State Space Models are Strong Text Rerankers

Jinghua Yan[◇ 1] **Zhichao Xu**[◇ 12] **Ashim Gupta**[1] **Vivek Srikumar**[1]
[1] Kahlert School of Computing, University of Utah
[2] Scientific Computing and Imaging Institute, University of Utah
{jhyan,brutusxu,ashim,svivek}@cs.utah.edu

Abstract

Transformers dominate NLP and IR; but their inference inefficiencies and challenges in extrapolating to longer contexts have sparked interest in alternative model architectures. Among these, state space models (SSMs) like Mamba offer promising advantages, particularly $O(1)$ time complexity in inference. Despite their potential, SSMs' effectiveness at text reranking — a task requiring fine-grained query-document interaction and long-context understanding — remains underexplored.

This study benchmarks SSM-based architectures (specifically, Mamba-1 and Mamba-2) against transformer-based models across various scales, architectures, and pre-training objectives, focusing on performance and efficiency in text reranking tasks. We find that (1) Mamba architectures achieve competitive text ranking performance, comparable to transformer-based models of similar size; (2) they are less efficient in training and inference compared to transformers with flash attention; and (3) Mamba-2 outperforms Mamba-1 in both performance and efficiency. These results underscore the potential of state space models as a transformer alternative and highlight areas for improvement in future IR applications.[1]

1 Introduction

The transformer architecture (Vaswani et al., 2017) is an established standard within NLP and IR community. Compared to recurrent neural networks (RNNs) transformers better capture long-range dependencies and also admit large scale pre-training. However, for inference with a sequence of length L and D-dimensional hidden states, transformers cost $O(L)$ time and $O(LD)$ space complexity — proving to be less efficient than RNNs.

Recently, there has been a growing interest in developing alternative architectures for modeling sequence data. For example, RWKV (Peng et al., 2023) combines the efficient parallelizable training of transformers with the efficient inference of RNNs. Another notable architecture is the state space model (SSM, Gu and Dao, 2023; Gu et al., 2020, 2021b), which is related to convolutional and recurrent neural networks, and also to signal processing literature.

In essence, state space models compress the context into a smaller state of size N, achieving $O(1)$ time complexity and $O(ND)$ space complexity in inference time. However, the capabilities of SSMs are limited by the amount of information that can be compressed in its hidden state. To mitigate this, Gu and Dao (2023) propose a novel selective state space model named Mamba. Mamba selectively encodes the input to the hidden state to improve model expressiveness, while also addressing the computation problem with a selective scan method and hardware-aware optimization. Gu and Dao (2023) and followup work (Dao and Gu, 2024; Zhu et al., 2024; Wang et al., 2024; Waleffe et al., 2024, *inter alia*) examine the efficacy of Mamba models for various sequence modeling tasks, notably language modeling, and also image and audio tasks. The parameterized SSMs are able to achieve performance close to transformer-based models of similar sizes while also demonstrating efficiency in training and inference.

Despite the growing popularity of state space models, their effectiveness in information retrieval remains underexplored. Modern search systems typically consist of at least two stages: retrieval and reranking. During retrieval, offline indexes first fetch a preliminary list of candidate documents, which is refined by the reranking model. Reranking requires models to understand long context input, and to capture fine-grained query-document interactions. The attention mechanism in the transformer

[◇] Equal Contribution, order decided randomly.
[1] The code for reproducing our experiments is available at https://github.com/zhichaoxu-shufe/RankMambaV2

Proceedings of the 10th Workshop on Representation Learning for NLP (RepL4NLP-2025), pages 152–169

naturally allows for the latter; it allows query tokens to attend to document tokens. In contrast, state space models may fail to model long-range dependencies due to their recurrent nature.

In this paper, we examine the following research questions about Mamba-1 and Mamba-2:

1. **Performance RQ**: How do Mamba models compare to transformers for text reranking?

2. **Efficiency RQ**: How efficient are the Mamba architectures with respect to training throughput and inference speed?

To this end, we conduct a rigorous benchmarking study comparing the two model families, across varying architectures, sizes, pre-training objectives, and attention patterns. Specifically, we train neural reranking models following established training methodologies outlined in prior literature (Gao et al., 2021; Boytsov et al., 2022; Ma et al., 2023). Our experiments allow us to address the two research questions above. We find that:

1. Mamba-based language models can achieve strong text reranking performance, matching transformer-based models of similar scales.

2. Although Mamba architectures have better complexity theoretically, in practice they are less efficient compared to transformer architecture with I/O-aware optimization (e.g., flash attention (Dao, 2024)).

3. Mamba-2 improves upon Mamba-1 in both performance and efficiency.

We discuss the implications of our results and point out future directions of transformer alternative architectures for IR tasks.

2 Background: State Space Models

We will briefly survey state space models and their connection to RNNs and transformers. We use Structured State Space Sequence Models (S4, Gu et al., 2021a) to illustrate the idea behind state space models before describing the Mamba models.

State space models. In its simplest form, an SSM maps a 1-dimensional function or sequence $x(t) \in \mathbb{R}$ to $y(t) \in \mathbb{R}$ via a latent state $h(t) \in \mathbb{R}^N$. Here, t denotes a timestep and N is the state size (different from hidden dimensionality D). It is parameterized by $(\Delta, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ and defines a *continuous* sequence-to-sequence transformation as:

$$h'(t) = \boldsymbol{A}h(t) + \boldsymbol{B}x(t) \qquad y(t) = \boldsymbol{C}h(t) \tag{1}$$

The above transformation can be *discretized* as:

$$h_t = \overline{\boldsymbol{A}}h_{t-1} + \overline{\boldsymbol{B}}x_t \qquad y_t = \boldsymbol{C}h_t \tag{2}$$

The discretization of $\overline{\boldsymbol{A}}$ and $\overline{\boldsymbol{B}}$ is defined by the *discretization rule*, for example:

$$\overline{\boldsymbol{A}} = \exp(\Delta\boldsymbol{A}) \tag{3}$$

$$\overline{\boldsymbol{B}} = (\Delta\boldsymbol{A})^{-1}(\exp(\Delta\boldsymbol{A}) - I) \cdot \Delta\boldsymbol{B} \tag{4}$$

Expanding Eqn. (2) with the whole sequence $x = (x_1, x_2, \ldots, x_n)$ leads to a convolutional form:

$$y = x * \overline{\boldsymbol{K}} \tag{5}$$

$$\overline{\boldsymbol{K}} = (\boldsymbol{C}\overline{\boldsymbol{B}}, C\overline{\boldsymbol{A}\boldsymbol{B}}, \ldots, C\overline{\boldsymbol{A}}^{n-1}\overline{\boldsymbol{B}}) \tag{6}$$

While Eqn. (2) resembles an RNN, Eqn. (5) looks like a CNN, where $\overline{\boldsymbol{K}}$ is a large convolution kernel over the whole input sequence x. The parameterization of $(\Delta, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ is independent of input sequence x and is fixed during all time steps, a property referred to as linear time invariance (LTI). Structured state space models (S4) imposes a structure on the $\boldsymbol{A}$ matrix for efficiency. Existing works (Gu et al., 2021a; Gupta et al., 2022; Smith et al., 2022) employ a diagonal matrix, thus $\boldsymbol{A} \in \mathbb{R}^{N\times N}$, $\boldsymbol{B} \in \mathbb{R}^{N\times 1}$ and $\boldsymbol{C} \in \mathbb{R}^{1\times N}$ matrices are all represented by N parameters.

The above expressions can be generalized to D-channel features, i.e., $x_t, y_t \in \mathbb{R}^D$. A concrete example might be D-dimensional word embeddings or hidden states. In this case, computation of $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ is applied to each channel independently.

Mamba-1 Models. State space models compress potentially unbounded context into a state $h_t \in \mathbb{R}^N$, potentially limiting their effectiveness. Gu and Dao (2023) propose to make the parameters $(\Delta, \boldsymbol{B}, \boldsymbol{C})$ input-dependent. This modification changes the model from time-invariant to time-varying, therefore posing challenges to the model's computational efficiency; the model now cannot be trained in CNN mode. Gu and Dao (2023) address this via a hardware-aware optimization algorithm called *Selective Scan*. We refer the reader to the original paper for details.

