# Long Context Modeling with Ranked Memory-Augmented Retrieval

Anonymous ACL submission

## Abstract

Effective long-term memory management is crucial for language models handling extended contexts. We introduce a novel framework that dynamically ranks memory entries based on relevance. Unlike previous works, our model introduces a novel relevance scoring and a pointwise re-ranking model for key-value embeddings, inspired by learning-to-rank techniques in information retrieval. Enhanced Ranked Memory Augmented Retrieval (**ERMAR**) achieves state-of-the-art results on standard benchmarks.

## 1 Introduction

Large Language Models (LLMs) face a fundamental limitation in processing long-context scenarios due to the quadratic complexity of attention mechanisms and increasing memory demands during generation (Vaswani, 2017; Tworkowski et al., 2024). Consider a scenario in an automated customer service system: *A customer reports an issue with their printer, referencing a setup process from a previous conversation that occurred two hours ago. After 50 messages of troubleshooting, the customer mentions that the same error from the beginning has resurfaced. Traditional LLMs, constrained by their context window, would struggle to access the crucial earlier context about the initial setup process, leading to inconsistent or incomplete responses, Figure 1.* It is well known that handling extended contexts remains a significant challenge, particularly in applications requiring document analysis and sustained dialogue interactions.

The recent MemLong (Liu et al., 2024) architecture stores and accesses historical context through basic chunk-level memory operations. The memory bank model is a large, non-trainable store of past context representations. Instead of recomputing representations for all past tokens every time, these representations are pre-computed and stored. Given the current context, MemLong retrieves relevant segments from the memory bank. It uses a dot product similarity search to find the memory entries most related to the current context. This allows the model to focus only on the most pertinent past information, rather than processing the entire history. However, its treatment of all key-value (K-V) pairs with equal weight, regardless of their contextual relevance, often leads to information overload and reduced retrieval precision. This limitation becomes particularly evident in scenarios requiring context management.

We have developed a novel model that addresses the aforementioned limitations by building upon Memlong (Liu et al., 2024), a publicly available baseline on GitHub[1]. Our **Enhanced Ranked Memory Augmented Retrieval** (**ERMAR**) model has a novel relevance scoring mechanism that fundamentally improves context retrieval and utilization for K-V embeddings. Unlike MemLong, ERMAR employs multiplication (Cao et al., 2007) to compute relevance scores, enabling a more nuanced and context-aware assessment of semantic alignment between queries and stored memory. ERMAR also incorporates a re-ranking mechanism that dynamically reorders K-V embeddings based on their relevance scores, ensuring that the most pertinent information is prioritized during retrieval. This re-ranking process, combined with an adaptive retrieval system that integrates historical usage patterns, allows ERMAR to capture subtle contextual relationships better and refine memory prioritization. As shown in Figure 1, ERMAR processes incoming queries and long-context conversations through a novel ranking architecture, employing K-V pairs ranking ($K_0$-$V_0$, $K_1$-$V_1$,...,$K_i$-$V_i$) and corresponding embeddings to perform semantic search and ranking of relevant historical information.

Our novel ERMAR model introduces three key

[1] https://github.com/Bui1dMySea/MemLong

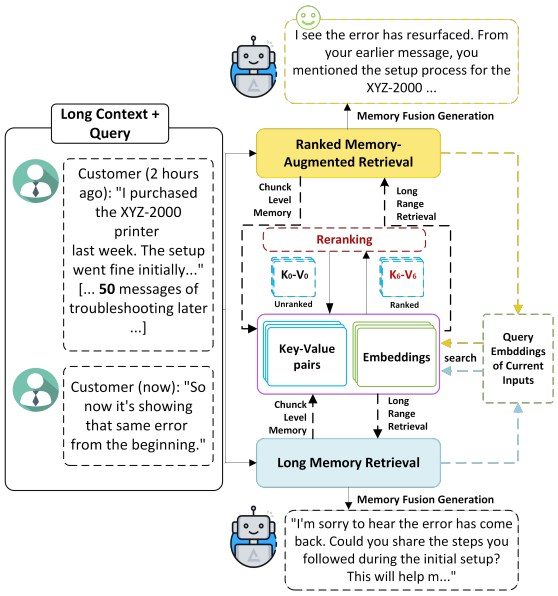


Figure 1: Our novel proposed ERMAR system. Note the difference from the MemLong architecture where we have introduced a novel *Reranking* model.

