RESONA: Improving Context Copying in Linear Recurrence Models with Retrieval

Xinyu Wang[♠*] **Linrui Ma**[♣*] **Jerry Huang**[♡◊*] **Peng Lu**[♡]
Prasanna Parthasarathi[♣] **Xiao-Wen Chang**[♠] **Boxing Chen**[♣] **Yufei Cui**[♣†]

♠ McGill University ♣ Noah's Ark Lab, Montreal ♡ Université de Montréal ◊ Mila

Abstract

Recent shifts in the space of large language model (LLM) research have shown an increasing focus on novel architectures to compete with prototypical Transformer-based models that have long dominated this space. Linear recurrent models have proven to be a viable competitor due to their computational efficiency. However, such models still demonstrate a sizable gap compared to Transformers in terms of in-context learning among other tasks that require recalling information from a context. In this work, we introduce RESONA, a simple and scalable framework for augmenting linear recurrent models with retrieval. RESONA augments models with the ability to integrate retrieved information from the provided input context, enabling tailored behavior to diverse task requirements. Experiments on a variety of linear recurrent models demonstrate that RESONA-augmented models observe significant performance gains on a variety of synthetic as well as real-world natural language tasks, highlighting its ability to act as a general purpose method to improve the in-context learning and language modeling abilities of linear recurrent LLMs.

1 Introduction

Improvements in building state-of-the-art large language models (LLMs) (OpenAI, 2024; Grattafiori et al., 2024; Qwen, 2024; Gemma Team, 2024) through increased scale (Chung et al., 2024; Kaplan et al., 2020) and downstream tuning (Ouyang et al., 2022; Dubois et al., 2023) have enabled them to attain human-level performance on a number of complex tasks. One feature that has enabled this is **in-context learning** (Brown et al., 2020), where models can use user-provided content to provide a specific response tailed to that example. This relies on the Transformer (Vaswani et al., 2017) backbone that underlies many of these models, enabling for models to observe the complete past when generating content.

Recently, linear recurrent models (LRMs) (Gu et al., 2022; Peng et al., 2023; Orvieto et al., 2023; Gu and Dao, 2024; Yang et al., 2024b) have emerged as an alternative, aiming to address computational bottlenecks associated with attention mechanisms (Bahdanau et al., 2015). Unlike Transformers, LRMs do not operate over all previous parts of the input. Instead they compress prior context into a unified hidden representation/state with a recurrent structure, similar to original recurrent neural networks (RNNs) (Rumelhart, 1989; Hochreiter and Schmidhuber, 1997; Cho et al., 2014; Balduzzi and Ghifary, 2016; Lu et al., 2019; Martin and Cundy, 2018), enabling more efficient inference. However, this unified representation introduces an information bottleneck, as it limits the capacity to represent the full range of vocabulary elements within a fixed-size state that does not scale with the combinatorial complexity of token sequences. This has raised questions about the ability of LRMs to effectively learn from input contexts (Jelassi et al., 2024; Park et al., 2024; Grazzi et al., 2024; Lu et al., 2023) and perform comparably to Transformer-based LLMs in such settings.

[*]Equal Contribution
[†]Corresponding author: yufei.cui@huawei.com

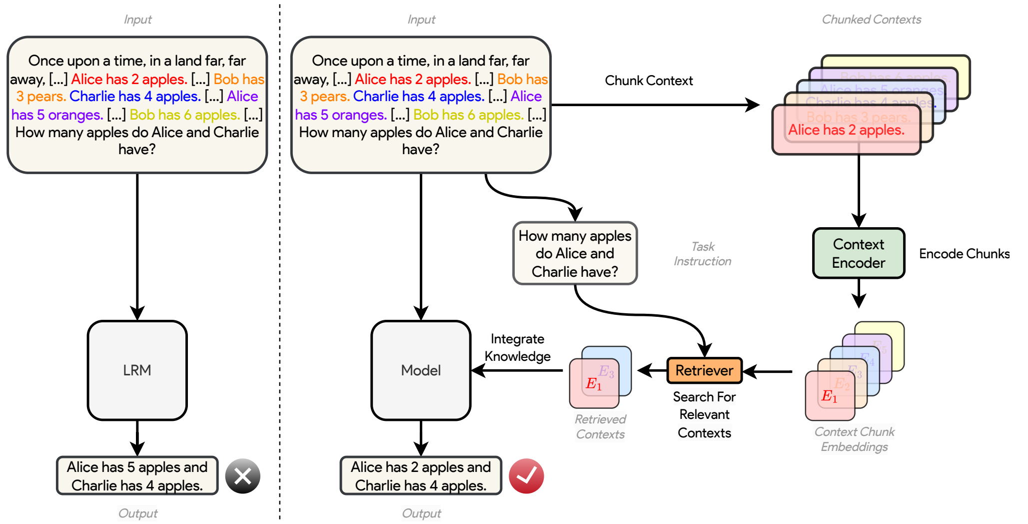


Figure 1: Simplified overview of RESONA. The input is separated into the example-specific context along with a task-specific instruction. The context is chunked and encoded, after which a retriever uses the instruction to determine which chunks are relevant to solving the task. The retrieved context is integrated into the model's reasoning for improved response.

Linear methods rely on fixed-size states to address limitations of traditional attention. A trade-off, however, is that these fixed-size states cannot perfectly preserve all historical information, making exact retrieval challenging. This manifests itself in practice in various tasks such as language modeling; with the key-value associative memory system that underlies such methods, adding new key-value associations leads to accumulating retrieval errors that hinder performance. These errors may build up in a variety of data-independent (Gu et al., 2022; Orvieto et al., 2023) or dependent (Yang et al., 2024a; Gu and Dao, 2024; Peng et al., 2024; Zhang et al., 2024; Beck et al., 2017) manners. While recent works have proposed strategies to mitigate such errors (Yang et al., 2024b; 2025; Sun et al., 2024), they retain a fixed-size hidden state that remains a fundamental constraint.

In an effort to bridge this gap, we propose RESONA, a retrieval-based method designed to improve context-based learning in LRMs (Figure 1). By introducing a retrieval mechanism that facilitates information flow from the context, RESONAmitigates the hidden state bottleneck and enables more effective in-context learning. Specifically, we augment LRM layers within the backbone model with a contextual search component. The input context is first chunked into passages, which are then retrieved based on the current LRM state. A knowledge integration module subsequently incorporates the retrieved passages into the model's output by directly modifying its representation. This architecture allows previous context to bypass the fixed-size hidden state constraint, improving information flow from context to generation. These processes (chunking, retrieval, and integration) are parallelized with the main LRM, ensuring easy adaptation and improved in-context learning across a variety of models.

Empirical results on a range of representative tasks, spanning both synthetic and real-world data, demonstrate that RESONA significantly improves the ability of LRMs to utilize context-specific information with minimal or no latency overhead. We evaluate RESONA on a diverse set of tasks, including synthetic retrieval and recall tasks, language modeling and question-answering tasks, across multiple scenarios such as pre-training and direct fine-tuning. Our analysis demonstrates the effectiveness of using RESONA for overall performance improvements as well as test-time model customizations, such as balancing the trade-off between efficiency and performance.

2 Related Works

Linear Recurrent Models. Despite vast improvements in building language models that solve real-world natural language tasks since the introduction of the Transformer (Vaswani et al., 2017), significant concerns remain about their efficiency and scalability. While this has spurred interest in rendering them more efficient (Katharopoulos et al., 2020; Dao et al., 2022; Yang et al., 2024a), LRMs (Gu et al., 2022; Orvieto et al., 2023; Qin et al., 2023; Gu and Dao, 2024; Dao and Gu, 2024; Peng et al., 2023; Sun et al., 2023; Lu et al., 2025; Yang et al., 2024b) have grown as a popular alternative due to highly efficient inference costs compared to attention-based alternatives while retaining the ability to be train on elements of a sequence in parallel, an issue with traditional recurrent models. Further attempts at leveraging advantages from both paradigms (Lieber et al., 2024; De et al., 2024; Dong et al., 2025) have also garnered interest, leading to further exploration of similar models.