Scalar Structured SSM. Mamba-2 (Dao and Gu, 2024) restricts the matrix $\boldsymbol{A}$ to be a scalar times identity matrix; i.e., all the diagonal elements of $\boldsymbol{A}$ are the same value. It also introduces a new hyperparameter P, the SSM head dimension, which is analogous to the transformer head dimension, i.e., $D = P \times \#\text{heads}$. Mamba-2 uses different $(\Delta, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ for each SSM head, and P is set to 64 or 128, similar to transformers. Further, Dao and Gu (2024) develop efficient implementations for training and inference, enabling much larger state size (from $N = 16$ in Mamba-1 to $N = 64, 256$ or larger in Mamba-2), while simultaneously being faster in training. Subsequent works (Yang et al., 2024; Qin et al., 2024; Dao and Gu, 2024, *inter alia*) also reported Mamba-2's performance and efficiency improvement over Mamba-1.

3 The Text Reranking Problem

Modern IR systems employ a two-stage retrieval-reranking pipeline (Schütze et al., 2008; Zhang et al., 2021; Asai et al., 2024; Xu et al., 2025, *inter alia*). After the initial retrieval by an efficient, scalable first-stage retriever, a reranker refines the ranklist to optimize ranking metrics. Reranking involves ordering texts (passages or documents) by their *relevance* to a query, with passage reranking being a finer-grained form of document reranking. Our focus is to study this second stage and perform a comprehensive analysis of different rerankers for the tasks of both passage reranking and document reranking.

Let q be an input query, and $d \in \mathcal{D}$ be a text, where $\mathcal{D}$ is the set of all texts (passages for passage reranking and documents for document reranking). The reranking model $f_\theta(q, d)$, parameterized by θ, predicts a scalar relevance score. The model f is instantiated as a linear layer on top of a language model. We adopt the common practice of concatenating the query and the document as input to the model (Nogueira et al., 2019; Yates et al., 2021; Boytsov et al., 2022; Ma et al., 2023, *inter alia*).

Training a Reranker. Training the reranking model involves sampling negatives from the document collection. We use the recommended setup from literature (Gao et al., 2021; Ma et al., 2023; Boytsov et al., 2022; Xu, 2024) to sample hard negatives from the retrieval results obtained from the first-stage retriever.

Let us denote the relevant document to query q_i as d_i^+, and sampled negatives as $d_i^- \in \mathcal{D}_i^-$, training pair $(q_i, d_i^+) \in \mathcal{S}$, the reranking model is trained with optimizing the following softmax loss:

$$-\frac{1}{|\mathcal{S}|} \sum_{(q_i, d_i^+) \in \mathcal{S}} \log \frac{\exp f_\theta(q_i, d_i^+)}{\exp f_\theta(q_i, d_i^+) + \sum_{j \in \mathcal{D}_i^-} \exp f_\theta(q_i, d_i^-)} \tag{7}$$

We pack multiple training instances into a mini-batch and jointly optimize the backbone language model and the linear layer.

4 Experiments

In this section, we describe the experimental setup for passage (§ 4.1) and document reranking (§ 4.3). For **Performance RQ**, we report results and analyze the implications in § 4.2 and § 4.4 respectively. Then we address **Efficiency RQ** in § 4.5.

4.1 Passage Reranking

First, let us examine the passage reranking task.

Datasets and Evaluation Metrics. We employ the passage ranking subset of the well-known MS MARCO dataset (Bajaj et al., 2016) which contains a total of 524K training instances. For the passage retriever in the first stage, we use `BGE-large-en-v1.5` (Xiao et al., 2023) due to its strong trade-off between retrieval performance and size of the retriever. (See Table 7 in the appendix for this comparison). The training set for our passage reranker is constructed by uniformly sampling 15 hard negatives from the ranklist of top-1000 passages returned by the BGE retriever. Zhuang et al. (2023); Ma et al. (2023) highlight that increasing the number of negatives along with the global batch size leads to better ranking performance. Training demands significant GPU RAM. We determined these hyperparameters by balancing performance and the available hardware resources.

The in-domain evaluation is conducted using the official passage ranking development set (`Dev`) containing 6,980 queries. We also include TREC DL19/DL20 (Craswell et al., 2020, 2021) evaluation set that contains 43/54 queries with in-depth annotation for passage ranking. We report the official evaluation metrics for passage ranking, i.e., MRR@10 for Dev and NDCG@10 for DL19/DL20. For out-of-domain evaluation, we use 13 publicly available BEIR testsets (Thakur et al., 2021) covering different text domains. Again, we report the official evaluation metric NDCG@10. All evaluations involve first constructing the index with the first-stage retriever, then retrieving pas-

Model	Size	Architecture	Pre-train #Tokens
Encoder-only Models *(Bi-directional)*			
BERT-base	110M	Transformer	3.3B
RoBERTa-base	120M	Transformer	33B
ELECTRA-base	105M	Transformer	3.3B
BERT-large	330M	Transformer	3.3B
RoBERTa-large	335M	Transformer	33B
ELECTRA-large	320M	Transformer	33B
Encoder-Decoder Models *(Bi-directional)*			
BART-base	130M	Transformer	33B
BART-large	385M	Transformer	33B
Decoder-only Models *(Uni-directional)*			
OPT-125M	125M	Transformer	180B
Mamba-1-130M	130M	Mamba-1	300B
Mamba-2-130M	130M	Mamba-2	300B
OPT-350M	350M	Transformer	180B
Mamba-1-370M	370M	Mamba-1	300B
Mamba-2-370M	370M	Mamba-2	300B
Mamba-1-790M	790M	Mamba-1	300B
Mamba-2-780M	780M	Mamba-2	300B
OPT-1.3B	1.3B	Transformer	180B
Mamba-1-1.4B	1.4B	Mamba-1	300B
Mamba-2-1.3B	1.3B	Mamba-2	300B
Llama-3.2-1B	1.3B	Transformer++	15T

Table 1: Details of the Pre-trained language models used in our comparative study. Transformer++ indicates the state-of-the-art transformer architecture. Note the pre-training #tokens is not directly comparable between encoder-only, encoder-decoder and decoder-only models due to different pre-training objectives.

sages with the retriever, followed by refining the ranklist using our trained rerankers.

Language Models Used. We conduct a comparative study between rerankers using state space models and several previously studied language models. Among encoder-only models, we use BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and ELECTRA (Clark et al., 2020) with their `base` as well as `large` variants. For encoder-decoder models, we select both `base` and `large` variants of the BART model (Lewis et al., 2020). Among decoder-only models, we compare with OPT (Zhang et al., 2022), and Llama3 (Dubey et al., 2024) models. The Llama3 model serves as an upper bound for transformer-based models, given that is the state-of-the-art pre-trained model at the 1B scale and high pre-training cost. We compare these with both Mamba-1 and Mamba-2-based rerankers at four different parameter settings. The details of the models used in our comparison study are shown in Table 1.

This extensive selection of pre-trained language models enables the comparison across different architecture types (e.g., encoder-only vs decoder-only), pre-trained model scales (from 110M to 1.4B parameters), as well as different pre-training objectives (e.g., masked language modeling in BERT vs replaced token detection in ELECTRA). It is important to acknowledge that the pre-trained models are trained under different pre-training setups (such as varying datasets and hyperparameter configurations, etc.); comparing them is not entirely fair. Nevertheless, we can gain insights by evaluating different language models using a consistent fine-tuning approach.

Baselines. We include MonoBERT (Nogueira and Cho, 2019), cross-SimLM (Wang et al., 2022), MonoT5 (Nogueira et al., 2020), RankT5 (Zhuang et al., 2023) as well as the state-of-the-art RankLlama model (Ma et al., 2023). Appx. B gives a detailed description of these methods .

Implementation Details. Our code is implemented in PyTorch (Paszke et al., 2019) using the Huggingface library (Wolf, 2019). The weights of the pre-trained language models are obtained from the Huggingface Hub[2]. Wherever applicable, we use Flash Attention 2 (Dao, 2024), gradient accumulation, and activation checkpointing. Note that the models are trained *without* parameter-efficient fine-tuning techniques such as LoRA (Hu et al., 2021) which is different from Ma et al. (2023). We also do not investigate alternative compression techniques for improved parameter efficiency, such as low-rank factorization (Gupta et al., 2024), and leave these avenues for future research.

We do not extensively tune hyperparameters; as discussed by prior works (Boytsov et al., 2022; Ma et al., 2023) fine-tuning of reranking models is less sensitive to hyperparameters. We found the vanilla AdamW optimizer along with learning rate warm-up with linear scheduler to work for all training runs. Refer to Appx. D for an overview of hyperparameters throughout the experiments. Out implementation as well as checkpoints will be made public to facilitate reproducibility.

