg, Xinyu Ye, S. M. Ali Eslami, Nan Hua, Jon Simon, Pratik Joshi, Yelin Kim, Ian Tenney, Sahitya Potluri, Lam Nguyen Thiet, Quan Yuan, Florian Luisier, Alexandra Chronopoulou, Salvatore Scellato, Praveen Srinivasan, Minmin Chen, Vinod Koverkathu, Valentin Dalibard, Yaming Xu, Brennan Saeta, Keith Anderson, Thibault Sellam, Nick Fernando, Fantine Huot, Junehyuk Jung, Mani Varadarajan, Michael Quinn, Amit Raul, Maigo Le, Ruslan Habalov, Jon Clark, Komal Jalan, Kalesha Bullard, Achintya Singhal, Thang Luong, Boyu Wang, Sujeevan Rajayogam, Julian Eisenschlos, Johnson Jia, Daniel Finchelstein, Alex Yakubovich, Daniel Balle, Michael Fink, Sameer Agarwal, Jing Li, Dj Dvijotham, Shalini Pal, Kai Kang, Jaclyn Konzelmann, Jennifer Beattie, Olivier Dousse, Diane Wu, Remi Crocker, Chen Elkind, Siddhartha Reddy Jonnalagadda, Jong Lee, Dan Holtmann-Rice, Krystal Kallarackal, Rosanne Liu, Denis Vnukov, Neera Vats, Luca Invernizzi, Mohsen Jafari, Huanjie Zhou, Lilly Taylor, Jennifer Prendki, Marcus Wu, Tom Eccles, Tianqi Liu, Kavya Kopparapu, Francoise Beaufays, Christof Angermueller, Andreea Marzoca, Shourya Sarcar, Hilal Dib, Jeff Stanway, Frank Perbet, Nejc Trdin, Rachel Sterneck, Andrey Khorlin, Dinghua Li, Xihui Wu, Sonam Goenka, David Madras, Sasha Goldshtein, Willi Gierke, Tong Zhou, Yaxin Liu, Yannie Liang, Anais White, Yunjie Li, Shreya Singh, Sanaz Bahargam, Mark Epstein, Sujoy Basu, Li Lao, Adnan Ozturel, Carl Crous, Alex Zhai, Han Lu, Zora Tung, Neeraj

Gaur, Alanna Walton, Lucas Dixon, Ming Zhang, Amir Globerson, Grant Uy, Andrew Bolt, Olivia Wiles, Milad Nasr, Ilia Shumailov, Marco Selvi, Francesco Piccinno, Ricardo Aguilar, Sara McCarthy, Misha Khalman, Mrinal Shukla, Vlado Galic, John Carpenter, Kevin Villela, Haibin Zhang, Harry Richardson, James Martens, Matko Bosnjak, Shreyas Rammohan Belle, Jeff Seibert, Mahmoud Alnahlawi, Brian McWilliams, Sankalp Singh, Annie Louis, Wen Ding, Dan Popovici, Lenin Simicich, Laura Knight, Pulkit Mehta, Nishesh Gupta, Chongyang Shi, Saaber Fatehi, Jovana Mitrovic, Alex Grills, Joseph Pagadora, Dessie Petrova, Danielle Eisenbud, Zhishuai Zhang, Damion Yates, Bhavishya Mittal, Nilesh Tripuraneni, Yannis Assael, Thomas Brovelli, Prateek Jain, Mihajlo Velimirovic, Canfer Akbulut, Jiaqi Mu, Wolfgang Macherey, Ravin Kumar, Jun Xu, Haroon Qureshi, Gheorghe Comanici, Jeremy Wiesner, Zhitao Gong, Anton Ruddock, Matthias Bauer, Nick Felt, Anirudh GP, Anurag Arnab, Dustin Zelle, Jonas Rothfuss, Bill Rosgen, Ashish Shenoy, Bryan Seybold, Xinjian Li, Jayaram Mudigonda, Goker Erdogan, Jiawei Xia, Jiri Simsa, Andrea Michi, Yi Yao, Christopher Yew, Steven Kan, Isaac Caswell, Carey Radebaugh, Andre Elisseeff, Pedro Valenzuela, Kay McKinney, Kim Paterson, Albert Cui, Eri Latorre-Chimoto, Solomon Kim, William Zeng, Ken Durden, Priya Ponnapalli, Tiberiu Sosea, Christopher A. Choquette-Choo, James Manyika, Brona Robenek, Harsha Vashisht, Sebastien Pereira, Hoi Lam, Marko Velic, Denese Owusu-Afriyie, Katherine Lee, Tolga Bolukbasi, Alicia Parrish, Shawn Lu, Jane Park, Balaji Venkatraman, Alice Talbert, Lambert Rosique, Yuchung Cheng, Andrei Sozanschi, Adam Paszke, Praveen Kumar, Jessica Austin, Lu Li, Khalid Salama, Wooyeol Kim, Nandita Dukkipati, Anthony Baryshnikov, Christos Kaplanis, XiangHai Sheng, Yuri Chervonyi, Caglar Unlu, Diego de Las Casas, Harry Askham, Kathryn Tunyasuvunakool, Felix Gimeno, Siim Poder, Chester