Retrieval-Augmented Generation (RAG). RAG-based methods augment the input of an LM with passages retrieved from an outside source (Guu et al., 2020; Lewis et al., 2020). Such methods have significantly improved performance on knowledge-intensive tasks, where it is difficult to store the information required for strong performance explicitly within the model's parametric knowledge (Roberts et al., 2020). Further improvements have come under the form of improved filtering of retrieved passages (Asai et al., 2024; Ma et al., 2025), robustness to irrelevant passages (Yoran et al., 2024; Xu et al., 2024) or tuning of more components (Lin et al., 2024). However, RAG is not directly applicable to learning from contexts, as such methods do not search within the input-specific query but rather from an external database, leading to a critical failure point of such methods.

Linear Recurrent Models and Context Usage. Despite their practical benefits, questions have arisen regarding the ability of LRMs to learn from input contexts (Akyürek et al., 2024). Jelassi et al. (2024) show that they struggle to directly copy information from contexts due to their fixed-sized latent state. Park et al. (2024) further show that they can struggle at retrieval-based (Arora et al., 2024) in-context learning, only solving such tasks through the addition of attention. Such observations have extended to real-world data, where LRMs have been shown to exhibit similar difficulties as Transformer LLMs (Wang et al., 2024; Huang, 2025; Liu et al., 2024; Huang et al., 2025) for long contexts (Ivgi et al., 2023; Hsieh et al., 2024). Accordingly, we introduce RESONA as a potential solution that provides additional information flow paths from the context to the generated input, enabling better utilization of the context for problem-solving.

Memory-enhanced Transformers. Due to the quadratic complexity of self-attention with respect to the length of a sequence, Transformer models face significant computational challenges when processing long inputs. Numerous approaches have been proposed to enhance Transformers for long sequential data, both to reduce the time/space complexity of the models as well as to improve performance. Dai et al. (2019); Munkhdalai et al. (2024) segment long inputs into shorter sequences and process them recurrently. Mohtashami and Jaggi (2023) append landmark tokens to represent each block of input and uses group attention to select relevant information. Borgeaud et al. (2022) enhance Transformer performance with external data by leveraging a separated retriever module. Nunez et al. (2024) develop a Span-Expanded Attention for the hybridized attention model to retrieve the most relevant block and integrate it with the recent context for attention computation. However, it remains unclear whether and how retrieval-based modules can enhance the performance of generalized linear recurrent models such as GLA (Mao, 2022; Yang et al., 2024a; Lu et al., 2025), Mamba (Gu and Dao, 2024; Dao and Gu, 2024), RWKV (Peng et al., 2023), and DeltaNet (Schlag et al., 2021; Yang et al., 2024b).

3 RESONA

We introduce RESONA (Figure 2 and Algorithm 1) as a framework to enhance the context-copying ability of LRMs through retrieval, without sacrificing original performance and versatility. Our end-to-end training enables models to utilize the context as a retrieval base from which information can be extracted. This helps the model to first overcome the fixed-size latent space bottleneck by integrating information from the context directly into the hidden state. This is in contrast to traditional LRMs, where the information from the context must flow through the hidden state.

Algorithm 1 RESONA Algorithm.

Require: Model M
Input: Input $\boldsymbol{s} \in \mathbb{R}^T$ and LRM hidden states $\mathbf{H} \in \mathbb{R}^{L \times T \times H}$,
Output: $\boldsymbol{Y} \in \mathbb{R}^{T \times D}$
Embed sequence $\boldsymbol{s}$ into $\boldsymbol{X} \in \mathbb{R}^{T \times D}$.
for $l \leftarrow 1$ to L **do**
  $\boldsymbol{H} \leftarrow \mathbf{H}_l$ ▷ Hidden state of layer l
  if Layer l is an RESONA layer **then**
    $(\boldsymbol{H}, \boldsymbol{Y}^m) = M_l^{\text{LRM}}(\boldsymbol{X}, \boldsymbol{H})$
      ▷ LRM Output
    $\boldsymbol{M} = M_l^{\text{C-and-S}}(\boldsymbol{X}, \boldsymbol{H})$
      ▷ Chunk-and-Search
    $\boldsymbol{Y}^r = M_l^{\text{KI}}(\boldsymbol{X}, \boldsymbol{H}, \boldsymbol{M})$
      ▷ Knowledge Integration
    $\alpha = f(\boldsymbol{X})$
    $\boldsymbol{Y} = \alpha \cdot \boldsymbol{Y}^m + (1-\alpha)\boldsymbol{Y}^r$
  else
    $(\boldsymbol{H}, \boldsymbol{Y}) = M_l(\boldsymbol{X}, \boldsymbol{H})$
  $\boldsymbol{X} = \boldsymbol{Y}$
return $\boldsymbol{Y}$

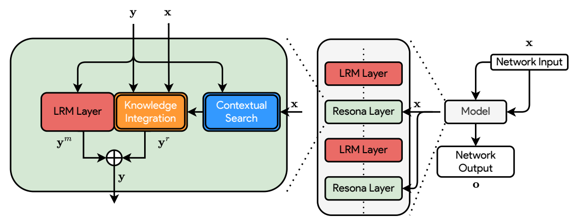


Figure 2: Primary components of RESONA: 1) **Contextual Search** which searches the context for relevant information and 2) **Knowledge Integration** which re-integrates the retrieved information into the model state. These enable information from the original network input to flow to arbitrary depths in the model, overcoming information decay within the model.

3.1 Problem Formulation and Overview

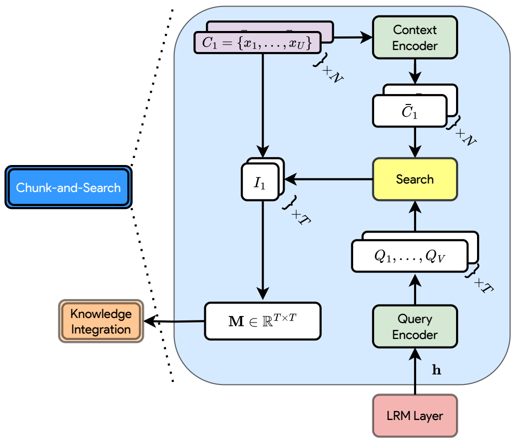
Let $\mathcal{V}$ be a vocabulary, i.e. a set of discrete elements, of size $|\mathcal{V}|$. A model M applies a function $f : \mathcal{V}^* \to \mathcal{V}^*$, taking as input a sequence of tokens from the vocabulary (of arbitrary length) while outputting a sequence of tokens from the vocabulary. We denote the input sequence as $\boldsymbol{x} = [x_1, \ldots, x_T]$, which we refer to as the model's prompt. The corresponding output sequence $\boldsymbol{y} = f(\boldsymbol{x})$ is referred to as the model's answer or generated response. Furthermore, a sequence-to-token mapping is a function $g : \mathcal{V}^* \to \mathcal{V}$ used to define f through auto-regressive inference. Specifically, given an input sequence $\boldsymbol{x} \in \mathcal{V}^*$, the output tokens are generated one at a time using the recurrence: $x_{i+j} = g(x_1, \ldots, x_{i+j-1})$ and $f(\boldsymbol{x}_{1:i}) = (x_{i+1}, x_{i+2}, \ldots)$, where $1 \leq i \leq T$ and $j \in \mathbb{N}$.

An LRM is defined by a state update rule $u : \mathcal{S} \times \mathcal{V} \to \mathcal{S}$ and an output function $r : \mathcal{S} \to \mathcal{V}$, where $\mathcal{S}$ is a finite set of states and a state is a representation of the system after processing a sequence from $\mathcal{V}^*$. Let $s_0 \in \mathcal{S}$ be some initial state. Given some sequence $\boldsymbol{x}$ of length L, the state of the model at iteration i is denoted by $S_i(x_1, \ldots, x_i)$ and the output token is denoted by $R_i(x_1, \ldots, x_i)$. These are defined recursively as:

1) $S_0(\emptyset) = s_0$,
2) $\boldsymbol{h}_i = S_i(x_1, \ldots, x_i) = u(S_{i-1}(x_1, \ldots, x_{i-1}), x_i)$,
3) $y_i = R_i(x_1, \ldots, x_i) = r(S_i(x_1, \ldots, x_i))$.

Information from $\boldsymbol{x}$ flows to $\boldsymbol{y}$ through a state $\boldsymbol{h} \in \mathbb{R}^{d_h}$ where d_h is fixed and finite. Thus for increasing sequence length or information dense settings, LRMs can struggle from the limited size of $\boldsymbol{h}$.