For the autoregressive models, including Mamba models, we provide input with the following template: `document`: $\{d\}$; `query`: $\{q\}$; `[EOS]`. The linear layer then takes the last layer representation of the `[EOS]` token and outputs the relevance score:

$$f_\theta(q, d) = \text{Linear}\big(\text{model}(\text{input})[-1]\big) \quad (8)$$

For encoder-only and encoder-decoder models, we use a different template: `[CLS]` ; `query`: $\{q\}$;

[2] https://huggingface.co/models

document: $\{d\}$. The linear layer in this case uses the representation of the [CLS] token.

4.2 Passage Reranking Results

In-domain Evaluation. We show the in-domain passage reranking results in Table 2. First, note that our trained models are comparable to previously reported results. For example, we report that BERT-base achieves 38.5, 73.3, 73.1 on Dev, DL19, DL20 respectively, compared to MonoBERT (Nogueira and Cho, 2019)'s 37.2, 72.3 and 72.2. This suggests the correctness of our training setup.

Between transformer models of different architectures, we notice that both encoder-only and encoder-decoder models outperform decoder-only models (OPT-125M and 350M in our case), despite OPT being pre-trained with more tokens. We hypothesize the reason is that the bi-directional attention in encoders better capture the interaction between query and document tokens. But decoder-only models are easier to scale.

Between transformer and Mamba architectures, Mamba models are able to achieve strong performance. For example, despite being uni-directional, Mamba-2-370M achieves 38.6, 75.8, and 74.0 on three datasets compared to the best transformer-based model in that parameter range—BERT-large's 39.1, 76.4, and 72.4. The overall best transformer-based model—Llama-3.2-1B outperforms the Mamba models of similar size. However, note that Llama-3.2-1B is pre-trained on 15T tokens compared to Mamba model's 300B tokens. We conclude that Mamba models are competitive in the passage ranking task.

Among Mamba models, despite being trained on the same number of tokens, we notice overall that Mamba-2 achieves better performance than Mamba-1. A similar trend is shown in BEIR and document reranking results. In conclusion, Mamba-2 is a better SSM architecture compared to Mamba-1 for text reranking.

Out-of-domain Evaluation. We report part of BEIR results in § 4.2 and leave full results to Appx. E. Overall, Mamba models are able to achieve competitive performance compared to the transformer-based models of similar sizes. Specifically, Mamba-2-1.3B achieves 53.6 NDCG@10 averaged over 13 datasets compared to OPT-1.3B's 52.7. Compared to baselines, Mamba-based models are only outperformed by the much larger RankLlama — a 7B model based on 7B-sized retrieval model RepLlama. This reinforces our findings from in-domain evaluation, suggesting the efficacy of Mamba models in the passage reranking task.

One surprising observation is the underperformance of the Llama-3.2-1B model. This pre-trained model was not only trained on more tokens (15B) but was also trained on a much more diverse set of web documents. Ideally, a model pre-trained on a more diverse set of documents should perform better on out-of-domain evaluation sets, but we find that to not be the case with Llama-3.2-1B model.

4.3 Document Reranking

Next, we discuss the experimental setup and results for the document reranking task. The setup closely aligns with that used for passage reranking, with specific differences highlighted where applicable.

Datasets and Evaluation Metrics. We use the document ranking subset from the MSMARCO dataset containing 320K training instances. We use Pyserini's implementation of BM25 (Robertson et al., 1995)[3] as the first-stage document retriever and use top-100 documents to uniformly sample 7 hard negatives for each positive query-document pair. We train two model variants — FirstP and LongP — which truncate the input at 512 and 1,536 tokens respectively. Prior works (Boytsov et al., 2022; Ma et al., 2023) note that longer training lengths only yield marginal performance improvements. So we do not experiment with them.

For evaluation, we use the official development set (Dev) containing 5,193 queries and report MRR@100 for comparison. For the TREC DL19/DL20 (Craswell et al., 2020, 2021) evaluation set that includes 43/45 queries, we use the official NDCG@10 as our evaluation metric.

Other Details. Among the baselines, we include MonoT5 (Nogueira et al., 2020), RankT5 (Zhuang et al., 2023) along with RankLlama that is the current state-of-the-art model using 7 Billion parameters. We also report two baseline runs from Boytsov et al. (2022): BERT-base-FirstP and BERT-base-MaxP as a sanity check for our implementation. MaxP method first segments the long document into several shorter passages, then uses the maximum relevance of segmented passages as the relevance of the document. We train document reranking models for each of the pre-trained models highlighted in § 4.1. Note that as encoder-only models

[3] https://github.com/castorini/pyserini

Model	Size	Retriever	Dev MRR@10	DL19 NDCG@10	DL20 NDCG@10
MonoBERT (Nogueira and Cho, 2019)	110 M	BM25	37.2	72.3	72.2
cross-SimLM (Wang et al., 2022)	110 M	bi-SimLM	43.7	74.6	72.7
MonoT5 (Nogueira et al., 2020)	220 M	BM25	38.1	-	-
RankT5 (Zhuang et al., 2023)	335 M	GTR	42.2	-	-
RankLlama (Ma et al., 2023)	7 B	RepLlama	**44.9**†	**75.6**†	**77.4**†
BERT-base ^{E}	110 M	BGE	38.5	73.3	73.1
RoBERTa-base ^{E}	120 M	BGE	39.1	**75.4**	72.0
ELECTRA-base ^{E}	105 M	BGE	**39.8**	73.4	**74.1**
BART-base ^{ED}	130 M	BGE	37.8	74.7	70.2
OPT-125M ^{D}	125 M	BGE	35.2	70.6	69.2
Mamba-1-130M ^{D}	130 M	BGE	37.8	73.7	70.5
Mamba-2-130M ^{D}	130 M	BGE	37.0	73.8	70.8
BERT-large ^{E}	330 M	BGE	39.1	**76.4**	72.4
RoBERTa-large ^{E}	335 M	BGE	37.8	75.1	69.4
ELECTRA-large ^{E}	320 M	BGE	38.8	74.9	73.2
BART-large ^{ED}	385 M	BGE	**39.2**	74.6	72.2
OPT-350M ^{D}	350 M	BGE	36.3	72.1	68.9
Mamba-1-370M ^{D}	370 M	BGE	38.9	74.7	72.5
Mamba-2-370M ^{D}	370 M	BGE	38.6	75.8	**74.0**
Mamba-1-790M ^{D}	790 M	BGE	38.2	76.4	72.9
Mamba-2-780M ^{D}	780 M	BGE	**39.0**	**76.8**	**73.6**
OPT-1.3B ^{D}	1.3 B	BGE	38.9	74.2	73.7
Mamba-1-1.4B ^{D}	1.4 B	BGE	38.9	74.7	72.5
Mamba-2-1.3B ^{D}	1.3 B	BGE	38.6	75.8	74.0
Llama-3.2-1B ^{D}	1.3 B	BGE	**40.4**‡	**76.8**‡	**76.2**‡

Table 2: Results for passage reranking in-domain evaluation. We denote BGE-large-en-v1.5 as BGE for simplicity. We mark best results in each section bold; † indicates the overall best result and ‡ indicates the best result among our trained models. For the reranking threshold, RankLlama reranks top-100 results from RepLlama while other models reranks top-1000 results. Superscript E denotes encoder-only model, ED denotes encoder-decoder model and D denotes decoder-only model.

