Beyond Attention: Breaking the Limits of Transformer Context Length with Recurrent Memory

Anonymous Author(s)
Affiliation
Address
email

Abstract

A major limitation for the broader scope of problems solvable by transformers is the quadratic scaling of computational complexity with input size. In this study, we investigate the recurrent memory augmentation of pre-trained transformer models to extend input context length while linearly scaling compute. Our approach demonstrates the capability to store information in memory for sequences of up to an unprecedented two million tokens while maintaining high retrieval accuracy. Experiments with language modeling tasks show perplexity improvement as the number of processed input segments increases. These results underscore the effectiveness of our method, which has significant potential to enhance long-term dependency handling in natural language understanding and generation tasks, as well as enable large-scale context processing for memory-intensive applications.

1 Introduction

Transformer-based models show their effectiveness across multiple domains and tasks. The self-attention allows to combine information from all sequence elements into context-aware representations. However, global and local information has to be stored mostly in the same element-wise representations. Moreover, the length of an input sequence is limited by quadratic computational complexity of self-attention. In this work, we propose and study a memory-augmented segment-level recurrent Transformer (Recurrent Memory Transformer). Memory allows to store and process local and global information as well as to pass information between segments of the long sequence with the help of recurrence. We implement a memory mechanism with no changes to Transformer model by adding special memory tokens to the input or output sequence. Then Transformer is trained to control both memory operations and sequence representations processing.

This study we show that by using simple token-based memory mechanism introduced in [Bulatov et al., 2022] can be combined with pretrained transformer models like BERT [Devlin et al., 2019] and GPT-2 [Radford et al., 2019] with full attention and full precision operations.

Contributions

1. We enhance both encoder-only and decoder-only pre-trained Transformer language models by incorporating token-based memory storage and segment-level recurrence with recurrent memory (RMT).

2. We demonstrate that language models pre-trained on much shorter lengths can be trained with RMT approach to tackle tasks on sequences many times longer than its originally designed input length.

3. We discovered the trained RMT's capacity to successfully extrapolate to tasks of varying lengths, including those exceeding 1 million tokens with linear scaling of computations required.

4. Through attention pattern analysis, we found the operations RMT employs with memory, enabling its success in handling exceptionally long sequences.

2 Related work

Our work revolves around the concept of memory in neural architectures. Memory has been a recurrent theme in neural network research, dating back to early works [McCulloch and Pitts, 1943, Stephen, 1956] and significantly advancing in the 1990s with the introduction of the *Backpropagation Through Time* learning algorithm [Werbos, 1990] and *Long-Short Term Memory* (LSTM) neural architecture [Hochreiter and Schmidhuber, 1997]. Contemporary memory-augmented neural networks (MANNs) typically utilize some form of recurrent external memory separate from the model's parameters. *Neural Turing Machines* (NTMs) [Graves et al., 2014] and *Memory Networks* [Weston et al., 2015] are equipped with storage for vector representations accessible through an attention mechanism. Memory Networks [Weston et al., 2015, Sukhbaatar et al., 2015] were designed to enable reasoning through sequential attention over memory content.

NTMs, followed by *Differentiable Neural Computer* (DNC) [Graves et al., 2016] and *Sparse DNC* [Rae et al., 2016], are implemented as recurrent neural networks capable of writing to memory storage over time. All these models are differentiable and trainable via backpropagation through time (BPTT). Parallel research lines extend recurrent neural networks, such as LSTM, with data structures like stacks, lists, or queues [Joulin and Mikolov, 2015, Grefenstette et al., 2015]. MANN architectures with more advanced addressing mechanisms, such as address-content separation and multi-step addressing, have been proposed in [Gulcehre et al., 2016, 2017, Meng and Rumshisky, 2018]. The Global Context Layer model [Meng and Rumshisky, 2018] employs address-content separation to address the challenge of training content-based addressing in canonical NTMs.