We observe that the LRM is directly limited by the size of its hidden state, which can be insufficient for modeling problems with many possible states, unless the hidden size grows with the size of the possible set of states. RESONA introduces two flexible components to overcome this constraint without sacrificing the primary benefits of LRMs (namely parallel training and inference time efficiency): 1) a **contextual search** operation that operates on the input to retrieve context-specific information and 2) a **knowledge integration** component that mixes the retrieved information with the LRM output.



Algorithm 2 Chunk-and-Search Algorithm.

Require: Context and Query Encoders $\mathcal{C}$, $\mathcal{Q}$

Input: Input $\boldsymbol{X} \in \mathbb{R}^{T \times D}$ and LRM hidden state $\boldsymbol{H} \in \mathbb{R}^{T \times H}$.

Output: Attention mask $\boldsymbol{M} \in \mathbb{R}^{T \times T}$

Chunk $\boldsymbol{X}$ into $\mathbf{X}' \in \mathbb{R}^{N \times U \times D}$

Use $\mathcal{C}$ to encode each context chunk $\{\boldsymbol{X}'_i\}_{i=1}^{N} \in \mathbf{X}'$ into context embeddings $\bar{\boldsymbol{C}} \in \mathbb{R}^{N \times E}$

Use $\mathcal{Q}$ to encode $\boldsymbol{H}$ into query embeddings $\bar{\boldsymbol{Q}} \in \mathbb{R}^{T \times E}$.

Compute chunk index sets for each $\bar{\boldsymbol{Q}}_j$: $\{I_j\}_{j=1}^{T} = \{\text{Top-}k(\bar{\boldsymbol{Q}}_j, \bar{\boldsymbol{C}})\}_{j=1}^{T}$

With $\{I_j\}_{j=1}^{T}$, compute a mask $\boldsymbol{M} \in \mathbb{R}^{T \times T}$ such that $\boldsymbol{M}_{ji} = 1 \iff \Big(i \in \{I_j\}\Big)$.

Figure 3: A breakdown of our Chunk-and-Search implementation. The initial input context is chunked while the hidden state of the LRM layer is used as a query. Corresponding indices are retrieved for each query, creating a mask that is used for Knowledge Integration.

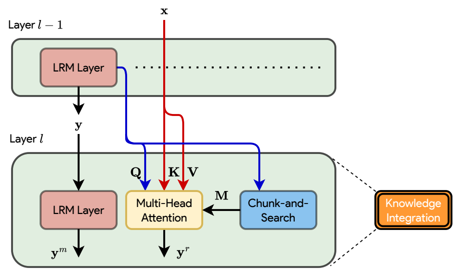
Contextual Search. In order to retrieve relevant context, RESONA implements contextual search as a chunk-and-search mechanism (Algorithm 2). The initial input $\boldsymbol{X}$ is first split into N chunks, each consisting of U tokens, to create $\mathbf{X}' \in \mathbb{R}^{N \times U \times D}$, where D is the model input dimension. First, each of these chunks C_i is encoded using a context encoder $\mathcal{C}$ into a context embedding $\bar{C}_i$. Simultaneously, the hidden state $\boldsymbol{H} \in \mathbb{R}^{T \times H}$ of the adjacent linear-recurrent layer, is used to encode a number of queries into query embeddings $\bar{Q}_{1:T}$ using a query encoder $\mathcal{Q}$. For each query, we search for the top-k similar contexts using a cosine-similarity search, which produces chunk indices that we can then use to retrieve the relevant input token positions. These are used to create a mask, which is passed to the **Knowledge-Integration** module.

Knowledge Integration. To integrate knowledge from the retrieved chunks, RESONA does as follows (Figure 4 and Algorithm 3). The knowledge integration module is a cross-attention module that can directly integrate information from the initial embedding into the LRM layer representation. To do so, the queries $\boldsymbol{Q}$ are directly computed from the hidden state of the prior LRM layer[1], while the keys $\boldsymbol{K}$ and values $\boldsymbol{V}$ are computed directly from the input embeddings $\boldsymbol{X}$ that directly follow after the initial embedding matrix $\boldsymbol{E}$. Within the cross-attention module, we use a mask computed from our **Chunk-and-Search** implementation of the contextual search. This ensures that the cross attention module can mix in only the most relevant information from the input back into the cross-attention module, producing an output $\boldsymbol{Y}^r \in \mathbb{R}^{T \times D}$, which is then integrated with the output of the adjacent LRM layer $\boldsymbol{Y}^m$, computed as

$$\boldsymbol{Y} = \alpha \cdot \boldsymbol{Y}^m + (1 - \alpha) \cdot \boldsymbol{Y}^r.$$

α can be computed on an input-dependent basis for each element of $\boldsymbol{Y}$ or can alternatively be set as fixed hyper-parameter value. Because only the chunk-and-search design, each query attends to at most kU elements from the initial input in k contiguous blocks, enabling us to compute $\boldsymbol{Y}^r$ efficiently using existing sparse attention mechanisms. For simplicity, we maintain the use of a fixed constant α for the experiments that follow.

[1] In the event that the first layer is augmented with RESONA, the queries are generated directly from the initial embeddings.



Algorithm 3 Knowledge Integration.

Require: Attention weights W_Q, W_K, W_V and output weights W_{out}

1: **Input:** Input $\boldsymbol{X} \in \mathbb{R}^{T\times D}$ and LRM hidden state $\boldsymbol{h} \in \mathbb{R}^{T\times H}$, mask $\boldsymbol{M} \in \mathbb{R}^{T\times T}$

2: **Output:** $\boldsymbol{Y}^r \in \mathbb{R}^{T\times D}$

3: From $\boldsymbol{h}$, compute queries $\boldsymbol{Q} \in \mathbb{R}^{T\times d_k}$ using W_Q. In parallel, compute $\boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{T\times d_k}$ from $\boldsymbol{X}$ with W_K, W_V.

4: Compute multi-head attention output $\boldsymbol{O} =$ CrossAttn $(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}, \boldsymbol{M})$, where $\boldsymbol{O} \in \mathbb{R}^{T\times d_k}$.

5: Project $\boldsymbol{O}$ using W_{out} into $\boldsymbol{Y}^r \in \mathbb{R}^{T\times D}$.

Figure 4: Knowledge Integration portion of RESONA. The hidden state is used as a query while the initial network input is used as the query and keys. A mask constructed from the contextual search is used to determine which information needs to be mixed back into the LRM representation, which is then used to compute an attention output that is integrated into the adjacent LRM output.

3.2 Training and Inference

When training RESONA, an important consideration is the auto-regressive nature of the model that therefore requires a causal mask. Specific to our implementation, to maintain this during the parallel nature of training, we ensure that for the query at index i, only chunks that solely contain information prior to this position in the input are considered within the **Chunk-and-Search** process. This ensures that the mask $\boldsymbol{M}$ allows no information from a given token to affect the representation of those ahead of it in the sequence.

During inference, tokens are dynamically chunked based on a predefined size and embedded into a chunked cache in parallel with the main model's embedding, introducing no extra latency. For long prompts, chunking is integrated into the pre-filling stage, aligning with the token embedding pipeline and minimizing computation overhead.

4 Experiments, Results and Analysis

4.1 Tasks and Datasets

To test our method, we evaluate RESONA on both a number of synthetic benchmarks as well as real-world language benchmarks. In Section 4, we explain the experimental setting as well as evaluation methods for each. For each setting, we report a standard baseline where a backbone model is not augmented with RESONA. These baselines vary based on which backbones are capable of adequately learning the task without RESONA.

4.2 Main Results

4.2.1 Results on Synthetic Benchmarks.

We first evaluate on synthetic benchmarks, namely multi-query associative recall (MQAR) (Arora et al., 2024) and the Mechanistic Architecture Design (MAD) suite of tasks (Poli et al., 2024). For each, we report accuracy on a held-out test set, where a correct answer requires the entire output is being correctly predicted. We initialize models from scratch and train them on the task of interest, in particular a 4 layer model with a vocabulary size of 8192. Each model uses a hidden size of 128 and a context chunk size of 64 for those augmented with RESONA. Models are trained using 20K and evaluated on 1K examples.