Dataset	BM25 -	MonoT5 220M	RankT5 335M	RankLlama 7B	ELECTRA 335M	BART 385M	Llama-3.2 1.3B	OPT 1.3B	Mamba-1 1.4B	Mamba-2 1.3B
Arguana	39.7	19.4	22.3	**56.0**†	14.6	18.0	32.7	**35.7**‡	33.1	**34.4**
Climate-FEVER	16.5	24.5	20.6	**28.0**†	18.2	20.9	22.6	**26.7**‡	22.6	**26.2**
DBPedia	31.8	41.9	43.5	**48.3**†	43.2	43.5	43.1	**45.8**‡	**45.8**‡	**45.8**‡
FEVER	65.1	80.1	83.5	**83.9**†	76.8	77.5	72.9	**83.0**‡	80.9	**81.9**
FiQA	23.6	41.3	41.6	**46.5**†	38.8	41.4	40.5	**44.3**‡	**43.3**‡	**43.3**‡
HotpotQA	63.3	69.5	71.3	**75.3**†	68.6	71.9	69.2	**74.9**	75.8	**76.3**‡
NFCorpus	32.2	**35.7**	32.6	30.3	33.5	34.9	**37.9**	32.8	38.8	**39.2**†‡
NQ	30.6	56.7	59.6	**66.3**†	49.2	51.0	48.2	**52.6**‡	50.8	**52.1**
Quora	78.9	82.3	82.2	**85.0**†	79.3	73.6	**84.9**‡	84.0	80.9	**83.9**
SCIDOCS	14.9	16.4	**18.2**	17.8	16.5	17.0	17.7	**17.8**	19.0	**19.6**†‡
SciFact	67.9	73.5	**74.9**	73.2	65.9	65.7	71.7	**72.7**	**77.4**†‡	76.8
TREC-COVID	59.5	77.6	75.2	**85.2**†	67.2	70.6	77.0	**81.6**	**83.0**‡	79.9
Touche-2020	44.2	27.7	**45.9**†	40.1	34.3	**34.9**	32.8	33.2	36.7	**37.7**‡
Average	43.7	49.7	51.7	**56.6**†	46.6	47.8	50.1	**52.7**	52.9	**53.6**‡

Table 3: Results for passage reranking out-of-domain evaluation. We show results of the largest encoder-only, encoder-decoder model and decoder-only models. Full results are referred to Appx. E. We mark best results in each section bold; † indicates the overall best result and ‡ indicates best result among our trained models.

have a fixed context length (ex: BERT with 512), we do not have LongP variants for them.

Model	Size	Dev MRR@100	DL19 NDCG@10	DL20 NDCG@10
BERT-base-FirstP	110 M	39.4	63.1	59.8
BERT-base-MaxP	110 M	39.2	64.8	61.5
MonoT5	3 B	41.1	-	-
RankLlama	7 B	**50.3**†	**67.7**	**67.4**†
FirstP models				
BERT-base ^{E}	110 M	**41.3**	65.8	61.5
RoBERTa-base ^{E}	125 M	39.4	65.5	59.3
ELECTRA-base ^{E}	105 M	39.0	66.3	62.3
BART-base ^{ED}	130 M	37.5	63.9	59.9
OPT-125M ^{D}	125 M	38.8	63.8	61.8
Mamba-1-130M ^{D}	130 M	40.9	66.5	**64.4**
Mamba-2-130M ^{D}	130 M	38.3	**66.7**	63.9
BERT-large ^{E}	330 M	40.1	65.9	61.4
RoBERTa-large ^{E}	355 M	**43.3**‡	66.8	64.2
ELECTRA-large ^{E}	335 M	40.3	**67.8**	**64.9**
BART-large ^{ED}	385 M	40.3	64.7	61.6
OPT-350M ^{D}	350 M	39.0	64.7	63.1
Mamba-1-370M ^{D}	370 M	42.5	67.8	63.9
Mamba-2-370M ^{D}	370 M	41.0	67.2	64.7
Mamba1-790M ^{D}	790 M	**42.0**	67.4	**64.9**
Mamba2-780M ^{D}	780 M	**42.0**	**68.7**	64.6
OPT-1B ^{D}	1.3 B	40.8	65.3	61.8
Llama-3.2-1B ^{D}	1.3 B	40.6	67.6	60.8
Mamba-1-1.3B ^{D}	1.3 B	OOM	OOM	OOM
Mamba-2-1.3B ^{D}	1.3 B	**42.1**	**68.3**	**64.6**
LongP models				
OPT-125M ^{D}	125 M	38.8	63.8	61.8
Mamba-1-130M ^{D}	130 M	**39.2**	66.0	63.0
Mamba-2-130M ^{D}	130 M	38.3	**67.3**	**63.6**
OPT-350M ^{D}	350 M	35.7	64.3	60.5
Mamba-1-370M ^{D}	370 M	39.3	**67.8**	64.3
Mamba-2-370M ^{D}	370 M	**41.4**	67.3	**65.1**
Mamba-1-790M ^{D}	790 M	41.3	68.0	64.9
Mamba-2-780M ^{D}	780 M	**42.2**	**70.0**†‡	**66.9**‡
OPT-1.3B ^{D}	1.3 B	**41.8**	68.0	**63.9**
Llama-3.2-1B ^{D}	1.3 B	40.9	**68.5**	63.5
Mamba-1-1.4B ^{D}	1.4 B	OOM	OOM	OOM
Mamba-2-1.3B ^{D}	1.3 B	OOM	OOM	OOM

Table 4: Results for document reranking. We mark the best result in each section bold; † marks the overall best result and ‡ marks best result among our trained models. For the reranking threshold, MonoT5 reranks top-1000 documents from the retriever while others rerank top-100 results. Superscripts E, ED and D are the same as Table 2. OOM denotes Out-Of-Memory Error.

4.4 Document Reranking Results

The task of document reranking necessitates using models that can process lengthy contexts. Although transformer models can accommodate long contexts through adjustments like improved positional embeddings (Su et al., 2024) or specialized training methods (Xiong et al., 2024; Dubey et al., 2024), it remains unclear whether Mamba-based state space models possess this capability. Our experiments in document reranking aim to address this gap. Our document reranking results are shown in Table 4.

We make two important observations. First, in terms of the task performance, Mamba-based rerankers are comparable to their Transformer-based counterparts for every parameter budget. Notably, among the sub-1 billion parameter models, the best model is the 780 million Mamba-2 model trained with 1536 context length. Second, while the Mamba-1 and Mamba-2 variants perform comparably for document reranking, we found that Mamba-2 models in general require less GPU memory. One such instance is the training run with 1.3B parameters for context length of 512 (FirstP settings)—Mamba-1 leads to OOM error but Mamba-2 does not. This echoes prior works' observation that Mamba-2 is more memory efficient during training compared to Mamba-1 (Dao and Gu, 2024; Yang et al., 2024, *inter alia*).

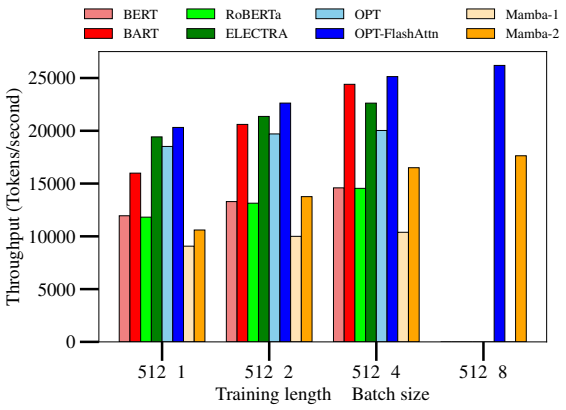


Figure 1: Training throughput comparison between models≈330M. For batch_size=8, all models except OPT-FlashAttn and Mamba-2 run out of memory with a 48 GB VRAM GPU.

4.5 Training Throughput and Inference Speed

To answer the **Efficiency RQ**, we evaluate the training throughput and inference speed of Mamba models and compare them to the transformer-based models. We perform this comparison with document ranking models as it involves a more challenging setting. All the numbers reported here are measured on a server with Intel Xeon Gold 6230 CPU @2.1GHz and a single Nvidia A40 GPU (48 GB VRAM).

We measure training throughput (#tokens/second) with 512 training length and the inference speed (#queries/second) on MS MARCO document ranking dataset with max input length of 512 and 1,536. The results for the training throughput with different training batch sizes are shown in Fig. 1. The average inference speed over the queries from DL19 eval set is

Model	Size	Max. Length	Queries. per Second (↑)
BERT-large	330 M	512	0.65
BART-large	385 M	512	0.65
OPT-350M	350 M	512	**0.69**
Mamba-1-370M	370 M	512	0.53
Mamba-2-370M	370 M	512	0.56
OPT-350M	370 M	1536	**0.45**
Mamba-1-370M	370 M	1536	0.40
Mamba-2-370M	370 M	1536	**0.45**
OPT-1.3B	1.3 B	512	0.29
Llama-3.2-1B	1.3 B	512	**0.33**
Mamba-1-1.4B	1.4 B	512	0.25
Mamba-2-1.3B	1.3 B	512	0.30
OPT-1.3B	1.3 B	1536	0.28
Llama-3.2-1B	1.3 B	1536	**0.31**
Mamba-1-1.4B	1.4 B	1536	0.24
Mamba-2-1.3B	1.3 B	1536	0.29

Table 5: Inference speed of different models. We use half precision and batch size 32 for all models.

shown in Table 5.