Kwak, Matt Miecnikowski, Vahab Mirrokni, Alek Dimitriev, Aaron Parisi, Dangyi Liu, Tomy Tsai, Toby Shevlane, Christina Kouridi, Drew Garmon, Adrian Goedeckemeyer, Adam R. Brown, Anitha Vijayakumar, Ali Elqursh, Sadegh Jazayeri, Jin Huang, Sara Mc Carthy, Jay Hoover, Lucy Kim, Sandeep Kumar, Wei Chen, Courtney Biles, Garrett Bingham, Evan Rosen, Lisa Wang, Qijun Tan, David Engel, Francesco Pongetti, Dario de Cesare, Dongseong Hwang, Lily Yu, Jennifer Pullman, Srini Narayanan, Kyle Levin, Siddharth Gopal, Megan Li, Asaf Aharoni, Trieu Trinh, Jessica Lo, Norman Casagrande, Roopali Vij, Loic Matthey, Bramandia Ramadhana, Austin Matthews, C. J. Carey, Matthew Johnson, Kremena Goranova, Rohin Shah, Shereen Ashraf, Kingshuk Dasgupta, Rasmus Larsen, Yicheng Wang, Manish Reddy Vuyyuru, Chong Jiang, Joana Ijazi, Kazuki Osawa, Celine Smith, Ramya Sree Boppana, Taylan Bilal, Yuma Koizumi, Ying Xu, Yasemin Altun, Nir Shabat, Ben Bariach, Alex Korchemniy, Kiam Choo, Olaf Ronneberger, Chimezie Iwuanyanwu, Shubin Zhao, David Soergel, Cho-Jui Hsieh, Irene Cai, Shariq Iqbal, Martin Sundermeyer, Zhe Chen, Elie Bursztein, Chaitanya Malaviya, Fadi Biadsy, Prakash Shroff, Inderjit Dhillon, Tejasi Latkar, Chris Dyer, Hannah Forbes, Massimo Nicosia, Vitaly Nikolaev, Somer Greene, Marin Georgiev, Pidong Wang, Nina Martin, Hanie Sedghi, John Zhang, Praseem Banzal, Doug Fritz, Vikram Rao, Xuezhi Wang, Jiageng Zhang, Viorica Patraucean, Dayou Du, Igor Mordatch, Ivan Jurin, Lewis Liu, Ayush Dubey, Abhi Mohan, Janek Nowakowski, Vlad-Doru Ion, Nan Wei, Reiko Tojo, Maria Abi Raad, Drew A. Hudson, Vaishakh Keshava, Shubham Agrawal, Kevin Ramirez, Zhichun Wu, Hoang Nguyen, Ji Liu, Madhavi Sewak, Bryce Petrini, DongHyun Choi, Ivan Philips, Ziyue Wang, Ioana Bica, Ankush Garg, Jarek Wilkiewicz, Priyanka Agrawal, Xiaowei Li, Danhao Guo, Emily Xue, Naseer Shaik, Andrew Leach, Sadh MNM Khan, Julia

Wiesinger, Sammy Jerome, Abhishek Chakladar, Alek Wenjiao Wang, Tina Ornduff, Folake Abu, Alireza Ghaffarkhah, Marcus Wainwright, Mario Cortes, Frederick Liu, Joshua Maynez, Andreas Terzis, Pouya Samangouei, Riham Mansour, Tomasz Kępa, François-Xavier Aubet, Anton Algymr, Dan Banica, Agoston Weisz, Andras Orban, Alexandre Senges, Ewa Andrejczuk, Mark Geller, Niccolo Dal Santo, Valentin Anklin, Majd Al Merey, Martin Baeuml, Trevor Strohman, Junwen Bai, Slav Petrov, Yonghui Wu, Demis Hassabis, Koray Kavukcuoglu, Jeffrey Dean, and Oriol Vinyals. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, August 2024. URL http://arxiv.org/abs/2403.05530. arXiv:2403.05530.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long Range Arena: A Benchmark for Efficient Transformers, November 2020. URL http://arxiv.org/abs/2011.04006. arXiv:2011.04006.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. LongBench: A Bilingual, Multitask Benchmark for Long Context Understanding, June 2024. URL http://arxiv.org/abs/2308.14508. arXiv:2308.14508.