Figure 5 and Table 1 demonstrate that RESONA augmentations improves performance across all baselines, some by wide margins. Baseline models are often able to perfectly solve MQAR for shorter sequence lengths or a smaller number of KV-pairs, but fail catastrophically upon

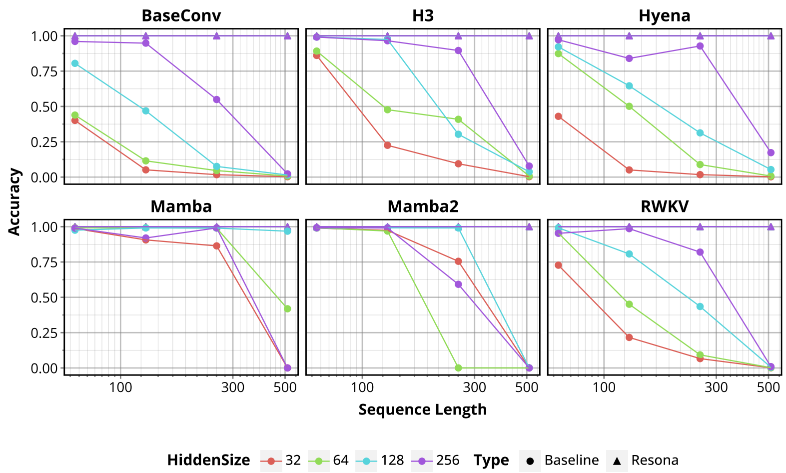


Figure 5: Results on MQAR tasks on varying sequence lengths. Baseline models remain limited in their ability to model increasingly long sequences even with increasing hidden size, whereas augmentation with RESONA perfect performance on arbitrary lengths.

Table 1: Performance on synthetic MAD (Poli et al., 2024) tasks. The best result for each metric is highlighted in bold. RESONA consistently boosts retrieval performance and shows gains even on compression and memorization tasks.

Model	Comp.	ICR	Noisy ICR	Fuzzy ICR	SC	Mem.	Average
Transformer	53.4	96.7	99.7	69.4	98.7	89.4	84.1
Mamba	**38.3**	76.7	74.9	9.3	33.2	88.5	53.5
+RESONA	38.2 (↓ 0.1)	**99.9** (↑ 23.2)	**100.0** (↑ 25.1)	**63.4** (↑ 54.1)	**42.7** (↑ 9.5)	**88.8** (↑ 0.3)	**72.1** (↑ 18.6)
Mamba2	43.6	96.4	96.7	21.1	93.3	86.9	73.0
+RESONA	**46.6** (↑ 3.0)	**100.0** (↑ 3.6)	**100.0** (↑ 3.3)	**62.9** (↑ 41.8)	**93.6** (↑ 0.3)	**88.1** (↑ 1.2)	**81.9** (↑ 8.9)
RWKV5	36.8	96.4	96.6	12.1	52.7	55.0	58.3
+RESONA	**40.4** (↑ 3.6)	**99.7** (↑ 3.3)	**99.8** (↑ 3.2)	**59.7** (↑ 47.6)	**58.0** (↑ 5.3)	**70.6** (↑ 15.6)	**71.5** (↑ 13.2)
Hyena	42.2	79.3	77.4	9.96	72.8	88.9	61.7
+RESONA	**42.6** (↑ 0.4)	**99.9** (↑ 20.6)	**99.9** (↑ 22.5)	**66.2** (↑ 56.2)	**74.3** (↑ 1.5)	**89.0** (↑ 0.1)	**78.7** (↑ 17.0)

increasing either of these values. RESONAmeanwhile retains near perfect accuracy even after these values. Similarly, models can struggle at specific tasks within the MAD suite, but RESONA achieves a significant improvement in performance. On tasks in which the base LRM models show strong performance, no degradation is observed through the addition of RESONA, highlighting its flexibility. We observe that this is consistent across multiple models (Poli et al., 2023; Fu et al., 2023), which are incapable of learning on some of the simpler settings but upon the addition of RESONA layers are able of maintaining nearly perfect accuracy for arbitrarily long sequences, showcasing its specific benefits in context-recall intensive settings.

4.2.2 *Language Modeling*

To assess language modeling, we compare a baseline models with one augmented with RESONA layers. Here, we train on the WIKITEXT-103 dataset (Merity et al., 2016). In order to train models augmented with RESONA, we also augment the dataset. Specifically, we first consolidate all samples from the same Wikipedia entry into single sample, eliminating excessively short title lines or empty lines. We then use a `LLaMA3.1-70B` (Grattafiori et al., 2024) model to augment each sample such that we can make use of the RESONA retrieval mechanism. We then conduct the Chunk-and-Search process offline to create masks, in order to save computation during training. To account for the additional parameters introduced by RESONA, we present results in the baseline settings for two version, one where each

Table 3: Results on QA benchmarks, where augmentation with RESONA improves performance over all evaluation metrics. **Hymba** indicates RESONA is added as a third branch with both the Hymba's original linear and attention branches, while **Hymba(256)** indicates where we reduce the window size of Hymba's sliding window attention from 1024 to 256, in order to create a fair comparison with our chunk size of 256.

Model	TRIVIAQA				COQA				NARRATIVEQA			
	BLEU	Rouge-L	Meteor	F1	BLEU	Rouge-L	Meteor	F1	BLEU	Rouge-L	Meteor	F1
Mamba	28.3	66.0	44.8	40.2	35.5	73.0	49.3	60.6	12.6	47.0	34.7	34.7
+RESONA	**30.7**	**68.0**	**45.6**	**41.8**	**44.3**	**75.0**	**52.2**	**61.2**	**13.2**	**50.0**	**36.9**	**36.2**
Hymba(256)	15.3	58.0	41.4	34.3	31.8	62.0	42.1	50.0	10.7	39.0	29.1	28.4
w/RETRIEVAL	2.9	57.0	39.0	33.9	4.6	52.0	38.2	41.3	3.9	36.0	30.8	27.6
w/RAG	12.9	66.0	46.0	38.8	23.2	67.0	47.9	54.5	12.5	49.0	38.4	35.2
+RESONA	**25.6**	**61.0**	**43.4**	**41.5**	**36.5**	**69.0**	**48.8**	**57.2**	**13.1**	**46.0**	**33.9**	**35.1**
Hymba	16.5	64.0	44.8	36.9	40.0	77.0	53.3	63.1	20.5	59.0	44.2	43.1
+RESONA	**29.3**	**69.0**	**48.4**	**45.4**	**51.7**	**82.0**	**59.3**	**68.9**	**20.9**	**60.0**	**45.2**	**44.0**

LRM layer matches exactly that of the augmented model as well as a version where the hidden size of the layer has been increased to match the parameter count of the augmented counterpart. Results in Table 2 demonstrates that integrating RESONA consistently achieves lower perplexity than baseline counterparts and their parameter-aligned variants, highlighting its applicability for language modeling. Notably, modifying Hymba's sliding window mask to our retrieval-based mask significantly improves performance. Furthermore, results on short-context tasks (Table 8) demonstrate no performance degradation, suggesting the ability to maintain performance on tasks that are not recall intensive.

4.2.3 *Direct Fine-Tuning*

To understand how well the addition of RESONA modules can improve performance on context-dependent tasks, we make use of pre-trained models in which we insert RESONA layers. The models are then fine-tuned directly, as described in Section 4.1, and we record performance using task-specific metrics. Due to some computational limitations, we provide results only for models where the baseline model is capable of performance above random on all benchmarks. In these settings, we choose 3 layers of the model to augment with RESONA. The models are then tuned using the corresponding training dataset of the task, before being tested on a held-out test set.

Table 2: Comparison of pre-training perplexity (PPL) on WikiText-103 across base LRM architectures, their parameter-aligned variants, and RESONA-enhanced versions.

Model	Param	PPL
GLA	131M	14.265
GLA (SP)	142M	14.223
+RESONA	145M	**13.892**
DeltaNet	131M	13.044
DeltaNet (SP)	142M	12.946
+RESONA	145M	**12.541**
RetNet	132M	15.471
RetNet (SP)	146M	15.431
+RESONA	142M	**14.742**
Mamba	140M	16.261
Mamba (SP)	157M	16.173
+RESONA	154M	**15.943**
Hymba(64)	133M	16.688
+RESONA(SP)	135M	**15.887**

Under this direct-fine-tuning setting, we evaluate on question answering benchmarks including NARRATIVEQA (Kociský et al., 2018), the Conversational Question Answering (COQA) challenge (Reddy et al., 2019) and TRIVIAQA (Joshi et al., 2017). For each task, we report results in terms of BLEU, ROUGE-L, Meteor (Banerjee and Lavie, 2005) and F1 scores. Table 3 presents these results, where we can observe improvements on both pre-trained Mamba and Hymba (with a sliding window of 256) models through the addition of RESONA layers. This is particularly evident with improvements across all metrics for all datasets, showing the general benefits that RESONA provides towards better context-dependent reasoning skills.