First, observe that Mamba-2 has a much higher training throughput than Mamba-1. Additionally, since the Mamba-2 models are more memory efficient during training compared to Mamba-1, we do not notice an Out-Of-Memory (OOM) errors with Mamba-2 — Mamba-1-370M does not train with batch size 8. The throughput of Mamba models is significantly worse than that of the transformer-based models. In other words, the Mamba-based models are much less efficient at training time.

As highlighted in prior research (Waleffe et al., 2024; Gu and Dao, 2023, *inter alia*), the true benefit of the Mamba models is realized with an improved inference speed. We do not observe this to be the case for the document reranking task (see Table 5). The main reason is that for inference, reranking only requires one single forward computation compared to multiple forward computations in autoregressive generation. We further discuss the deficiency of Mamba models in § 4.6.

4.6 Profiling Inference Computation

To better understand the inference performance of Mamba models, we use the PyTorch profiler[4] to analyze the execution time of Mamba models at the operator level, comparing it to Transformer-based models of similar sizes. As in § 4.5, we use the DL19 document ranking evaluation set, an input length of 512, an evaluation batch size of 32, and the same hardware configuration. The results are presented in Fig. 2.

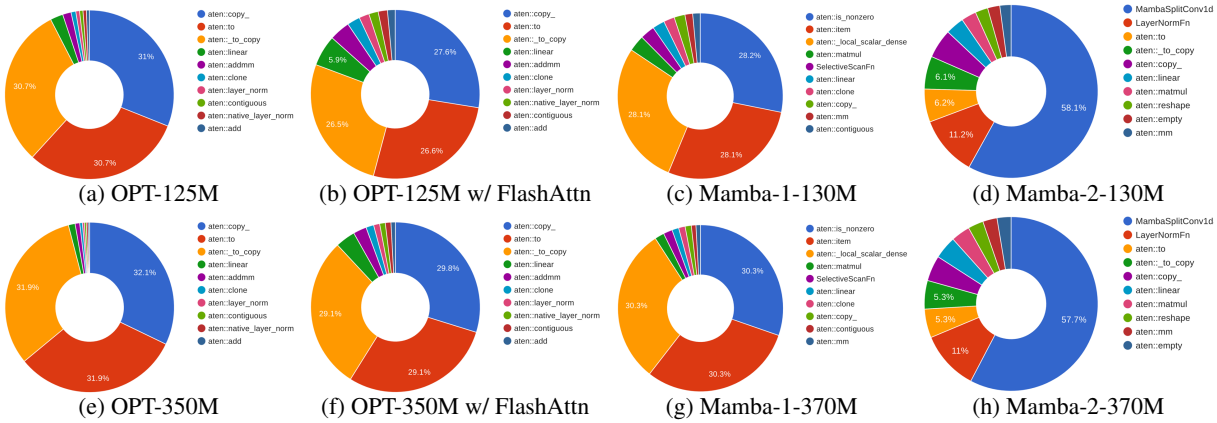
For Transformer-based models like OPT, I/O-related operators (e.g., `aten::copy_`, `aten::to`, `aten::_to_copy`, etc.) account for the majority of the execution time. Flash Attention (Dao, 2024) mitigates this by optimizing the I/O operations involved in attention computation, as seen in the reduced execution time for I/O-related operations in Figs. 2b and 2f. This optimization leads to a noticeable speedup in inference, highlighting the importance of improving I/O efficiency for Transformers.

In contrast, the total execution time of Mamba-1 is dominated by operators such as `aten::is_nonzero`, `aten::item`, and `aten::_local_scalar_dense`. The first operator, `aten::is_nonzero`, checks whether tensors contain any non-zero elements, while `aten::item` and `aten::_local_scalar_dense` are used to extract scalar values from tensors. This suggests that Mamba-1's architecture might suffer from computational inefficiencies due to an over-reliance on these scalar-extraction operations, which could be bottlenecking the performance. We hypothesize that these operations contribute to the model's overall computational deficiency, particularly in comparison to models that utilize more efficient tensor operations. Mamba-2 improves upon this by parameterizing the Mamba-2 block, allowing for more effective utilization of matrix multiplication. This change is reflected in the elimination of the aforementioned scalar-extraction operators, with the new operator `MambaSplitConv1D` now accounting for over half of the total execution time. Mamba-2's shift towards matrix multiplication suggests a more balanced computational load, although it still doesn't fully close the gap in terms of inference speed compared to Transformer models with Flash Attention. This empirical evidence points to the need for further architectural refinement to optimize performance and better leverage compute-optimized hardware.

5 Related Works

Text Ranking with Pre-trained Transformers. Fine-tuning pre-trained transformers has been the standard practice for text ranking tasks (Yates et al., 2021; Karpukhin et al., 2020, *inter alia*). Combining the query and the document as the input, the model predicts a scalar score indicating the

[4] https://pytorch.org/docs/stable/profiler.html



(a) OPT-125M (b) OPT-125M w/ FlashAttn (c) Mamba-1-130M (d) Mamba-2-130M

(e) OPT-350M (f) OPT-350M w/ FlashAttn (g) Mamba-1-370M (h) Mamba-2-370M

Figure 2: Inference profiling results for Mamba models versus OPT models of similar size.

relevance. Prior works have highlighted different aspects of training transformer-based text ranking models. Nogueira and Cho (2019); Nogueira et al. (2020); Dai and Callan (2019) are among the first efforts to showcase the effectiveness of fine-tuning pre-trained transformer-based language models. Gao et al. (2021) studied the retrieval-reranking pipeline and recommended training rerankers by sampling negatives from the results of first-stage retrievers. Li et al. (2023); Hofstätter et al. (2021) studied the effectiveness of chunking and pooling in long document ranking with shorter context transformer models. Boytsov et al. (2022) focused on benchmarking long context pre-trained transformers in long document ranking. Refer to (Yates et al., 2021; Xu et al., 2025) for detailed surveys.

Transformer Alternatives. Different works have explored transformer alternative model architectures for sequence modeling. For example, S4 (Gu et al., 2021b; Smith et al., 2022) demonstrate the effectiveness of structured state space models. Recent works (Peng et al., 2023; Yang et al., 2023, 2024; Qin et al., 2024, *inter alia*) have vastly improved the computational bottleneck of RNN-alike architectures and have shown comparable performance to modern transformer architectures at a moderate scale of comparison. We refer readers to these works for more details.

Within the IR community, works have explored the possibility of using state space models as retriever (Zhang et al., 2024) and reranker (Xu, 2024). This study extends prior works with more comprehensive experiments and points out new directions.

6 Conclusion and Future Work

This study investigates the suitability of Mamba architectures, a novel class of state space models, for text ranking. Our findings demonstrate that Mamba models, particularly Mamba-2, can achieve competitive performance compared to transformer-based models of comparable size, showcasing their potential as viable alternatives for sequence modeling in IR tasks. While Mamba architectures currently exhibit lower training and inference efficiency compared to transformers with flash attention, continuous advancements in model optimization and hardware acceleration have the potential to mitigate these limitations.

We picture two future directions of this work: the task direction and the model direction. From the task perspective, the efficacy of state space models, including Mamba should be further examined in other IR tasks (e.g., text retrieval). From the model perspective, hybrid models (Lieber et al., 2024; Lenz et al., 2024; Glorioso et al., 2024; Nvidia, 2025) have shown promise in certain NLP tasks. We believe the effectiveness of hybrid models should be thoroughly tested. Additionally, optimization for state space models is an interesting challenge that may offer substantial improvements.

Limitations and Potential Risks

This paper studies the efficacy of state space models in text ranking tasks. Our experiments are carried out by fine-tuning pre-trained language models, which differ in the pre-training corpus as well as pre-training FLOPs. Limited by hardware and budget, we are not able to carry out an apples-to-apples comparison with the exact same pre-training setup. We believe this leaves room for future direction.

This paper studies a well established task with publicly available datasets licensed for academic usage (see Appx. A). To the best of our knowledge this paper does not introduce potential risks.

Acknowledgements

This material is based upon work supported in part by NSF under grants 2007398, 2217154, 2318550, 2205418, and 2134223. This research is supported by the National Artificial Intelligence Research Resource (NAIRR) Pilot and the Delta advanced computing and data resource which is supported by the National Science Foundation (award NSF-OAC 2005572). Ashim Gupta is supported by the Bloomberg Data Science Ph.D. Fellowship. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsers.