Memory is often combined with Transformers in a recurrent approach. Long inputs are divided into smaller segments, processed sequentially with memory to access information from past segments. Transformer-XL [Dai et al., 2019] preserves previous hidden states for reuse in subsequent segments, while Compressive Transformer [Rae et al., 2020] adds new compressed memory. Ernie-Doc [Ding et al., 2021] enhances contextual information flow by employing same-layer recurrence instead of attending to previous layer outputs of preceding segments. Memformer [Wu et al., 2022a] introduces a dedicated memory module to store previous hidden states in summarized representations. Using a similar approach to Memformer, MART [Lei et al., 2020] and Block-Recurrent Transformer [Hutchins et al., 2022] adopt memory update rules analogous to LSTM [Hochreiter and Schmidhuber, 1997] and GRU [Cho et al., 2014]. FeedBack Transformer [Fan et al., 2020] implements full recurrence beyond the segment level and merges low and high layers representations into a memory state.

A drawback of most existing recurrent methods is the need for architectural modifications that complicate their application to various pre-trained models. In contrast, the Recurrent Memory Transformer can be built upon any model that uses a common supported interface.

Some approaches redesign the self-attention mechanism to reduce computational complexity while minimizing input coverage loss. *Star-Transformer* [Guo et al., 2019], *Longformer* [Beltagy et al., 2020], *GMAT* [Gupta and Berant, 2020], *Extended Transformer Construction* (ETC) [Ainslie et al., 2020], and *Big Bird* [Zaheer et al., 2020] limit attention distance and employ techniques such as global representations to preserve long-range dependencies. *Memory Transformer* [Burtsev et al., 2020] introduces memory by extending the unchanged model input with special memory tokens.

A common constraint of these methods is that memory requirements grow with input size during both training and inference, inevitably limiting input scaling due to hardware constraints. The longest Longformer, Big Bird, and Long T5 [Guo et al., 2022] models reported in their respective papers have a maximum length of less than 33,000 tokens. CoLT5 [Ainslie et al., 2023] can handle up to 64,000 tokens before running out of memory, and Memorizing Transformers [Wu et al., 2022b] and Unlimiformer [Bertsch et al., 2023] further extend memory through k-NN.

3 Recurrent Memory Transformer

Starting from the initial Recurrent Memory Transformer [Bulatov et al., 2022] (RMT), we adapted it for a plug-and-play approach as a wrapper for a range of popular Transformers.

This adaptation augments its backbone with memory, composed of m real-valued trainable vectors (Figure 1). The lengthy input is divided into segments, and memory vectors are prepended to the first segment embeddings and processed alongside the segment tokens. For encoder-only models like BERT, memory is added only once at the beginning of the segment, unlike [Bulatov et al., 2022], where decoder-only models separate memory into read and write sections. For the time step τ and segment H_τ^0, the recurrent step is performed as follows:

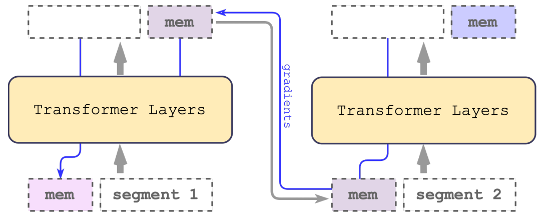

Figure 1: **Recurrent memory mechanism.** Memory is passed to Transformer along input sequence embeddings, and memory output is passed to the next segment. During training gradients flow from the current segment through memory to the previous segment.

$$\tilde{H}_\tau^0 = [H_\tau^{mem} \circ H_\tau^0], \bar{H}_\tau^N = \text{Transformer}(\tilde{H}_\tau^0), [\bar{H}_\tau^{mem} \circ H_\tau^N] := \bar{H}_\tau^N,$$

here N is a number of Transformer layers.

After the forward pass, $\bar{H}_\tau^{mem}$ contains updated memory tokens for the segment τ.