5 Analysis and Discussion

RESONA vs. RAG-like methods. A natural comparison to make with RESONA is typical RAG-like methods, which append the retrieved passages directly to the input prior to applying the model. To compare with such methods, we design the following methods with RESONA: **1)** Using the input context directly as the data-store from which a pre-trained retriever can retrieve passages directly that are used as context (RETRIEVAL) and **2)** we remove the Knowledge Integration component and instead directly append all (decoded) retrieved passages from the Chunk-and-Search procedure to the initial input (RAG). Rows 5 and 6 in Table 3 shows these results on a Hymba model, again on the tested QA datasets. The RESONA-augmented variant remains significantly more performant than the other alternatives, demonstrating the benefits that acting directly on the representations can have.

Relationship with Hybrid Methods. Some previous works have suggested methods that enable linear recurrent models to improve their in-context learning abilities. Park et al. (2024) introduce a MambaFormer architecture which interleaves self-attention and Mamba layers, improving the ability to learn in-context on tasks in which pure Mamba models struggle. Similarly, RESONA can be interpreted as a form of hybrid mixture of attention and recurrence, similar to Dong et al. (2025), with the difference lying in the frequency of attention and its sparsity within different layers. To better compare these two methods, we provide an ablation (Table 4) where we replace Hymba layers with RESONA. In this setting, 3 consecutive layers are trained in either setup. We observe a meaningful increase in performance on COQA, indicating that for such types of tasks, the mechanism presented by RESONA could be more robust and suitable on a number of real world tasks.

Table 4: A comparison of RESONA and Hymba (Dong et al., 2025). Three consecutive layers are un-freezed for fine-tuning. In the RESONA-augmented setup, the attention branch of the selected layers is modified to the RESONA retrieval mechanism. A sliding window size of 256 is used while the corresponding number indicates the start index of the replaced layers.

Model	**Layer**	CoQA BLEU	Rouge-L	Meteor	F1
Hymba	0	31.8	62.0	42.1	50.0
+RESONA	0	**36.5**	**69.0**	**48.8**	**57.2**
Hymba	8	25.1	62.0	42.5	48.8
+RESONA	8	**41.2**	**73.0**	**51.5**	**61.4**
Hymba	15	21.8	60.0	41.5	48.0
+RESONA	15	**37.1**	**70.0**	**48.9**	**57.7**
Hymba	23	5.9	41.0	27.8	32.3
+RESONA	23	**29.1**	**62.0**	**42.8**	**50.1**
Hymba	29	5.8	40.0	27.6	30.3
+RESONA	29	**31.2**	**57.0**	**39.2**	**44.5**

Ablating on the position of layers. Due to its nature, a natural question that emerges relates to the ease and effectiveness of determining the layers at which RESONA augmentations need to be made. The same results (Table 4) show that the placement of the 3 RESONA layers do not have a significant impact on the performance improvement relative to the baseline, which can be improved upon in nearly all ways in which the layers are selected. This highlights a level of robustness of the framework and method, hinting towards an ability to be used for a variety of additional tasks. This robustness suggests that the method is not overly sensitive to architectural fine-tuning, reducing the need for extensive hyperparameter optimization when integrating RESONA into existing models. Such flexibility is particularly advantageous in practical applications, where manual layer selection may be non-trivial or computationally expensive.

Efficiency Evaluation. Given the architectural modifications that following from the RESONA augmentations, we provide a comparison between the efficiency tradeoffs with backbone models. We conducted ablation experiments using **Mamba**, **DeltaNet** and **GLA**, comparing both backbone models and those augmented with RESONA to demonstrate these tradeoffs in Table 5-7. While the RESONA augmentations do lead to a marginal increase in each factor, the plug-in remains lightweight and does not add significant overhead in computation, particularly for longer sequences. Furthermore, direct comparison with a Transformer shows this method to be significantly more lightweight while retaining Transformer-like performance on our tasks.

Pre-filling Length	Transformer	Mamba	Mamba + Resona	DeltaNet	DeltaNet + Resona	GLA	GLA + Resona
2k	29	45	52	53	64	37	46
4k	34	77	88	59	78	43	64
8k	71	149	170	72	103	62	104
16k	173	294	349	106	181	109	202
32k	503	571	653	208	338	202	386
64k	1665	1118	1285	412	652	407	757
128k	6094	2257	2412	807	1289	806	1518

Table 5: The time (in milliseconds) used to pre-fill a context of a pre-specified length. Numbers are rounded to the nearest millisecond.

Pre-filling Length	Transformer	Mamba	Mamba + Resona	DeltaNet	DeltaNet + Resona	GLA	GLA + Resona
2k	2679	2972	3329	2945	3686	2749	3528
4k	2758	3044	3360	3042	3772	2777	3543
8k	2866	3155	3499	3023	3789	2774	3523
16k	3389	3164	3613	3057	3829	2869	3689
32k	5759	3491	3910	3080	4207	2912	4044
64k	11144	4119	4656	3171	4404	3145	4352
128k	24050	4747	5601	3611	5278	3509	5076

Table 6: The time (in milliseconds) taken to generate 128 tokens following a prespecified pre-filling length. Numbers are rounded to the nearest millisecond.

Pre-filling Length	Transformer	Mamba	Mamba + Resona	DeltaNet	DeltaNet + Resona	GLA	GLA + Resona
2k	3.1	2.8	4.0	2.9	4.8	2.9	3.6
4k	3.5	2.8	4.1	3.0	4.9	3.0	3.7
8k	4.5	3.1	4.4	3.3	5.2	3.2	3.9
16k	6.4	3.7	5.1	3.7	5.5	3.6	4.4
32k	10.2	4.9	6.5	4.6	6.6	4.3	5.4
64k	17.7	7.2	9.2	6.4	8.4	5.9	7.5
128k	32.9	11.9	14.7	10.2	12.3	9.1	11.7

Table 7: The total memory consumption (in GB) given a pre-specified pre-filling length.

6 Conclusion

In this work, we propose RESONA, a lightweight retrieval-based knowledge integration mechanism that significantly improves the ability of LRMs to use example-specific context. RESONA utilizes a novel mechanism to use the input-context as a retrieval data-store and integrate such information with the input during training and inference, enabling models to use it more effectively and overcome information bottleneck concerns. Across a number of both synthetic and real-world datasets, LRMs augmented with RESONA demonstrate significant performance gains compared to their base counterparts, demonstrating its ability to function as a general method applicable to broader scenarios.

References

E. Akyürek, B. Wang, Y. Kim, and J. Andreas. In-context language learning: Architectures and algorithms. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=3Z9CRr5srL.

S. Arora, S. Eyuboglu, A. Timalsina, I. Johnson, M. Poli, J. Zou, A. Rudra, and C. Ré. Zoology: Measuring and improving recall in efficient language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=LY3ukUANko.

A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=hSyW5go0v8.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

D. Balduzzi and M. Ghifary. Strongly-typed recurrent neural networks. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1292–1300, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/balduzzi16.html.

S. Banerjee and A. Lavie. METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In J. Goldstein, A. Lavie, C. Lin, and C. R. Voss, editors, *Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005*, pages 65–72. Association for Computational Linguistics, 2005. URL https://aclanthology.org/W05-0909/.

M. Beck, K. Pöppel, M. Spanring, A. Auer, O. Prudnikova, M. Kopp, G. Klambauer, J. Brandstetter, and S. Hochreiter. xlstm: Extended long short-term memory. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, December 10-15, 2024, Vancouver, Canada*, pages 5998–6008, 2017.

S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. van den Driessche, J. Lespiau, B. Damoc, A. Clark, D. de Las Casas, A. Guy, J. Menick, R. Ring, T. Hennigan, S. Huang, L. Maggiore, C. Jones, A. Cassirer, A. Brock, M. Paganini, G. Irving, O. Vinyals, S. Osindero, K. Simonyan, J. W. Rae, E. Elsen, and L. Sifre. Improving language models by retrieving from trillions of tokens. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvári, G. Niu, and S. Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR, 2022. URL https://proceedings.mlr.press/v162/borgeaud22a.html.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In A. Moschitti, B. Pang, and W. Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL, 2014. doi: 10.3115/V1/D14-1179. URL https://doi.org/10.3115/v1/d14-1179.