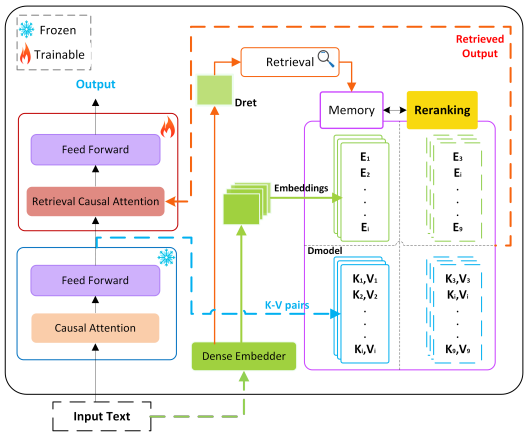


Figure 2: The architecture diagram for ERMAR.

improvements: (i) A semantic similarity metric to measure contextual alignment between query embeddings and key-value pairs; (ii) A weighted scoring function that considers content similarity and contextual relevance; and (iii) Integration of historical usage patterns to refine relevance assessment.

## 2 Related Work

Existing memory-augmented architectures such as RetroMAE (Xiao et al., 2022) have demonstrated promising results but often struggle with semantic coherence and retrieval efficiency. Similarly, sparse attention mechanisms reduce computational complexity but frequently sacrifice model capability (Beltagy et al., 2020). Memorizing Transformers (Wu et al., 2022) developed dedicated memory tokens. Recent advances have focused on memory management through hierarchical cache structures (Wu et al., 2022) and context-aware retention strategies (Borgeaud et al., 2022). However, these systems often struggle with distribution shifts during training and effective information retrieval. While these methods offer partial solutions, they often face challenges in maintaining consistent performance across varying context lengths and ensuring reliable information retrieval.

Architectural modifications have focused primarily on improving position representation and attention mechanisms. Position encoding adaptations such as RoPE (Su et al., 2024) and ALiBi (Press et al., 2022) have enhanced models' ability to handle longer sequences by providing more robust position information. YARN (Peng et al., 2023) further advanced this approach through dynamic position embeddings, demonstrating reliable generalization up to 128k tokens. Complementing these developments, sparse attention mechanisms such as Longformer (Beltagy et al., 2020) and BigBird (Zaheer et al., 2020) have reduced computational complexity while maintaining model capabilities through selective attention patterns.

## 3 Our Novel ERMAR Model

Figure 2 illustrates our ERMAR framework and Figure 1 presents the contextual ranking mechanism of key, value pairs components that enable effective retrieval. ERMAR maintains consistency through frozen lower layers and selective parameter updating. ERMAR: (i) stores important information from earlier parts of the text; (ii) assigns relevance scores to stored information based on its importance to the current context, and (iii) retrieves only the most relevant historical information when needed. Our relevance scoring is analogous to attention, allowing the model to focus on important parts of the memory. There is also a "loose" pointwise connection because the primary objective is sequence likelihood.

Let $\mathcal{V}$ be a finite vocabulary, and $\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{V}^n$ a token sequence with preceding context $\mathbf{x}_{<i}$. The embedding function $\mathcal{E} : \mathcal{V}^* \to \mathbb{R}^{d_{\text{ret}}}$ maps sequences to retrieval space. We introduce a memory function $\mathcal{M}$ augmented with a relevance scoring mechanism

$$\mathcal{M} : \underbrace{\mathbb{R}^{d_{\text{model}}}}_{\text{keys}} \times \underbrace{\mathbb{R}^{d_{\text{model}}}}_{\text{values}} \times \underbrace{\mathbb{R}^{d_{\text{ret}}}}_{\text{embeddings}} \to \mathcal{S}.$$

**Relevance Score:** Given a query embedding $\mathbf{q} \in \mathbb{R}^{d_{\text{ret}}}$ and a matrix of key embeddings $\mathbf{K} = [k_1, \ldots, k_m] \in \mathbb{R}^{m \times d_{\text{ret}}}$ (where each row $k_i \in \mathbb{R}^{d_{\text{ret}}}$ corresponds to a key embedding), the relevance score is:

$$\alpha(\mathbf{q}, \mathbf{K}) = \text{softmax}\left(\frac{\mathbf{q}\mathbf{K}^\top}{\sqrt{d_{\text{ret}}}}\right) \tag{1}$$

Here, $\sqrt{d_{\text{ret}}}$ normalizes the similarity scores to prevent excessively large values. The relevance score $\alpha(\mathbf{q}, \mathbf{K})$ can be interpreted as a probability distribution over the keys, where each entry $\alpha_i$ represents the relative importance (or attention weight) of the $i$-th key to the query $\mathbf{q}$. This score is used to rank memory entries based on their importance to the current query.