Segments of the input sequence are processed sequentially. To enable the recurrent connection, we pass the outputs of the memory tokens from the current segment to the input of the next one:

$$H_{\tau+1}^{mem} := \bar{H}_\tau^{mem}, \tilde{H}_{\tau+1}^0 = [H_{\tau+1}^{mem} \circ H_{\tau+1}^0].$$

Both memory and recurrence in the RMT are based only on global memory tokens. This allows the backbone Transformer to remain unchanged, making the RMT memory augmentation compatible with any model from the Transformer family.

3.1 Computational efficiency

We can estimate the required FLOPs for RMT and Transformer models of different sizes and sequence lengths. We took configurations (vocabulary size, number of layers, hidden size, intermediate hidden size, and number of attention heads) for the OPT model family [Zhang et al., 2022] and computed the number of FLOPs for the forward pass following [Hoffmann et al., 2022]. We also modified FLOP estimates to account for the effect of RMT recurrence.

Figure 2 shows that RMT scales linearly for any model size if the segment length is fixed. We achieve linear scaling by dividing an input sequence into segments and computing the full attention matrix

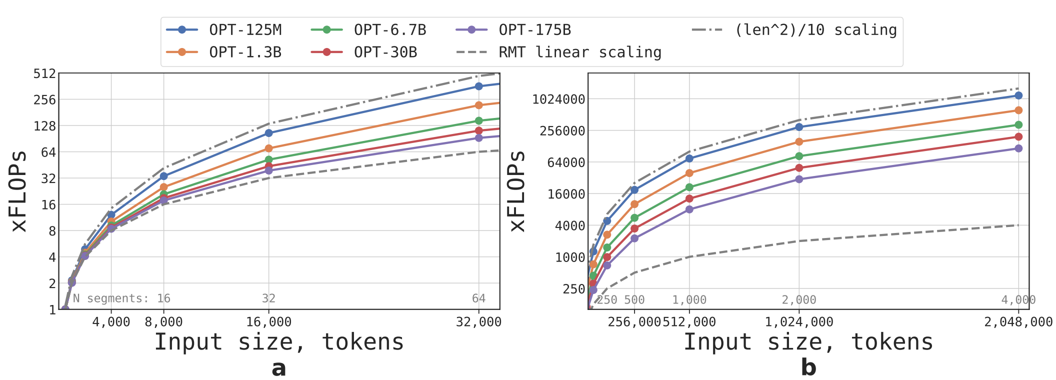

Figure 2: **RMT inference scales linearly with respect to the input sequence length**. We estimate the required FLOP increase for the forward pass compared to running models on sequences with 512 tokens. **a**: lengths from 512 to 32,000 tokens, **b**: lengths from 32,000 to 2,048,000 tokens. The RMT segment length is fixed at 512 tokens. While larger models (OPT-30B, OPT-175B) tend to exhibit near-linear scaling on relatively short sequences up to 32,000, they reach quadratic scaling on longer sequences. Smaller models (OPT-125M, OPT-1.3B) demonstrate quadratic scaling even on shorter sequences. On sequences with 2,048,000 tokens, RMT can run OPT-175B with $\times 29$ fewer FLOPs and with $\times 295$ fewer FLOPs than OPT-135M.