H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, Y. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, A. Castro-Ros, M. Pellat, K. Robinson, D. Valter, S. Narang, G. Mishra, A. Yu, V. Y. Zhao, Y. Huang, A. M. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei. Scaling instruction-finetuned language models. *Journal on Machine Learning Research*, 25: 70:1–70:53, 2024. URL https://jmlr.org/papers/v25/23-0870.html.

Z. Dai, Z. Yang, Y. Yang, J. G. Carbonell, Q. V. Le, and R. Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In A. Korhonen, D. R. Traum, and L. Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 2978–2988. Association for Computational Linguistics, 2019. doi: 10.18653/V1/P19-1285. URL https://doi.org/10.18653/v1/p19-1285.

T. Dao and A. Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=ztn8FCR1td.

T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/67d57c32e20fd0a7a302cb81d36e40d5-Abstract-Conference.html.

S. De, S. L. Smith, A. Fernando, A. Botev, G. Cristian-Muraru, A. Gu, R. Haroun, L. Berrada, Y. Chen, S. Srinivasan, G. Desjardins, A. Doucet, D. Budden, Y. W. Teh, R. Pascanu, N. D. Freitas, and C. Gulcehre. Griffin: Mixing gated linear recurrences with local attention for efficient language models, 2024. URL https://arxiv.org/abs/2402.19427.

X. Dong, Y. Fu, S. Diao, W. Byeon, Z. Chen, A. S. Mahabaleshwarkar, S.-Y. Liu, M. V. Keirsbilck, M.-H. Chen, Y. Suhara, Y. Lin, J. Kautz, and P. Molchanov. Hymba: A hybrid-head architecture for small language models, 2025. URL https://openreview.net/forum?id=A1ztozypga.

Y. Dubois, C. X. Li, R. Taori, T. Zhang, I. Gulrajani, J. Ba, C. Guestrin, P. Liang, and T. B. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/5fc47800ee5b30b8777fdd30abcaaf3b-Abstract-Conference.html.

D. Y. Fu, T. Dao, K. K. Saab, A. W. Thomas, A. Rudra, and C. Ré. Hungry hungry hippos: Towards language modeling with state space models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=COZDy0WYGg.

Gemma Team. Gemma 2: Improving open language models at a practical size, 2024. URL https://arxiv.org/abs/2408.00118.

A. Grattafiori et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

R. Grazzi, J. Siems, S. Schrodi, T. Brox, and F. Hutter. Is mamba capable of in-context learning? In *Third International Conference on Automated Machine Learning, AutoML 2024, Paris, France, September 9-12, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=rJhOG0P8nr.

A. Gu and T. Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling, COLM 2024, Philadelphia, PA, United States, October 7-9, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=tEYskw1VY2.

A. Gu, K. Goel, and C. Ré. Efficiently modeling long sequences with structured state spaces. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=uYLFoz1vlAC.

K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR, 2020. URL http://proceedings.mlr.press/v119/guu20a.html.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, 1997. doi: 10.1162/NECO.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

C. Hsieh, S. Sun, S. Kriman, S. Acharya, D. Rekesh, F. Jia, Y. Zhang, and B. Ginsburg. RULER: what's the real context size of your long-context language models? In *First Conference on Language Modeling, COLM 2024, Philadelphia, PA, United States, October 7-9, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

J. Huang. How well can a long sequence model model long sequences? comparing architectural inductive biases on long-context abilities. In *International Conference on Computational Linguistics, COLING 2025, January 19 – 24 2025, Abu Dhabi, UAE*, 2025.

J. Huang, P. Parthasarathi, M. Rezagholizadeh, B. Chen, and S. Chandar. Do robot snakes dream like electric sheep? investigating the effects of architectural inductive biases on hallucination, 2025. URL https://arxiv.org/abs/2410.17477.

M. Ivgi, U. Shaham, and J. Berant. Efficient long-text understanding with short-text models. *Trans. Assoc. Comput. Linguistics*, 11:284–299, 2023. doi: 10.1162/TACL_A_00547. URL https://doi.org/10.1162/tacl_a_00547.

S. Jelassi, D. Brandfonbrener, S. M. Kakade, and E. Malach. Repeat after me: Transformers are better than state space models at copying. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=duRRoGeoQT.

M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In R. Barzilay and M. Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1601–1611. Association for Computational Linguistics, 2017. doi: 10.18653/V1/P17-1147. URL https://doi.org/10.18653/v1/P17-1147.

J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 5156–5165. PMLR, 2020. URL http://proceedings.mlr.press/v119/katharopoulos20a.html.

T. Kociský, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette. The narrativeqa reading comprehension challenge. *Trans. Assoc. Comput. Linguistics*, 6:317–328, 2018. doi: 10.1162/TACL_A_00023. URL https://doi.org/10.1162/tacl_a_00023.

P. S. H. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

O. Lieber, B. Lenz, H. Bata, G. Cohen, J. Osin, I. Dalmedigos, E. Safahi, S. Meirom, Y. Belinkov, S. Shalev-Shwartz, O. Abend, R. Alon, T. Asida, A. Bergman, R. Glozman, M. Gokhman, A. Manevich, N. Ratner, N. Rozen, E. Shwartz, M. Zusman, and Y. Shoham. Jamba: A hybrid transformer-mamba language model, 2024. URL https://arxiv.org/abs/2403.19887.

X. V. Lin, X. Chen, M. Chen, W. Shi, M. Lomeli, R. James, P. Rodriguez, J. Kahn, G. Szilvasy, M. Lewis, L. Zettlemoyer, and W. Yih. RA-DIT: retrieval-augmented dual instruction tuning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=22OTbutug9.

N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang. Lost in the middle: How language models use long contexts. *Trans. Assoc. Comput. Linguistics*, 12:157–173, 2024. doi: 10.1162/TACL_A_00638. URL https://doi.org/10.1162/tacl_a_00638.

P. Lu, T. Bai, and P. Langlais. SC-LSTM: Learning task-specific representations in multi-task learning for sequence labeling. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2396–2406, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1249. URL https://aclanthology.org/N19-1249/.

P. Lu, S. Wang, M. Rezagholizadeh, B. Liu, and I. Kobyzev. Efficient classification of long documents via state-space models. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 6559–6565. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.EMNLP-MAIN.404. URL https://doi.org/10.18653/v1/2023.emnlp-main.404.

P. Lu, I. Kobyzev, M. Rezagholizadeh, B. Chen, and P. Langlais. Regla: Refining gated linear attention. In L. Chiruzzo, A. Ritter, and L. Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pages 2884–2898. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.NAACL-LONG.147. URL https://doi.org/10.18653/v1/2025.naacl-long.147.

S. Ma, C. Xu, X. Jiang, M. Li, H. Qu, C. Yang, J. Mao, and J. Guo. Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=oFBu7qaZpS.

H. H. Mao. Fine-tuning pre-trained transformers into decaying fast weights. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 10236–10242. Association for Computational Linguistics, 2022. doi: 10.18653/V1/2022.EMNLP-MAIN.697. URL https://doi.org/10.18653/v1/2022.emnlp-main.697.

E. Martin and C. Cundy. Parallelizing linear recurrent neural nets over sequence length. In *International Conference on Learning Representations*, 2018.

S. Merity, C. Xiong, J. Bradbury, and R. Socher. Pointer sentinel mixture models, 2016.

A. Mohtashami and M. Jaggi. Landmark attention: Random-access infinite context length for transformers. *CoRR*, abs/2305.16300, 2023. doi: 10.48550/ARXIV.2305.16300. URL https://doi.org/10.48550/arXiv.2305.16300.

T. Munkhdalai, M. Faruqui, and S. Gopal. Leave no context behind: Efficient infinite context transformers with infini-attention. *CoRR*, abs/2404.07143, 2024. doi: 10.48550/ARXIV.2404.07143. URL https://doi.org/10.48550/arXiv.2404.07143.

E. Nunez, L. Zancato, B. Bowman, A. Golatkar, W. Xia, and S. Soatto. Expansion span: Combining fading memory and retrieval in hybrid state space models. *CoRR*, abs/2412.13328, 2024. doi: 10.48550/ARXIV.2412.13328. URL https://doi.org/10.48550/arXiv.2412.13328.