**Ranked Key-Value Pairs:** Each embedding $\mathbf{e}$ maintains a ranked set of key-value pairs: $\mathcal{R}_{\text{ranked}}(\mathbf{e}) = \{(K_j, V_j, s_j)\}_{j=1}^{m}$,, where $s_j = \alpha(\mathbf{e}, K_j)$ is the relevance score between the embedding $\mathbf{e}$ and the key $K_j$.

ERMAR objection function is formulated as:

Given a sequence $\mathbf{x}$, maximize: $\mathcal{L}(\theta) = \sum_{i=1}^{n} p_\theta(x_i \mid \mathcal{R}_{\text{RSAR}}(t_i, s), \mathbf{x}_{<i})$, subject to: $s_i = \mathcal{M}(K_{1:i-1}, V_{1:i-1}; \mathcal{E}(t_{1:i-1}))$, $t_i = \text{text}(c_{\lceil i/\tau \rceil})$, $\alpha_i = \alpha(\mathcal{E}(t_i), K_{1:i-1})$,

where $\alpha_i$ guides key-value pair selection, and $p_\theta$ represents the model's probability distribution.

For new content $(K_{\text{n}}, V_{\text{n}})$, update the memory state as: $s_{i+1} = \begin{cases} \mathcal{M}(K_{\text{n}}, V_{\text{n}}; \mathcal{E}(t_{\text{n}})) & \text{if } |s_i| < \text{capacity}, \\ M_{\text{u}}(s_i, K_{\text{n}}, V_{\text{n}}, \alpha_{\text{n}}) & \text{otherwise}, \end{cases}$ where $M_{\text{u}}$ prunes the least relevant entries based on historical scores, specifically by ranking the scores and pruning those with the lowest values relative to the current context.

We now develop the **Relevance Scoring with Adaptive Retrieval (RSAR)**. This approach dynamically ranks memory entries based on their importance to the current query, significantly improving the retrieval process. The relevance score, $\alpha(\mathbf{q}, \mathbf{K})$, as defined in equation 1, is used to rank memory entries.

RSAR enhances the memory module by introducing ranked key-value entries, represented as $(K_j, V_j, s_j)$, where $s_j$ denotes the relevance score for each entry. These scores enable the system to prioritize the most relevant information during retrieval while maintaining computational efficiency. To ensure optimal memory utilization, a pruning strategy is applied to remove less relevant entries. Specifically, entries with scores below a predefined threshold are discarded, preserving only the most critical context. The enhanced retrieval mechanism is expressed as: $R_{\text{RSAR}}(t_q, s) = \text{TopK}\{\text{sim}(E(t_q), e) \cdot \max_j s_j \mid e \in s\}$, where $E(t_q)$ represents the encoded query, and the operation identifies the top-$K$ relevant entries based on their scores. This mechanism efficiently retrieves the most relevant information, even for extended contexts.

### 3.1 Experimental Setup

We fine-tuned ERMAR on the *SlimPajama* dataset (Fu et al., 2024), a high-quality, deduplicated corpus designed for long-context tasks. It contains 84.7K training rows, making it a compact yet effective resource for pre-training and fine-tuning. Performance was measured across context lengths from 1024 to 32768 tokens, using perplexity on the last 2048 tokens (Yen et al., 2024).

We fine-tuned OpenLLaMA-3B, a pre-trained LLM with rotational position encoding (Su et al., 2024), using LoRA (Hu et al., 2021) for efficiency. The model has $L = 26$ layers, $H = 32$ attention heads, and $d = 100$ dimensions. The 13th layer serves as the memory layer, while layers [14, 18, 22, 26] are used for retrieval augmentation.

ERMAR was evaluated against state-of-the-art 7B and 3B parameter models. The 7B models include LLaMA-2-7B (Touvron et al., 2023b) as a transformer baseline, LongLoRA-7B-32k (Chen et al., 2023) with sparse attention for 32k-token contexts, and YARN-128k-7B (Peng et al., 2023) with dynamic position embeddings for 128k tokens. The 3B models include OpenLLaMA-3B (Touvron et al., 2023a), LongLLaMA-3B (Tworkowski et al., 2024) (evaluated in two retrieval configurations), and Phi3-128k (Abdin et al., 2024), which performs well across varying context lengths. This diverse benchmark suite ensures a robust evaluation of ERMAR's long-context capabilities.