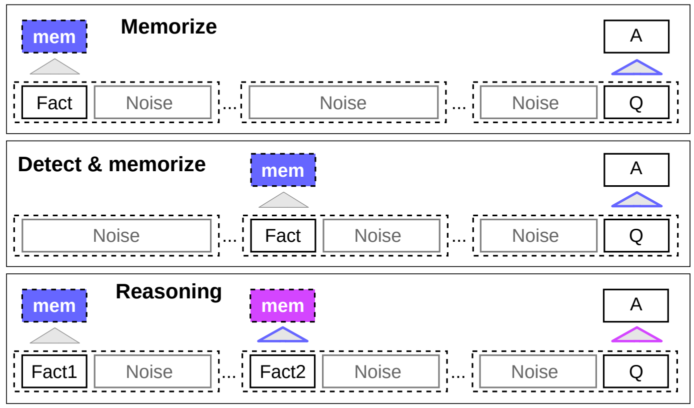

Figure 3: **Memory-intensive synthetic tasks**. Synthetic tasks and the required RMT operations to solve them are presented. In the Memorize task, a fact statement is placed at the start of the sequence. In the Detect and Memorize task, a fact is randomly placed within a text sequence, making its detection more challenging. In the Reasoning task, two facts required to provide an answer are randomly placed within the text. For all tasks, the question is at the end of the sequence. 'mem' denotes memory tokens, 'Q' represents the question, and 'A' signifies the answer.

only within segment boundaries. Larger Transformer models tend to exhibit slower quadratic scaling with respect to sequence length because of compute-heavy FFN layers (which scale quadratically with respect to hidden size). However, on extremely long sequences $> 32{,}000$, they fall back to quadratic scaling. RMT requires fewer FLOPs than non-recurrent models for sequences with more than one segment (> 512 in this study) and can reduce the number of FLOPs by up to $\times 295$ times. RMT provides a larger relative reduction in FLOPs for smaller models, but in absolute numbers, a $\times 29$ times reduction for OPT-175B models is highly significant.

4 Memorization Tasks

To test memorization abilities, we constructed synthetic datasets that require memorization of simple facts and basic reasoning. The task input consists of one or several facts and a question that can be answered only by using all of these facts. To increase the task difficulty, we added natural language text unrelated to the questions or answers. This text acts as noise, so the model's task is to separate facts from irrelevant text and use them to answer the questions. The task is formulated as a 6-class classification, with each class representing a separate answer option.

Facts are generated using the bAbI dataset [Weston et al., 2016], while the background text is sourced from questions in the QuALITY [Pang et al., 2022] long QA dataset.

```
Background text: ... He was a big man, broad-shouldered and still thin-waisted.
Eddie found it easy to believe the stories he had heard about his father ...
```

The first task tests the ability of RMT to write and store information in memory for an extended time (Figure 3, top). In the simplest case, the fact is always located at the beginning of the input, and the question is always at the end. The amount of irrelevant text between the question and answer is gradually increased, so that the entire input does not fit into a single model input.

```
Fact: Daniel went back to the hallway.
Question: Where is Daniel?
Answer: hallway
```

Fact detection increases the task difficulty by moving the fact to a random position in the input (Figure 3, middle). This requires the model to first distinguish the fact from irrelevant text, write it to memory, and later use it to answer the question located at the end.

Another important operation with memory is being able to operate with several facts and current context. To evaluate this function, we use a more complicated task, called "reasoning", where two