OpenAI. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

A. Orvieto, S. L. Smith, A. Gu, A. Fernando, Ç. Gülçehre, R. Pascanu, and S. De. Resurrecting recurrent neural networks for long sequences. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 26670–26698. PMLR, 2023. URL https://proceedings.mlr.press/v202/orvieto23a.html.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. F. Christiano, J. Leike, and R. Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

J. Park, J. Park, Z. Xiong, N. Lee, J. Cho, S. Oymak, K. Lee, and D. Papailiopoulos. Can mamba learn how to learn? A comparative study on in-context learning tasks. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=GbFluKMmtE.

B. Peng, E. Alcaide, Q. Anthony, A. Albalak, S. Arcadinho, S. Biderman, H. Cao, X. Cheng, M. Chung, L. Derczynski, X. Du, M. Grella, K. K. GV, X. He, H. Hou, P. Kazienko, J. Kocon, J. Kong, B. Koptyra, H. Lau, J. Lin, K. S. I. Mantri, F. Mom, A. Saito, G. Song, X. Tang, J. S. Wind, S. Wozniak, Z. Zhang, Q. Zhou, J. Zhu, and R. Zhu. RWKV: reinventing rnns for the transformer era. In H. Bouamor, J. Pino, and K. Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 14048–14077. Association for Computational Linguistics, 2023. doi: 10.18653/V1/2023.FINDINGS-EMNLP.936. URL https://doi.org/10.18653/v1/2023.findings-emnlp.936.

B. Peng, D. Goldstein, Q. Anthony, A. Albalak, E. Alcaide, S. Biderman, E. Cheah, X. Du, T. Ferdinan, H. Hou, P. Kazienko, K. K. GV, J. Kocon, B. Koptyra, S. Krishna, R. M. Jr., N. Muennighoff, F. Obeid, A. Saito, G. Song, H. Tu, S. Wozniak, R. Zhang, B. Zhao, Q. Zhao, P. Zhou, J. Zhu, and R. Zhu. Eagle and finch: RWKV with matrix-valued states and dynamic recurrence. In *First Conference on Language Modeling, COLM 2024, Philadelphia, PA, United States, October 7-9, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=soz1SEiPeq.

M. Poli, S. Massaroli, E. Nguyen, D. Y. Fu, T. Dao, S. Baccus, Y. Bengio, S. Ermon, and C. Ré. Hyena hierarchy: Towards larger convolutional language models. In A. Krause,

E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 28043–28078. PMLR, 2023. URL https://proceedings.mlr.press/v202/poli23a.html.

M. Poli, A. W. Thomas, E. Nguyen, P. Ponnusamy, B. Deiseroth, K. Kersting, T. Suzuki, B. Hie, S. Ermon, C. Ré, C. Zhang, and S. Massaroli. Mechanistic design and scaling of hybrid architectures. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=GDp7Gyd9nf.

Z. Qin, S. Yang, and Y. Zhong. Hierarchically gated recurrent neural network for sequence modeling. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/694be3548697e9cc8999d45e8d16fe1e-Abstract-Conference.html.

Qwen. Qwen2.5 technical report, 2024. URL https://arxiv.org/abs/2412.15115.

S. Reddy, D. Chen, and C. D. Manning. Coqa: A conversational question answering challenge. *Trans. Assoc. Comput. Linguistics*, 7:249–266, 2019. doi: 10.1162/TACL_A_00266. URL https://doi.org/10.1162/tacl_a_00266.

A. Roberts, C. Raffel, and N. Shazeer. How much knowledge can you pack into the parameters of a language model? In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5418–5426. Association for Computational Linguistics, 2020. doi: 10.18653/V1/2020.EMNLP-MAIN.437. URL https://doi.org/10.18653/v1/2020.emnlp-main.437.

D. E. Rumelhart. *Parallel distributed processing, 9th Edition*. MIT Pr., 1989. ISBN 026268053X. URL https://www.worldcat.org/oclc/60445750.

I. Schlag, K. Irie, and J. Schmidhuber. Linear transformers are secretly fast weight programmers. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 9355–9366. PMLR, 2021. URL http://proceedings.mlr.press/v139/schlag21a.html.

Y. Sun, L. Dong, S. Huang, S. Ma, Y. Xia, J. Xue, J. Wang, and F. Wei. Retentive network: A successor to transformer for large language models. *arXiv preprint arXiv:2307.08621*, 2023.

Y. Sun, X. Li, K. Dalal, J. Xu, A. Vikram, G. Zhang, Y. Dubois, X. Chen, X. Wang, S. Koyejo, T. Hashimoto, and C. Guestrin. Learning to (learn at test time): Rnns with expressive hidden states, 2024. URL https://arxiv.org/abs/2407.04620.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

S. Wang, I. Kobyzev, P. Lu, M. Rezagholizadeh, and B. Liu. Resonance rope: Improving context length generalization of large language models. In *ACL (Findings)*, 2024.

F. Xu, W. Shi, and E. Choi. RECOMP: improving retrieval-augmented lms with context compression and selective augmentation. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=mlJLVigNHp.

S. Yang and Y. Zhang. Fla: A triton-based library for hardware-efficient implementations of linear attention mechanism, Jan. 2024. URL https://github.com/fla-org/flash-linear-attention.

S. Yang, B. Wang, Y. Shen, R. Panda, and Y. Kim. Gated linear attention transformers with hardware-efficient training. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024a. URL https://openreview.net/forum?id=ia5XvxFUJT.

S. Yang, B. Wang, Y. Zhang, Y. Shen, and Y. Kim. Parallelizing linear transformers with the delta rule over sequence length. In A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024b. URL http://papers.nips.cc/paper_files/paper/2024/hash/d13a3eae72366e61dfdc7eea82eeb685-Abstract-Conference.html.

S. Yang, J. Kautz, and A. Hatamizadeh. Gated delta networks: Improving mamba2 with delta rule. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=r8H7xhYPwz.

O. Yoran, T. Wolfson, O. Ram, and J. Berant. Making retrieval-augmented language models robust to irrelevant context. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=ZS4m74kZpH.

Y. Zhang, S. Yang, R.-J. Zhu, Y. Zhang, L. Cui, Y. Wang, B. Wang, F. Shi, B. Wang, W. Bi, P. Zhou, and G. Fu. Gated slot attention for efficient linear-time sequence modeling. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, December 10-15, 2024, Vancouver, BC, Canada*, 2024. URL https://openreview.net/forum?id=jY4PhQibmg.

A Additional Results

A.1 Detailed Pretraining Results

Table 8: Detailed `lm-evaluation-harness` evaluation results from pre-training.

	Param	Wiki.	ARC-C		ARC-E		Hella.		OBQA		PIQA		PM	RACE	Wino.	AVG
		ppl	acc	acc_n	acc	acc_n	acc	acc_n	acc	acc_n	acc	acc_n	acc	acc	acc	acc
GLA	131M	14.265	0.1843	0.221	0.3683	0.3502	0.2671	0.2692	0.184	0.268	0.5392	0.5109	0.336	0.2364	0.5028	0.3259
+(SP)	142M	14.223	0.2073	0.2483	0.3409	0.3418	0.2701	0.2696	0.158	0.234	0.5321	0.5141	0.358	0.2478	0.4949	0.3243
+RESONA	145M	**13.892**	0.192	0.2517	0.3405	0.3253	0.2646	0.2716	0.174	0.272	0.5457	0.5261	0.352	0.2469	0.5249	**0.3297**
DeltaNet	131M	13.044	0.2201	0.2628	0.2849	0.2912	0.2613	0.2607	0.146	0.262	0.5234	0.5038	0.35	0.2545	0.5059	0.3174
+(SP)	142M	12.946	0.2159	0.256	0.3001	0.2845	0.2592	0.2655	0.178	0.278	0.5305	0.5125	0.34	0.266	0.4728	0.3199
+RESONA	145M	**12.541**	0.2073	0.2449	0.2963	0.2963	0.2686	0.2757	0.156	0.272	0.5419	0.5283	0.334	0.2517	0.5012	**0.3210**

A.2 Detailed Supervised Fine-Tuning Results

Table 9: Results on Needle-in-a-Haystack (NIAH) using a haystack of varying sizes. Models are scored on performance on a continuous scale from 0 (worst) to 5 (best). In all settings, there is a single needle placed arbitrarily within the haystack. Different variants mean that the format of the needle or haystack changes, such as being a number, keyword or UUID sequence. Here $\alpha \times \mathsf{lr}$ denotes RESONA is trained with a learning rate multiplied by α.