### 3.2 Results and Discussion

#### 3.2.1 Long-Context Language Modeling

Following the experimental strategy adopted in (Liu et al., 2024), Table 1 presents the mean perplexity scores of our model across different sequence lengths and datasets, demonstrating its effectiveness in long-context modeling. Evaluation was performed on test splits of three datasets: *WikiText-103* (Merity et al., 2016) (4,358 rows), *PG-19* (Rae et al., 2019) (100 rows), and *Proof-Pile* (Azerbayev et al., 2023) (46.3k rows).

| Model | PG19 | | | | Proof-pile | | | | Wikitext-103 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1k | 2k | 4k | 16k | 1k | 2k | 4k | 16k | 1k | 2k | 4k | 16k |
| 7B Model | | | | | | | | | | | | |
| YARN-128k-7b | 7.22 | 7.47 | 7.17 | - | 3.03 | 3.29 | 2.98 | - | 5.71 | 6.11 | 5.71 | - |
| LongLoRA-7B-32k | 9.76 | 9.71 | 10.37 | 7.62 | 3.68 | 3.35 | 3.23 | 2.60 | 7.99 | 7.83 | 8.39 | 5.47 |
| LLaMA-2-7B | 10.82 | 10.06 | 8.92 | - | 3.24 | 3.40 | 2.72 | - | 10.82 | 6.49 | 5.66 | - |
| 3B Model | | | | | | | | | | | | |
| Phi3-128k | 11.31 | 9.90 | **9.66** | -/9.65 | 4.25 | 3.11 | 2.77 | -/3.08 | 7.54 | 7.22 | 7.01 | -/7.20 |
| OpenLLaMA-3B | 11.60 | 9.77 | $> 10^3$ | - | **2.96** | **2.70** | $> 10^3$ | - | 10.57 | 8.08 | $> 10^3$ | - |
| LongLLaMA-3B* | 10.59 | 10.02 | $> 10^3$ | - | 3.55 | 3.15 | $> 10^3$ | - | 8.88 | 8.07 | $> 10^3$ | - |
| LongLLaMA-3B† | 10.59 | 10.25 | 9.87 | - | 3.55 | 3.22 | **2.94** | - | 10.69 | 8.33 | 7.84 | - |
| MemLong-3B* | 10.66 | 10.09 | $> 10^3$ | - | 3.58 | 3.18 | $> 10^3$ | - | 8.72 | 7.93 | $> 10^3$ | - |
| w/ 4K MemLong | 10.54 | 9.95 | 9.89 | **9.64** | 3.53 | 3.16 | 3.15 | **2.99** | 8.53 | 7.92 | 7.87 | 7.99 |
| w/ 4K **ERMAR** | **10.32** | **9.75** | 9.78 | 9.81 | 3.24 | 2.98 | 3.03 | 3.18 | **7.92** | **7.41** | **7.34** | **7.08** |

Table 1: Perplexity comparison of 7B and 3B models across PG19, Proof-pile, and WikiText-103, using a sliding window evaluation. "-" denotes Out of Memory (OOM) errors, and "x/y" indicates results from single/dual GPU setups. Memory-augmented models are tested with varying capacities. All runs use a single 3090 24GB GPU.

Among 7B models, YARN-128k-7B excels in shorter contexts, while LongLoRA-7B-32k scales effectively to 16k-token sequences, though with some performance degradation. This highlights the trade-off between performance and scalability, guiding model selection based on use-case needs.

The 3B models demonstrate ERMAR's significant advantages in long-context tasks. While OpenLLaMA-3B struggles beyond 4k tokens, and Phi3-128k shows more consistent performance, ERMAR sets new performance benchmarks. ERMAR outperforms MemLong and larger 7B models in several configurations, including achieving a remarkable 2.98 perplexity on Proof-pile, surpassing MemLong's 3.16. Despite slight underperformance in specific Proof-pile configurations, ERMAR's enhanced memory retrieval mechanism proves more effective overall. Additionally, ERMAR maintains stable performance up to 16k tokens, with only a 3.2% degradation in perplexity from 4k to 16k tokens, demonstrating its superior scalability in long-context modelling.

### 3.2.2 In-Context Learning Performance

The results in Table 2 show ERMAR's strong performance across five natural language understanding tasks in both 4-shot and 20-shot settings.

In the 4-shot setting, ERMAR achieves state-of-the-art results across all tasks, outperforming OpenLLaMA and other memory-augmented models. It excels even in challenging tasks like SST-5 and MPQA, maintaining high performance with limited examples. Its stability across different memory configurations highlights its robustness in low-resource scenarios.