References

Akari Asai, Jacqueline He, Rulin Shao, Weijia Shi, Amanpreet Singh, Joseph Chee Chang, Kyle Lo, Luca Soldaini, Sergey Feldman, Mike D'arcy, et al. 2024. Openscholar: Synthesizing scientific literature with retrieval-augmented lms. *arXiv preprint arXiv:2411.14199*.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Alexander Bondarenko, Maik Fröbe, Meriem Beloucif, Lukas Gienapp, Yamen Ajjour, Alexander Panchenko, Chris Biemann, Benno Stein, Henning Wachsmuth, Martin Potthast, et al. 2020. Overview of touché 2020: argument retrieval. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction: 11th International Conference of the CLEF Association, CLEF 2020, Thessaloniki, Greece, September 22–25, 2020, Proceedings 11*, pages 384–395. Springer.

Vera Boteva, Demian Gholipour, Artem Sokolov, and Stefan Riezler. 2016. A full-text learning to rank dataset for medical information retrieval. In *Advances in Information Retrieval: 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20–23, 2016. Proceedings 38*, pages 716–722. Springer.

Leonid Boytsov, Tianyi Lin, Fangwei Gao, Yutian Zhao, Jeffrey Huang, and Eric Nyberg. 2022. Understanding performance of long-document ranking models through comprehensive evaluation and leaderboarding. *arXiv preprint arXiv:2207.01262*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020. Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180*.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, and Daniel Campos. 2021. Overview of the trec 2020 deep learning track. corr abs/2102.07662 (2021). *arXiv preprint arXiv:2102.07662*.

Nick Craswell, Bhaskar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M Voorhees. 2020. Overview of the trec 2019 deep learning track. *arXiv preprint arXiv:2003.07820*.

Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 985–988.

Tri Dao. 2024. Flashattention-2: Faster attention with better parallelism and work partitioning. In *The Twelfth International Conference on Learning Representations*.

Tri Dao and Albert Gu. 2024. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 10041–10071. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Thomas Diggelmann, Jordan Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. Climate-fever: A dataset for verification of real-world climate claims. *arXiv preprint arXiv:2012.00614*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Luyu Gao and Jamie Callan. 2022. Unsupervised corpus aware language model pre-training for dense passage retrieval. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2843–2853.

Luyu Gao, Zhuyun Dai, and Jamie Callan. 2021. Rethink training of bert rerankers in multi-stage retrieval pipeline. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021,*

Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43, pages 280–286. Springer.

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

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

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

Albert Gu, Karan Goel, and Christopher Re. 2021a. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*.

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

Ankit Gupta, Albert Gu, and Jonathan Berant. 2022. Diagonal state spaces are as effective as structured state spaces. *Advances in Neural Information Processing Systems*, 35:22982–22994.

Ashim Gupta, Sina Mahdipour Saravani, P Sadayappan, and Vivek Srikumar. 2024. An empirical investigation of matrix factorization methods for pre-trained transformers. *arXiv preprint arXiv:2406.11307*.

Faegheh Hasibi, Fedor Nikolaev, Chenyan Xiong, Krisztian Balog, Svein Erik Bratsberg, Alexander Kotov, and Jamie Callan. 2017. Dbpedia-entity v2: a test collection for entity search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1265–1268.

Sebastian Hofstätter, Bhaskar Mitra, Hamed Zamani, Nick Craswell, and Allan Hanbury. 2021. Intra-document cascading: learning to select passages for neural document ranking. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1349–1358.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

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

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

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Canjia Li, Andrew Yates, Sean MacAvaney, Ben He, and Yingfei Sun. 2023. Parade: Passage representation aggregation fordocument reranking. *ACM Transactions on Information Systems*, 42(2):1–26.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xueguang Ma, Liang Wang, Nan Yang, Furu Wei, and Jimmy Lin. 2023. Fine-tuning llama for multi-stage text retrieval. *arXiv preprint arXiv:2310.08319*.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: financial opinion mining and question answering. In *Companion proceedings of the the web conference 2018*, pages 1941–1942.

Arvind Neelakantan, Tao Xu, Raul Puri, Alec Radford, Jesse Michael Han, Jerry Tworek, Qiming Yuan, Nikolas Tezak, Jong Wook Kim, Chris Hallacy, et al. 2022. Text and code embeddings by contrastive pre-training. *arXiv preprint arXiv:2201.10005*.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022.

Sentence-t5: Scalable sentence encoders from pretrained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pre-trained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Nvidia. 2025. Nemotron-h: A family of accurate, efficient hybrid mamba-transformer models.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, Xingjian Du, Matteo Grella, Kranthi Gv, Xuzheng He, Haowen Hou, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartłomiej Koptyra, Hayden Lau, Jiaju Lin, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Johan Wind, Stanisław Woźniak, Zhenyuan Zhang, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. 2023. RWKV: Reinventing RNNs for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14048–14077, Singapore. Association for Computational Linguistics.

Ronak Pradeep, Rodrigo Nogueira, and Jimmy Lin. 2021. The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models. *arXiv preprint arXiv:2101.05667*.

Zhen Qin, Songlin Yang, Weixuan Sun, Xuyang Shen, Dong Li, Weigao Sun, and Yiran Zhong. 2024. Hgrn2: Gated linear rnns with state expansion. *arXiv preprint arXiv:2404.07904*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at trec-3. *Nist Special Publication Sp*, 109:109.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*, volume 39. Cambridge University Press Cambridge.

Jimmy TH Smith, Andrew Warrington, and Scott Linderman. 2022. Simplified state space layers for sequence modeling. In *The Eleventh International Conference on Learning Representations*.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogeneous benchmark for zero-shot evaluation of information retrieval models.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

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

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

Ellen Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2021. Trec-covid: constructing a pandemic information retrieval test collection. In *ACM SIGIR Forum*, volume 54, pages 1–12. ACM New York, NY, USA.

Henning Wachsmuth, Shahbaz Syed, and Benno Stein. 2018. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 241–251.

David Wadden, Shanchuan Lin, Kyle Lo, Lucy Lu Wang, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or fiction: Verifying scientific claims. *arXiv preprint arXiv:2004.14974*.

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

Junxiong Wang, Daniele Paliotta, Avner May, Alexander Rush, and Tri Dao. 2024. The mamba in the

llama: Distilling and accelerating hybrid models. *Advances in Neural Information Processing Systems*, 37:62432–62457.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Simlm: Pre-training with representation bottleneck for dense passage retrieval. *arXiv preprint arXiv:2207.02578*.

T Wolf. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighof. 2023. C-pack: Packaged resources to advance general chinese embedding. *arXiv preprint arXiv:2309.07597*.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. 2024. Effective long-context scaling of foundation models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4643–4663.

Zhichao Xu. 2023. Context-aware decoding reduces hallucination in query-focused summarization. *arXiv preprint arXiv:2312.14335*.

Zhichao Xu. 2024. Rankmamba, benchmarking mamba's document ranking performance in the era of transformers. *arXiv preprint arXiv:2403.18276*.

Zhichao Xu and Daniel Cohen. 2023. A lightweight constrained generation alternative for query-focused summarization. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1745–1749.

Zhichao Xu, Daniel Cohen, Bei Wang, and Vivek Srikumar. 2024a. In-context example ordering guided by label distributions. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 2623–2640, Mexico City, Mexico. Association for Computational Linguistics.

Zhichao Xu, Ashim Gupta, Tao Li, Oliver Bentham, and Vivek Srikumar. 2024b. Beyond perplexity: Multi-dimensional safety evaluation of LLM compression. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 15359–15396, Miami, Florida, USA. Association for Computational Linguistics.

Zhichao Xu, Hemank Lamba, Qingyao Ai, Joel Tetreault, and Alex Jaimes. 2023. Counterfactual editing for search result explanation. *arXiv preprint arXiv:2301.10389*.

Zhichao Xu, Hemank Lamba, Qingyao Ai, Joel Tetreault, and Alex Jaimes. 2024c. Cfe2: Counterfactual editing for search result explanation. In *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval*, ICTIR '24, page 145–155, New York, NY, USA. Association for Computing Machinery.

Zhichao Xu, Fengran Mo, Zhiqi Huang, Crystina Zhang, Puxuan Yu, Bei Wang, Jimmy Lin, and Vivek Srikumar. 2025. A survey of model architectures in information retrieval. *arXiv preprint arXiv:2502.14822*.

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

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. 2024. Parallelizing linear transformers with the delta rule over sequence length. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. 2021. Pretrained transformers for text ranking: Bert and beyond. In *Proceedings of the 14th ACM International Conference on web search and data mining*, pages 1154–1156.