Chenxin An, Shansan Gong, Ming Zhong, Xingjian Zhao, Mukai Li, Jun Zhang, Lingpeng Kong, and Xipeng Qiu. L-Eval: Instituting Standardized Evaluation for Long Context Language Models, October 2023. URL http://arxiv.org/abs/2307.11088. arXiv:2307.11088.

Jiaqi Li, Mengmeng Wang, Zilong Zheng, and Muhan Zhang. LooGLE: Can

Long-Context Language Models Understand Long Contexts?, September 2024. URL http://arxiv.org/abs/2311.04939. arXiv:2311.04939.

gkamradt. gkamradt/LLMTest_needleinahaystack, November 2024. URL https://github.com/gkamradt/LLMTest_NeedleInAHaystack. original-date: 2023-11-11T00:50:02Z.

Xinrong Zhang, Yingfa Chen, Shengding Hu, Zihang Xu, Junhao Chen, Moo Hao, Xu Han, Zhen Thai, Shuo Wang, Zhiyuan Liu, and Maosong Sun. ınfty-Bench: Extending Long Context Evaluation Beyond 100K Tokens. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15262–15277, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.814. URL https://aclanthology.org/2024.acl-long.814.

Yingfa Chen, Xinrong Zhang, Shengding Hu, Xu Han, Zhiyuan Liu, and Maosong Sun. Stuffed Mamba: State Collapse and State Capacity of RNN-Based Long-Context Modeling, October 2024. URL http://arxiv.org/abs/2410.07145. arXiv:2410.07145.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 small, November 2022. URL http://arxiv.org/abs/2211.00593. arXiv:2211.00593.

Michael Hanna, Ollie Liu, and Alexandre Variengien. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. *Advances in Neural Information Processing Systems*, 36:76033–76060, December 2023. URL https://proceedings.neurips.cc/paper_fil

es/paper/2023/hash/efbba7719cc5172d175240f24be11280-Abstract-Conference.html.

Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards Automated Circuit Discovery for Mechanistic Interpretability. *Advances in Neural Information Processing Systems*, 36: 16318–16352, December 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/34e1dbe95d34d7ebaf99b9bcaeb5b2be-Abstract-Conference.html.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, Anthony Bau, and James Glass. What Is One Grain of Sand in the Desert? Analyzing Individual Neurons in Deep NLP Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6309–6317, July 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.33016309. URL https://ojs.aaai.org/index.php/AAAI/article/view/4592. Number: 01.

Nadir Durrani, Hassan Sajjad, Fahim Dalvi, and Yonatan Belinkov. Analyzing Individual Neurons in Pre-trained Language Models, October 2020. URL http://arxiv.org/abs/2010.02695. arXiv:2010.02695.

David Bau, Jun-Yan Zhu, Hendrik Strobelt, Agata Lapedriza, Bolei Zhou, and Antonio Torralba. Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences*, 117(48):30071–30078, December 2020. doi: 10.1073/pnas.1907375117. URL https://www.pnas.org/doi/abs/10.1073/pnas.1907375117. Publisher: Proceedings of the National Academy of Sciences.

Zeyu Yun, Yubei Chen, Bruno A. Olshausen, and Yann LeCun. Transformer visualization via dictionary learning: contextualized embedding as a linear

superposition of transformer factors, April 2023. URL http://arxiv.org/abs/2103.15949. arXiv:2103.15949.

Sam Marks. Some open-source dictionaries and dictionary learning infrastructure, December 2023. URL https://www.alignmentforum.org/posts/AaoWLcmpY3LKvtdyq/%20some-open-source-dictionaries-and-dictionary-learning.

Joseph Bloom. Open Source Sparse Autoencoders for all Residual Stream Layers of GPT2-Small, February 2024. URL https://www.lesswrong.com/posts/f9EgfLSurAiqRJySD/open-source-sparse-autoencoders-for-all-residual-stream.