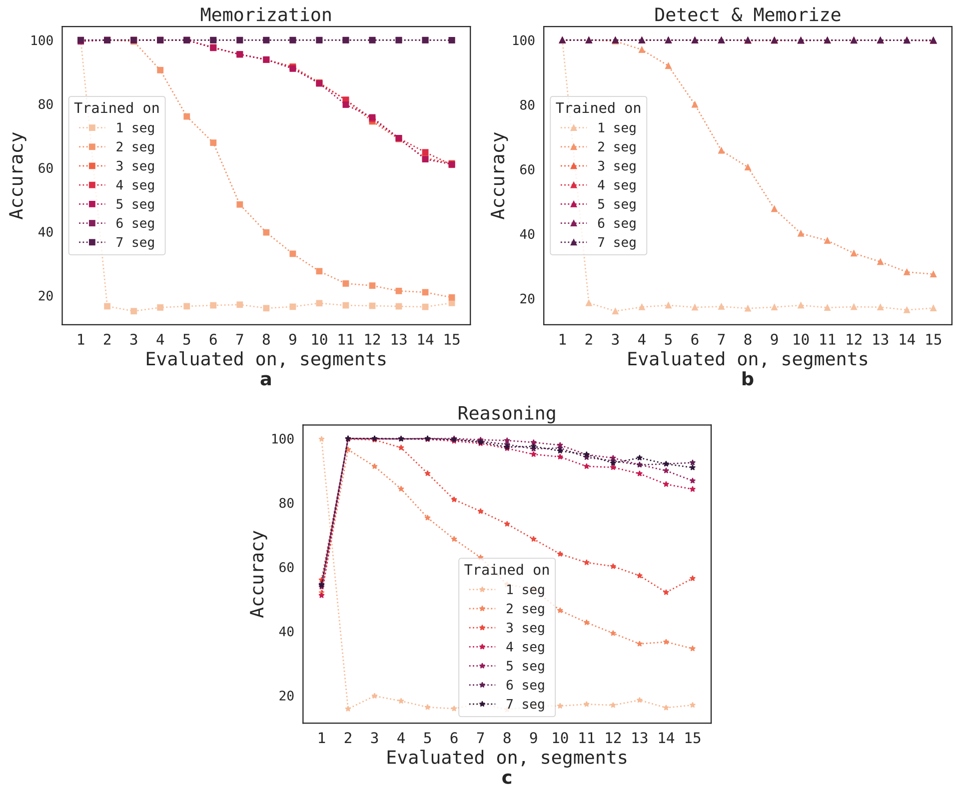

Figure 4: **Generalization of memory retrieval**. Evaluation of checkpoints trained on 1-7 segment tasks with memory size 10 on varying input lengths. **a**: Memorization task, **b**: Detection & memorization, **c**: Reasoning. Models trained on more than 5 segments generalize well on longer tasks.

facts are generated and positioned randomly within the input sequence (Figure 3, bottom). The question posed at the end of the sequence is formulated in a way that any of the facts must be used to answer the question correctly (i.e., the *Two Argument Relation* bAbI task).

```
Fact1: The hallway is east of the bathroom.
Fact2: The bedroom is west of the bathroom.
Question: What is the bathroom east of?
Answer: bedroom
```

5 Learning Memory Operations

We use the pretrained models from Hugging Face Transformers [Wolf et al., 2020] as backbones for RMT in our experiments. All models are augmented with memory and trained using the AdamW optimizer [Loshchilov and Hutter, 2019] with linear learning rate scheduling and warmup. Technical details of training and full set of hyperparameters will be available in the Appendix and training scripts in the GitHub repository.Memorization task experiments were conducted using 4-8 Nvidia 1080ti GPUs. For longer sequences, we speed up evaluation by switching to a single 40GB Nvidia A100.

5.1 Curriculum Learning

We observe that using a training schedule greatly improves solution accuracy and stability. Initially, RMT is trained on shorter versions of the task, and upon training convergence, the task length is increased by adding one more segment. The curriculum learning process continues until the desired input length is reached.