ERMAR continues to excel in the 20-shot scenario, achieving top results in tasks like MPQA and Subj, and setting a new benchmark for SST-5. While it lags behind MemLong in a few tasks, ERMAR outperforms it overall, showcasing its scalability with increased examples.

ERMAR consistently performs well across varying context lengths, effectively leveraging memory augmentation. Its ability to scale with more examples and handle both short and long-range dependencies makes it a strong candidate for general-purpose language modelling, advancing the state-of-the-art in language understanding tasks.

| Model | In-C ,In-M | SST-2 ACC↑ | MR ACC↑ | Subj ACC↑ | SST-5 ACC↑ | MPQA ACC↑ | Avg. |
|---|---|---|---|---|---|---|---|
| OpenLLaMA | 4,N/A | 90.7 | 84.0 | 58.2 | 41.0 | 70.5 | 68.9 |
| w./ Rag | 4,4 | 90.9 | 90.5 | 61.6 | 39.2 | 63.2 | 69.1 |
| LongLLaMA | 4,4 | 90.4 | 83.9 | 64.3 | 40.0 | 64.2 | 68.6 |
| MemLong | 4,4 | 91.5 | 84.5 | 61.5 | 41.4 | 70.2 | 69.8 |
| ERMAR | 4,4 | **93.6** | **90.8** | **65.3** | **45.8** | **85.2** | **76.14** |
| LongLLaMA | 4,18 | 91.4 | 87.1 | 59.1 | 41.0 | 64.5 | 68.7 |
| MemLong | 4,18 | 91.0 | 89.6 | 61.7 | 43.5 | 69.4 | 71.0 |
| ERMAR | 4,18 | **93.6** | **90.8** | **65.3** | **45.9** | **85.2** | **76.16** |
| OpenLLaMA | 20,N/A | 93.6 | 91.2 | 55.4 | 38.2 | 66.4 | 69.0 |
| w./ Rag | 20,18 | 92.2 | 91.3 | 75.8 | 39.8 | 57.6 | 71.3 |
| LongLLaMA | 20,18 | 94.1 | 90.8 | 64.2 | 41.4 | 72.1 | 72.7 |
| MemLong | 20,18 | 93.5 | **93.8** | 65.8 | 43.3 | 70.6 | 73.4 |
| ERMAR | 20,18 | **94.7** | 91.7 | **82.8** | **47** | **86.5** | **80.54** |

Table 2: 4-shot and 20-shot ICL accuracy [%] on 5 NLU tasks (SST-2, MR, Subj, SST-5, MPQA). We compare OpenLLaMA, LongLLaMA, MemLong, and ERMAR. **Note:** In-C = In-Context, In-M = In-Memory.

## 4 Conclusion

We presented a novel ERMAR framework that enhances long-context modelling through relevance scoring and adaptive memory retrieval. ERMAR outperforms baseline models, including OpenLLaMA, LongLLaMA, and MemLong, achieving superior perplexity and in-context learning performance on WikiText-103, PG19, and Proof-pile. Future work will focus on optimizing ERMAR for specialized datasets and expanding its applicability to complex reasoning tasks.

## 5 Limitations

While ERMAR improves retrieval efficiency and context retention, it has limitations. Its reliance on ranked memory structures increases computational overhead compared to standard LLMs, particularly for large-scale retrieval. Additionally, performance variations across different task domains indicate a need for further tuning. The framework's effectiveness in real-world, noisy environments also requires further validation.

## References


Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Zhangir Azerbayev, Edward Ayers, and Bartosz Piotrowski. 2023. Proofpile: A pre-training dataset of mathematical texts.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*.

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. 2024. Data engineering for scaling language models to 128k context. *arXiv preprint arXiv:2402.10171*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Weijie Liu, Zecheng Tang, Juntao Li, Kehai Chen, and Min Zhang. 2024. Memlong: Memory-augmented retrieval for long text modeling. *arXiv preprint arXiv:2408.16967*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*.

Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. *Preprint*, arXiv:2108.12409.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. 2019. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*.

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

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

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

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

A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Yuhuai Wu, Markus N Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. Memorizing transformers. *arXiv preprint arXiv:2203.08913*.

Shitao Xiao, Zheng Liu, Yingxia Shao, and Zhao Cao. 2022. Retromae: Pre-training retrieval-oriented language models via masked auto-encoder. *arXiv preprint arXiv:2205.12035*.

Howard Yen, Tianyu Gao, and Danqi Chen. 2024. Long-context language modeling with parallel context encoding. *arXiv preprint arXiv:2402.16617*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297.