Hanqi Zhang, Chong Chen, Lang Mei, Qi Liu, and Jiaxin Mao. 2024. Mamba retriever: Utilizing mamba for effective and efficient dense retrieval. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 4268–4272.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Yue Zhang, ChengCheng Hu, Yuqi Liu, Hui Fang, and Jimmy Lin. 2021. Learning to rank in the age of Muppets: Effectiveness–efficiency tradeoffs in multi-stage ranking. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, pages 64–73, Virtual. Association for Computational Linguistics.

Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. 2024. Vision mamba: Efficient visual representation learning with bidirectional state space model. In *Forty-first International Conference on Machine Learning*.

Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. 2023. Rankt5: Fine-tuning t5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2308–2313.

A Dataset Artifacts and Licenses

Four of the datasets we used in experiments (NFCorpus (Boteva et al., 2016), FiQA-2018 (Maia et al., 2018), Quora[5], Climate-Fever (Diggelmann et al., 2020)) do not report the dataset license in the paper or a repository. For the rest of the datasets, we list their licenses below:

- MS MARCO (Bajaj et al., 2016): MIT License for non-commercial research purposes.
- ArguAna (Wachsmuth et al., 2018): CC BY 4.0 license.
- DBPedia (Hasibi et al., 2017): CC BY-SA 3.0 license.
- FEVER (Thorne et al., 2018): CC BY-SA 3.0 license.
- HotpotQA (Yang et al., 2018): CC BY-SA 4.0 license.
- NQ (Kwiatkowski et al., 2019): CC BY-SA 3.0 license.
- SCIDOCS (Cohan et al., 2020): GNU General Public License v3.0 license.
- SciFact (Wadden et al., 2020): CC BY-NC 2.0 license.
- TREC-COVID (Voorhees et al., 2021): "Dataset License Agreement".
- Touche-2020 (Bondarenko et al., 2020): CC BY 4.0 license.

B Additional Experiment Details

B.1 Complexity Analysis of State Space Model

We use the complexity analysis from (Dao and Gu, 2024). For details, refer to Section 6 of Dao and Gu (2024). Denote the sequence length as L and state size as N, which means size N per channel. We skip the #channel dimension (D) for ease of comparison. SSD structure used in Mamba-2 is able to achieve better training and inference complexity, as reflected in our experiments (Fig. 1 and Table 5).

[5] https://www.kaggle.com/c/quora-question-pairs

	Attention	SSM	SSD
State size	$O(L)$	$O(N)$	$O(N)$
Training FLOPs	$O(L^2N)$	$O(LN^2)$	$O(LN^2)$
Inference FLOPs	$O(LN)$	$O(N^2)$	$O(N^2)$
(Naive) memory	$O(L^2)$	$O(LN^2)$	$O(LN)$
Matrix multiplication	✓	✗	✓

Table 6: Complexity analysis between state space structure and attention.

B.2 Baselines

B.2.1 Sparse and Dense Retrieval Methods

For both document and passage retrieval, we include the classical BM25 baseline. For passage retrieval, bi-SimLM (Wang et al., 2022) is a competitive baseline that uses specialized pre-training with encoder-only transformer architecture for text retrieval task; GTR (Ni et al., 2022) is based on T5 (Raffel et al., 2020) architecture and is extensively fine-tuned for passage representations; BGE-large-en-v1.5 (Xiao et al., 2023) is based on BERT style encoder architecture and is fine-tuned with millions of synthetic query-passage pairs to achieve strong performance; OpenAI Ada2 (Neelakantan et al., 2022) is a proprietary embedding model developed by OpenAI; RepLlama (Ma et al., 2023) is based on Llama-2 language model (Touvron et al., 2023) and is fine-tuned on the training split of MS MARCO datasets. It achieves state-of-the-art performance on passage retrieval. For document retrieval, a common practice in literature is to segment long documents into several passages to fit into the 512 context length of BERT-style encoder-only transformer models. Each passage is scored individually and the relevance score of the document is an aggregation of individual passage's relevance scores. We include two such retrieval baselines: BM25-Q2D (Nogueira et al., 2019) uses the document expansion technique to enhance BM25's performance. CoCondenser-MaxP is based on CoCondenser technique (Gao and Callan, 2022) and uses max pooling for document relevance.

B.2.2 Reranking Methods

We include results from prior works as a comparison. For long document ranking, a common practice is to segment the long document into shorter passages and score them individually. For example, Dai and Callan (2019) referred to models only computing the relevance between query and the first document segment as FirstP, and methods that use the maximum relevance of passages within the document as the relevance of the document as

MaxP. We refer to document ranking models that based on long context language models as LongP following Boytsov et al. (2022).

For document ranking, we include BERT-base-FirstP and BERT-base-MaxP from Boytsov et al. (2022). We also include another MaxP baseline MonoT5 (Pradeep et al., 2021) and a state-of-the-art LongP model RankLlama (Ma et al., 2023).

For passage ranking, we include results of MonoBERT (Nogueira and Cho, 2019), cross-SimLM (Wang et al., 2022), MonoT5 (Nogueira et al., 2020) and more recent RankT5 (Zhuang et al., 2023) and RankLlama (Ma et al., 2023). An additional note is these ranking models are coupled with different first-stage retrievers and with different training strategies. We refer to RankT5 (Zhuang et al., 2023) for a comprehensive study of loss functions and training strategies involved in training ranking models.

C Retrieval Results

We show the passage retrieval results in Table 7 and document retrieval results in Table 8.

D Hyperparameter Setting

We show the hyperparameters in Table 9 and Table 10.

E Full BEIR Results

We refer the full results on BEIR to Table 11.

Model	Size	Embed. Dim.	Dev		DL19		DL20	
			MRR@10	Recall@1000	NDCG@10	Recall@1000	NDCG@10	Recall@1000
BM25	-	-	18.4	85.3	50.6	75.0	48.0	78.6
bi-SimLM	110M	768	39.1	98.6	69.8	-	69.2	-
GTR-base	110M	768	36.6	98.3	-	-	-	-
GTR-XXL	4.8B	768	38.8	99.0	-	-	-	-
BGE-large-en-v1.5	335M	1024	35.7	97.6	70.8	84.5	70.7	83.0
OpenAI Ada2	**?**	1536	34.4	98.6	70.4	**86.3**	67.6	**87.1**
RepLlama	7B	4096	**41.2**	**99.4**	**74.3**	-	**72.1**	-

Table 7: Passage retrieval performance of different retrieval models. We mark the best performance bold.

Model	Size	Seg. Y/N	Embed. Dim.	Dev		DL19		DL20	
				MRR@100	Recall@1000	NDCG@10	Recall@100	NDCG@10	Recall@100
BM25	-	N	-	27.7	93.6	52.3	38.5	50.6	58.6
BM25-Q2D	-	Y	-	32.7	95.5	59.7	**39.9**	58.5	**61.8**
CoCondenser-MaxP	110M	Y	768	42.5	93.9	64.8	-	**64.0**	-
RepLlama	7B	N	4096	**45.6**	**98.9**	**65.0**	-	63.2	-

Table 8: Document retrieval performance of different models. We mark the best performance bold.

Model	Size	Architecture	LR	Warmup	#Epochs	Global BZ	AMP	FlashAttn
***Encoder-only Models** (Bi-directional)*								
BERT-base	110M	Transformer	2e-5	10%	2	8	FP16	✗
RoBERTa-base	120M	Transformer	2e-5	10%	2	8	FP16	✗
ELECTRA-base	105M	Transformer	2e-5	10%	2	8	FP16	✗
BERT-large	330M	Transformer	1e-5	10%	2	8	FP16	✗
RoBERTa-large	335M	Transformer	1e-5	10%	2	8	FP16	✗
ELECTRA-large	320M	Transformer	1e-5	10%	2	8	FP16	✗
***Encoder-Decoder Models** (Bi-directional)*								
BART-base	130M	Transformer	2e-5	10%	2	8	FP16	✗
BART-large	385M	Transformer	1e-5	10%	2	8	FP16	✗
***Decoder-only Models** (Uni-directional)*								
OPT-125M	125M	Transformer	2e-5	10%	2	8	BF16	✓
Mamba-1-130M	130M	Mamba-1	2e-5	10%	2	8	BF16	✗
Mamba-2-130M	130M	Mamba-2	2e-5	10%	2	4	BF16	✗
OPT-350M	350M	Transformer	1e-5	10%	2	8	BF16	✓
Mamba-1-370M	370M	Mamba-1	1e-5	10%	2	4	BF16	✗
Mamba-2-370M	370M	Mamba-2	1e-5	10%	2	4	BF16	✗
Mamba-1-790M	790M	Mamba-1	1e-5	10%	1	4	BF16	✗
Mamba-2-780M	780M	Mamba-2	1e-5	10%	1	4	BF16	✗
OPT-1.3B	1.3B	Transformer	1e-5	10%	1	4	BF16	✓
Mamba-1-1.4B	1.4B	Mamba-1	1e-5	10%	1	4	BF16	✗
Mamba-2-1.3B	1.3B	Mamba-2	1e-5	10%	1	4	BF16	✗
Llama-3.2-1B	1.3B	Transformer++	1e-5	10%	1	4	BF16	✓

Table 9: Hyperparameters for passage reranking models. We use 10% of the total training steps for linear learning rate warmup. Global BZ denotes global batch size; AMP denotes automatic mixed precision, FlashAttn denotes whether Flash Attention 2 (Dao, 2024) is used.