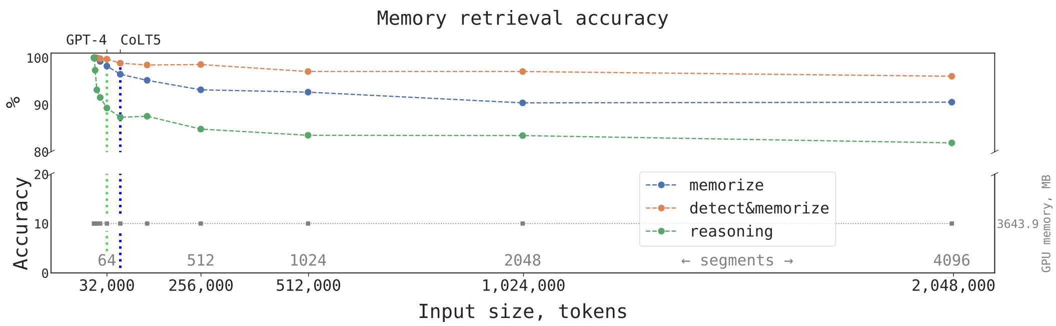

Figure 5: **Recurrent Memory Transformer retains information across up to 2×10^6 tokens**. By augmenting a pre-trained BERT model with recurrent memory [Bulatov et al., 2022], we enabled it to store task-specific information across 7 segments of 512 tokens each. During inference, the model effectively utilized memory for up to 4,096 segments with a total length of 2,048,000 tokens—significantly exceeding the largest input size reported for transformer models (64K tokens for CoLT5 [Ainslie et al., 2023], and 32K tokens for GPT-4 [OpenAI, 2023], and 100K tokens for Claude). This augmentation maintains the base model's memory size at 3.6 GB in our experiments.

In our experiments, we begin with sequences that fit in a single segment. The practical segment size is 499, as 3 special tokens of BERT and 10 placeholders for memory are reserved from the model input, sized 512. We notice that after training on shorter tasks, it is easier for RMT to solve longer versions as it converges to the perfect solution using fewer training steps.

5.2 Extrapolation Abilities

How well does RMT generalize to different sequence lengths? To answer this question, we evaluate models trained on a varying number of segments to solve tasks of larger lengths (Figure 4). We observe that most models tend to perform well on shorter tasks. The only exception is the single-segment reasoning task, which becomes hard to solve once the model is trained on longer sequences. One possible explanation is that since the task size exceeds one segment, the model stops expecting the question in the first segment, leading to quality degradation.

Interestingly, the ability of RMT to generalize to longer sequences also emerges with a growing number of training segments. After being trained on 5 or more segments, RMT can generalize nearly perfectly for tasks twice as long. To test the limits of generalization, we increase the validation task size up to 4096 segments or 2,043,904 tokens (Figure 5). RMT holds up surprisingly well on such long sequences, with Detect & memorize being the easiest and Reasoning task the most complex.

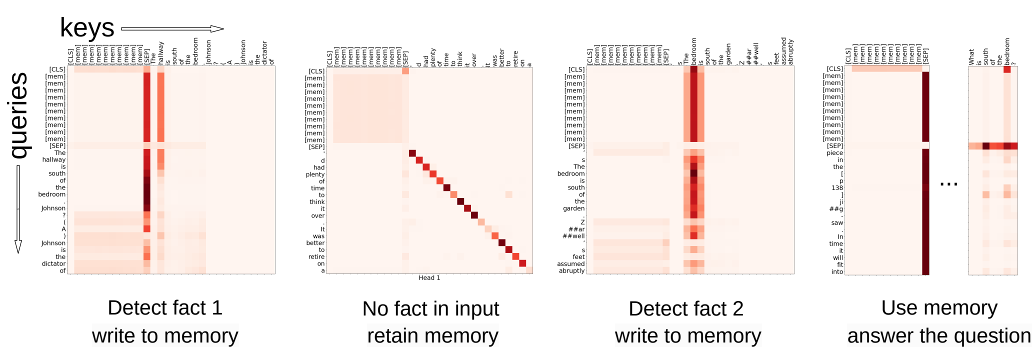

Figure 6: **Attention maps for operations with memory**. These heatmaps show operations performed during specific moments of a 4-segment reasoning task. The darkness of each pixel depends on the attention value between the corresponding key and value. From left to right: RMT detects the first fact and writes its content to memory ([mem] tokens); the second segment contains no information, so the memory keeps the content unchanged; RMT detects the second fact in reasoning tasks and appends it to memory; CLS reads information from the memory to answer the question.