Model	Setting	4K				8K				16K			
		V1	V2	V3	MV	V1	V2	V3	MV	V1	V2	V3	MV
Mamba	`Baseline`	5.00	1.80	0.75	0.898	0.65	0.45	0.15	0.458	0.00	**0.35**	**0.20**	0.494
	20×lr	5.00	**3.65**	**1.80**	**1.358**	**5.00**	**0.85**	**0.45**	0.528	**4.90**	0.05	0.00	**0.669**
	50×lr	5.00	3.00	1.10	1.267	4.95	0.60	0.20	**0.533**	4.00	0.05	0.00	0.550
DeltaNet	`Baseline`	2.15	2.95	1.45	1.301	1.90	1.45	0.70	0.321	1.00	0.25	0.00	0.883
	20×lr	5.00	4.10	0.60	0.876	5.00	**2.60**	**0.80**	**1.453**	5.00	**0.60**	**0.10**	**0.973**
	50×lr	**5.00**	**4.40**	**1.45**	**1.676**	**5.00**	1.25	0.35	1.312	**5.00**	0.45	0.05	0.885
GLA	`Baseline`	2.90	3.65	0.05	**1.389**	0.25	0.55	0.00	0.846	0.00	0.05	0.00	0.291
	20×lr	2.55	**3.90**	0.10	1.310	**0.30**	0.50	0.00	0.930	0.00	**0.10**	0.00	0.578
	50×lr	**3.00**	3.85	**0.15**	1.353	0.20	**0.60**	0.00	**0.900**	0.00	0.05	0.00	**0.657**

Table 10: Detailed `lm-evaluation-harness` evaluation results from supervised fine-tuning of different pre-trained LRMs. After undergoing the same SFT as the backbone models, RESONA-enhanced models achieve comparable or superior zero-shot lm-harness evaluation scores to baselines. Combined with Table 9, these results demonstrate that the RESONAmodule enhances the backbone's in-context learning (ICL) capability while maintaining its foundational language modeling performance.

	ARC-C		ARC-E		Hella.		OBQA		PIQA		PM	RACE	Wino.	AVG
	acc	acc_n	acc	acc_n	acc	acc_n	acc	acc_n	acc	acc_n	acc	acc	acc	acc
GLA	0.2381	0.2747	0.5442	0.5046	0.3852	0.4903	0.198	0.314	0.7008	0.7008	0.552	0.3110	0.5288	0.4417
+RESONA	0.2466	0.2918	0.5497	0.5227	0.3738	0.4722	0.188	0.308	0.6944	0.7010	0.550	0.3139	0.5399	**0.4424**
DeltaNet	0.2363	0.2637	0.5636	0.5341	0.3852	0.4893	0.198	0.316	0.7035	0.7002	0.552	0.3388	0.5375	0.4475
+RESONA	0.2440	0.2722	0.5812	0.5455	0.3909	0.4982	0.198	0.314	0.7024	0.6997	0.556	0.3292	0.5375	**0.4514**
Mamba	0.3558	0.3805	0.6953	0.6427	0.4958	0.6490	0.270	0.382	0.7535	0.7535	0.684	0.3598	0.6440	0.5435
+RESONA	0.3823	0.3951	0.6907	0.6904	0.4804	0.6292	0.300	0.406	0.7372	0.7383	0.690	0.3445	0.6338	**0.5475**

A.3 Training Details for Synthetic Benchmarks

Multi-Query Associative Recall (MQAR). We evaluate RESONA on the MQAR task by training six base architectures: **BaseConv**, **H3**, **Hyena**, **Mamba**, **Mamba2**, and **RWKV**. All models are trained with a 4-layer configuration and a hidden dimension (*d_model*) ranging from 32 to 256. The sequence lengths vary from 64 to 512, with the number of key-value (KV) pairs corresponding to 4–32, respectively. To integrate RESONA, we insert the **Resona Layer** into either the first or third layer of each model, using the same *d_model* settings. The learning rate is swept using the default settings from Arora et al. (2024). For the **Resona Layer**, we use a chunk size of 2 and a top-k value of 1. The exact results for each model, sequence length, and hidden dimension can be found in Table 11.

Mechanistic Architecture Design (MAD) Suite. For the MAD tasks, we adopt a 4-hybrid block configuration, where each block consists of a linear recurrent layer followed by a

Table 11: Best Accuracy for Different Models with Varying Sequence Length and Model Dimensions

Model	$L = 64$				$L = 128$				$L = 256$				$L = 512$			
	KV Pairs				KV Pairs				KV Pairs				KV Pairs			
	32	64	128	256	32	64	128	256	32	64	128	256	32	64	128	256
BaseConv	0.401	0.439	0.805	0.960	0.051	0.115	0.469	0.948	0.017	0.046	0.076	0.549	0.003	0.007	0.015	0.023
+RESONA0	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
+RESONA2	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
H3	0.861	0.892	0.991	0.991	0.225	0.477	0.974	0.965	0.094	0.409	0.303	0.896	0.003	0.007	0.038	0.078
+RESONA0	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
+RESONA2	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Hyena	0.430	0.874	0.922	0.972	0.051	0.501	0.646	0.840	0.018	0.089	0.313	0.928	0.002	0.007	0.053	0.173
+RESONA0	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
+RESONA2	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
RWKV	0.727	0.955	0.992	0.953	0.216	0.451	0.807	0.986	0.066	0.093	0.435	0.820	0.001	0.002	0.005	0.010
+RESONA0	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
+RESONA2	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Mamba	0.987	0.992	0.975	0.990	0.906	0.992	0.991	0.920	0.864	0.990	0.990	0.991	0.000	0.419	0.968	0.000
+RESONA0	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
+RESONA2	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Mamba2	0.990	0.992	0.991	0.993	0.974	0.969	0.992	0.991	0.755	0.000	0.991	0.592	0.000	0.000	0.001	0.000
+RESONA0	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
+RESONA2	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

SwiGLU layer. We follow the benchmark settings from Poli et al. (2024), using a batch size of 128 and a learning rate range of 1e-3 to 1e-4. For the RESONA layers, we set the chunk size to 6 and the top-k value to 1.

A.4 Training Details for Real-World Tasks

Pretraining. During pre-training, we prioritized maintaining consistent model depth across architectures. For `GLA`, `DeltaNet`, and `RetNet`, we adopted the architecture implementations from FLA-Hub (Yang and Zhang, 2024), configuring them with 24 layers and a hidden size of 600. For `Hymba`, we use NVIDIA's official 150M parameter implementation (24 layers with a hidden size of 512). For Mamba, we employed FLA-Hub's implementation with 48 layers and a hidden size of 600. For comparative models with equivalent parameter counts, we adjusted the hidden size from 600 to 640. Following the methodology outlined in the `Hymba` paper, we integrated RESONAmodules at the shallowest, middle, and deepest layers to reinforce critical information flow. Each model was trained for 8,000 steps, with model selection performed using a dedicated validation set. The training configuration employed a cosine annealing scheduler with warmup over 5% of the training steps, the AdamW optimizer (learning rate of 1e-3 and weight decay = 0.01), and gradient clipping of 1.0.

Finetuning. We employed a base learning rate of 1e-5 for fine-tuning on the three individual datasets, with other training configurations remaining similar to those used in pre-training, except that RESONAmodules benefited from a higher learning rate. For CoQA, NarrativeQA, and TriviaQA, we trained for 2K, 8K, and 10K steps, respectively. During general supervised fine-tuning, we created a unified training set by shuffling 10K samples from each of the three QA datasets, ensuring significant diversity in sequence length and content. Additionally, we constructed a general test set by selecting 500 validation samples from each dataset. We trained on this combined dataset for 4K steps.

A.5 NIAH Scoring Details

For the NIAH (Needle In A Haystack) task, we employ an automated scoring protocol based on prefix matching. The scoring methodology operates as follows: A full score of 5 points is awarded if the model's response contains the complete and accurate needle. If not, we iteratively truncate the needle from the end (removing the last few characters incrementally) and perform prefix matching. Partial credit (a proportional fraction of the

5-point maximum) is granted when a truncated prefix matches exactly, with the remaining character percentage determining the awarded score. Responses containing no matching prefix of the needle receive 0 points. The final task score is obtained by averaging the scores across 100 test samples.