Model	Size	Architecture	LR	Warmup	#Epochs	Global BZ	AMP	FlashAttn
Encoder-only Models *(Bi-directional)*								
BERT-base	110M	Transformer	2e-5	10%	2	8	FP16	✗
RoBERTa-base	120M	Transformer	2e-5	10%	2	8	FP16	✗
ELECTRA-base	105M	Transformer	2e-5	10%	2	8	FP16	✗
BERT-large	330M	Transformer	1e-5	10%	2	8	FP16	✗
RoBERTa-large	335M	Transformer	1e-5	10%	2	8	FP16	✗
ELECTRA-large	320M	Transformer	1e-5	10%	2	8	FP16	✗
Encoder-Decoder Models *(Bi-directional)*								
BART-base	130M	Transformer	2e-5	10%	2	8	FP16	✗
BART-large	385M	Transformer	1e-5	10%	2	8	FP16	✗
Decoder-only Models *(Uni-directional)*								
OPT-125M	125M	Transformer	2e-5	10%	2	8	BF16	✓
Mamba-1-130M	130M	Mamba-1	2e-5	10%	2	8	BF16	✗
Mamba-2-130M	130M	Mamba-2	2e-5	10%	2	4	BF16	✗
OPT-350M	350M	Transformer	1e-5	10%	2	8	BF16	✓
Mamba-1-370M	370M	Mamba-1	1e-5	10%	2	4	BF16	✗
Mamba-2-370M	370M	Mamba-2	1e-5	10%	2	4	BF16	✗
Mamba-1-790M	790M	Mamba-1	1e-5	10%	1	4	BF16	✗
Mamba-2-780M	780M	Mamba-2	1e-5	10%	1	4	BF16	✗
OPT-1.3B	1.3B	Transformer	1e-5	10%	1	4	BF16	✓
Mamba-1-1.4B	1.4B	Mamba-1	1e-5	10%	1	4	BF16	✗
Mamba-2-1.3B	1.3B	Mamba-2	1e-5	10%	1	4	BF16	✗
Llama-3.2-1B	1.3B	Transformer++	1e-5	10%	1	4	BF16	✓

Table 10: Hyperparameters for document reranking models. We use 10% of the total training steps for linear learning rate warmup. Global BZ denotes global batch size; AMP denotes automatic mixed precision, FlashAttn denotes whether Flash Attention 2 (Dao, 2024) is used. Note for LongP models, we additionally use gradient accumulation and/or activation checkpoint techniques to maintain a reasonably large global batch size. Mamba-1-1.4B gets OOM in FirstP setting; Mamba-1-1.4B and Mamba-2-1.3B get OOM in LongP setting with batch size 1 despite all optimization techniques at our hands.

Dataset	BM25 -	MonoT5 220M	RankT5 335M	RankLlama 7B	BERT-base 110M	BART-base 130M	RoBERTa-base 120M	ELECTRA-base 105M
Arguana	39.7	19.4	22.3	56.0	15.6	16.1	14.8	18.2
ClimateFever	16.5	24.5	20.6	28.0	16.9	16.6	17.8	20.3
DBPedia	31.8	41.9	43.5	48.3	38.5	42.5	42.1	42.1
FEVER	65.1	80.1	83.5	83.9	73.9	72.9	70.9	78.2
FiQA	23.6	41.3	41.6	46.5	34.6	38.4	36.4	40.1
HotpotQA	63.3	69.5	71.3	75.3	66.0	69.7	70.8	68.9
NFCorpus	32.2	35.7	32.6	30.3	29.3	32.7	26.1	29.9
NQ	30.6	56.7	59.6	66.3	45.2	48.6	49.6	50.1
Quora	78.9	82.3	82.2	85.0	75.8	75.3	74.8	79.3
SCIDOCS	14.9	16.4	18.2	17.8	16.1	15.8	15.4	17.1
SciFact	67.9	73.5	74.9	73.2	65.3	67.7	61.3	66.3
TREC-COVID	59.5	77.6	75.2	85.2	67.8	70.3	70.9	72.3
Touche-2020	44.2	27.7	45.9	40.1	30.7	33.2	30.1	33.3
Average	43.7	49.7	51.7	56.6	44.3	46.1	44.7	47.4

Dataset	OPT-125M 125M	Mamba-1-130M 130M	Mamba-2-130M 130M	BERT-large 330M	BART-large 385M	RoBERTa-large 335M	ELECTRA-large 320M	OPT-350M 350M
Arguana	10.1	32.8	33.8	19.5	18.0	15.4	14.6	21.0
ClimateFever	5.9	21.0	23.1	23.4	20.9	15.1	18.2	8.1
DBPedia	17.6	43.8	43.7	43.1	43.5	42.7	43.2	23.0
FEVER	9.5	76.6	76.3	79.5	77.5	71.9	76.8	19.8
FiQA	11.2	38.9	40.7	38.2	41.4	36.4	38.8	16.1
HotpotQA	31.7	72.2	72.8	70.2	71.9	66.8	68.6	48.1
NFCorpus	10.2	36.3	37.2	35.0	34.9	27.7	33.5	12.9
NQ	22.1	48.3	48.3	51.5	51.0	48.2	49.2	29.0
Quora	34.5	85.1	84.5	76.6	73.6	82.1	79.3	60.2
SCIDOCS	5.2	17.4	17.4	16.8	17.0	15.5	16.5	7.9
SciFact	9.7	72.2	73.0	68.8	65.7	55.4	65.9	28.6
TREC-COVID	51.9	75.9	79.0	68.0	70.6	70.8	67.2	57.3
Touche-2020	10.4	36.4	36.3	48.6	34.9	29.6	34.3	16.1
Average	17.7	50.5	51.2	49.2	47.8	44.4	46.6	26.8

Dataset	Mamba-1-370M 370M	Mamba-2-370M 370M	Mamba-1-790M 790M	Mamba-2-780M 780M	OPT-1.3B 1.3B	Llama-3.2-1B 1.3B	Mamba-1-1.4B 1.4B	Mamba-2-1.3B 1.3B
Arguana	33.3	34.8	34.4	33.7	35.7	32.7	33.1	34.4
ClimateFever	23.3	25.4	24.7	23.9	26.7	22.6	22.6	26.2
DBPedia	45.8	46.0	46.1	46.4	45.8	43.1	45.8	45.8
FEVER	76.5	79.1	81.8	80.4	83.0	72.9	80.9	81.9
FiQA	42.4	41.5	44.8	43.6	44.3	40.5	43.3	43.3
HotpotQA	75.7	75.0	75.6	76.2	74.9	69.2	75.8	76.3
NFCorpus	37.9	39.1	41.0	39.9	32.8	37.9	38.8	39.2
NQ	51.0	51.9	53.4	52.8	52.6	48.2	50.8	52.1
Quora	86.0	83.5	86.0	84.4	84.0	84.9	80.9	83.9
SCIDOCS	18.6	19.1	19.1	19.5	17.8	17.7	19.0	19.6
SciFact	75.2	76.0	77.7	77.1	72.7	71.7	77.4	76.8
TREC-COVID	82.7	81.2	82.7	85.1	81.6	77.0	83.0	79.9
Touche-2020	48.6	36.1	39.6	37.5	33.2	32.8	36.7	37.7
Average	53.6	53.0	54.4	53.9	52.7	50.1	52.9	53.6

Table 11: Full results for passage ranking out-of-domain evaluation.