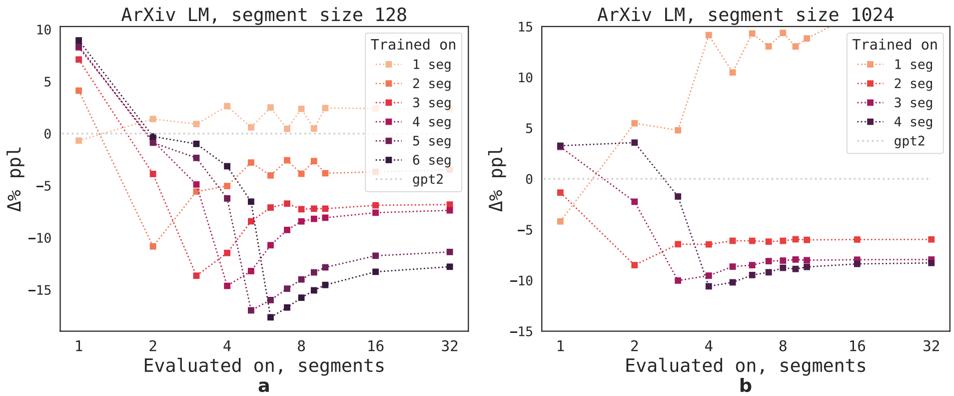

Figure 7: **Generalization of memory on language modeling task**. Models with input sizes **a:** 128 and **b:** 1024 trained with RMT show better performance and generalization across longer sizes of context. Perplexity improvement from training RMT with memory size 2 compared to training the baseline GPT-2 for same number of steps.

By examining the RMT attention on specific segments, as shown in Figure 6, we observe that memory operations correspond to particular patterns in attention. Furthermore, the high extrapolation performance on extremely long sequences, as presented in Section 5.2, demonstrates the effectiveness of learned memory operations, even when used thousands of times. The RMT does not have any specific memory read/write modules and Transformer learns how to operate with memory recurrently. This is particularly impressive, considering that these operations were not explicitly motivated by the task loss.

6 Language Modeling

To study the impact of memory on long text understanding, we focus on the long text language modeling task conducted using the recurrent approach. To capture long-term dependencies in text, memory is required to find and store various type of information between segments. We train the GPT-2 Hugging Face checkpoint with 2 memory tokens using the recurrent memory approach on the ArXiv documents from The Pile [Gao et al., 2020]. The dataset is preprocessed by splitting each document into non-overlapping segments of fixed length, which are prepended with their respective histories that consist of several segments. During both training and evaluation we process history and target segments one by one and calculate loss and perplexity only on the last target segment. Similarly to memorization tasks, we employ curriculum learning for training, starting without history and then gradually increasing context size. Language modeling experiments were done on 1-4 A100 GPUs with single curriculum stage taking up to 2 GPU-days.

As expected, increasing the effective context size leads to an improvement in perplexity (Figure 7). RMT trained for an equal number of steps as the baseline GPT-2 displays substantially lower perplexity values. With increasing number of segments in train RMT starts exhibiting better tolerance to higher history sizes. Performance of memory models trained without history suffers when applied to long contexts, but improves after multi-segment training.

We explore the limits of generalization using two tactics. First, we extend the context to contain up to 1024 segments and run RMT trained on constant number of segments, which is shown in Figure 8 **(a)**. After a certain context size the perplexity stops changing, remaining stable even when handling sequences with more than 1M tokens. Next we test robustness of RMT by introducing noise from another distribution in its context. Instead of containing relevant history, a fixed number of input segments is sampled using articles from Wikitext-2 dataset, introduced in [Merity et al., 2017]. Figure 8 **(b)** illustrates the ability of RMT to retain its superiority over GPT-2 even with useful context vastly outnumbered by noise. To understand how memory utilized during generation of the sequence we measured perplexity for every positon in it (see Figure 9). Baseline shows low prediction quality at the beginning of the sequence due to short context available to condition generation. On the other