Dan Mossing, Steven Bills, Henk Tillman, Tom Dupré la Tour, Nick Cammarata, Leo Gao, Joshua Achiam, Catherine Yeh, Jan Leike, Jeff Wu, and William Saunders. Transformer Debugger, 2024. URL https://github.com/openai/transformer-debugger.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping Ahead: Improving Reconstruction Fidelity with JumpReLU Sparse Autoencoders, August 2024. URL http://arxiv.org/abs/2407.14435. arXiv:2407.14435.

Glen Taggart. ProLU: A Nonlinearity for Sparse Autoencoders, April 2024. URL https://www.alignmentforum.org/posts/HEpufTdakGTTKgoYF/prolu-a-nonlinearity-for-sparse-autoencoders.

Dan Braun, Jordan Taylor, Nicholas Goldowsky-Dill, and Lee Sharkey. Identifying Functionally Important Features with End-to-End Sparse Dictionary

Learning, May 2024. URL http://arxiv.org/abs/2405.12241. arXiv:2405.12241.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-Time Intervention: Eliciting Truthful Answers from a Language Model. *Advances in Neural Information Processing Systems*, 36:41451–41530, December 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/81b8390039b7302c909cb769f8b6cd93-Abstract-Conference.html.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering Language Models With Activation Engineering, August 2023. URL https://ui.adsabs.harvard.edu/abs/2023arXiv230810248T. Publication Title: arXiv e-prints ADS Bibcode: 2023arXiv230810248T.

Yonatan Belinkov. Probing Classifiers: Promises, Shortcomings, and Advances. *Computational Linguistics*, 48(1):207–219, April 2022. ISSN 0891-2017. doi: 10.1162/coli_a_00422. URL https://doi.org/10.1162/coli_a_00422.

Arnab Sen Sharma, David Atkinson, and David Bau. Locating and Editing Factual Associations in Mamba, April 2024. URL http://arxiv.org/abs/2404.03646. arXiv:2404.03646 version: 1.

Jungo Kasai, Hao Peng, Yizhe Zhang, Dani Yogatama, Gabriel Ilharco, Nikolaos Pappas, Yi Mao, Weizhu Chen, and Noah A. Smith. Finetuning Pretrained Transformers into RNNs, September 2021. URL http://arxiv.org/abs/2103.13076. arXiv:2103.13076.

Michael Zhang, Kush Bhatia, Hermann Kumbong, and Christopher Ré. The

Hedgehog & the Porcupine: Expressive Linear Attentions with Softmax Mimicry, February 2024b. URL http://arxiv.org/abs/2402.04347. arXiv:2402.04347.

Jean Mercat, Igor Vasiljevic, Sedrick Keh, Kushal Arora, Achal Dave, Adrien Gaidon, and Thomas Kollar. Linearizing Large Language Models, May 2024. URL http://arxiv.org/abs/2405.06640. arXiv:2405.06640.

Aviv Bick, Kevin Y. Li, Eric P. Xing, J. Zico Kolter, and Albert Gu. Transformers to SSMs: Distilling Quadratic Knowledge to Subquadratic Models, August 2024. URL http://arxiv.org/abs/2408.10189. arXiv:2408.10189.

Junxiong Wang, Daniele Paliotta, Avner May, Alexander M. Rush, and Tri Dao. The Mamba in the Llama: Distilling and Accelerating Hybrid Models, August 2024b. URL http://arxiv.org/abs/2408.15237. arXiv:2408.15237.

Michael Zhang, Simran Arora, Rahul Chalamala, Alan Wu, Benjamin Spector, Aaryan Singhal, Krithik Ramesh, and Christopher Ré. LoLCATs: On Low-Rank Linearizing of Large Language Models, October 2024c. URL http://arxiv.org/abs/2410.10254. arXiv:2410.10254.

Richard E Ladner and Michael J Fischer. Parallel Prefix Computation. *Journal of the ACM (JACM)*, 27(4):831–838, 1980. Publisher: ACM New York, NY, USA.

Sivaramakrishnan Lakshmivarahan and Sudarshan K Dhall. *Parallel Computing Using the Prefix Problem*. Oxford University Press, 1994.

Albert Gu, Isys Johnson, Aman Timalsina, Atri Rudra, and Christopher Re. How to Train your HiPPO: State Space Models with Generalized Orthogonal Basis Projections. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=klK17OQ3KB.

Eric Nguyen, Karan Goel, Albert Gu, Gordon Downs, Preey Shah, Tri Dao, Stephen Baccus, and Christopher Ré. S4ND: Modeling Images and Videos as Multidimensional Signals with State Spaces. *Advances in neural information processing systems*, 35:2846–2861, 2022.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The Curious Case of Neural Text Degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.