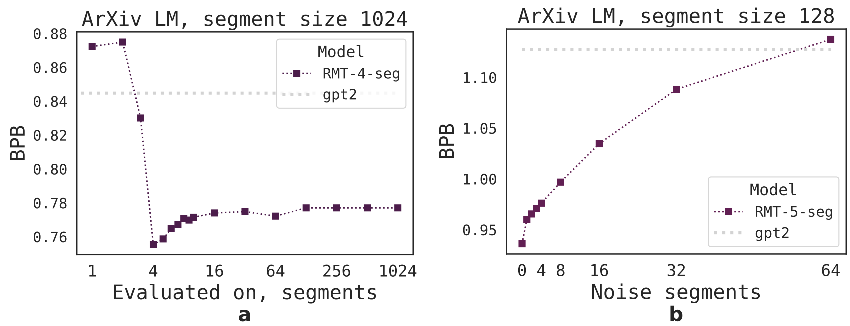

Figure 8: **Finding the context size limit of recurrent models**. **a:** Increasing the number of segments in context of RMT with 2 memory tokens and segment size 1024 reaches a perplexity plateau after a certain size but still below baseline up to more then one million tokens. **b:** Adding noise segments from another distribution to context gradually decreases RMT performance. The bits per byte value (BPB) is computed using the mean bits per token value for the Arxiv set from the Pile [Gao et al., 2020].

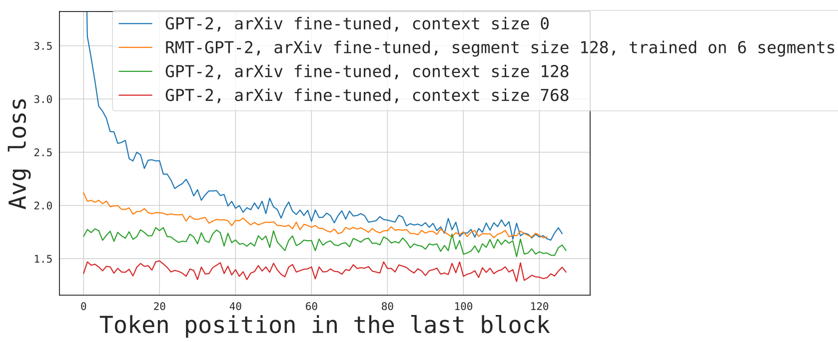

Figure 9: **Memory improves prediction at a beginning of a segment.** As we can see, there is an increase in the loss for tokens at the beginning for GPT-2 (context size 0), showing that it struggles to predict the first tokens since they have no context. The RMT keeps information about previous segments in memory tokens, which helps it to improve tokens predictions. However, showing the model the exact previous context (context size 128 and 768) allows for larger loss gains, but at a higher inference cost. This also shows the importance of local context for language modeling.

hand, RMT ensures equally good prediction for all tokens due to carryover of information from the previous segment.

7 Formal Mathematics

In this section, we fine-tune our model on a complex mathematical task: generating a proof for a given mathematical theorem in formal language. For our experiments, we utilized Lean 3 [de Moura et al., 2015] and its library, Mathlib [mathlib Community, 2020], which contains a range of formalized theories.

Each proof relies on known results, referred to as lemmas. To ensure an effective model, it must accurately assess the relevance of a lemma to the given proof. Subsequently, it should memorize the lemma's name and incorporate it within the proof. To construct our dataset, we organized each sample into a sequence format. The sequence comprises the theorem statement at the beginning, followed by a randomly ordered list of relevant and irrelevant lemmas, and concludes with the human-written proof. By adjusting the presence of irrelevant lemmas, we control the sequence length. We further divide the sequence into non-overlapping segments of fixed size.

For training and evaluation, we calculate the loss and perplexity of the entire sequence. Similar to memorization tasks, we train the RMT model and gradually increase size of the sequences. As our backbone, we employ GPTNeo [Black et al., 2021] with 1.3B parameters. We incorporate 10 memory tokens and set the segment size to 2028.

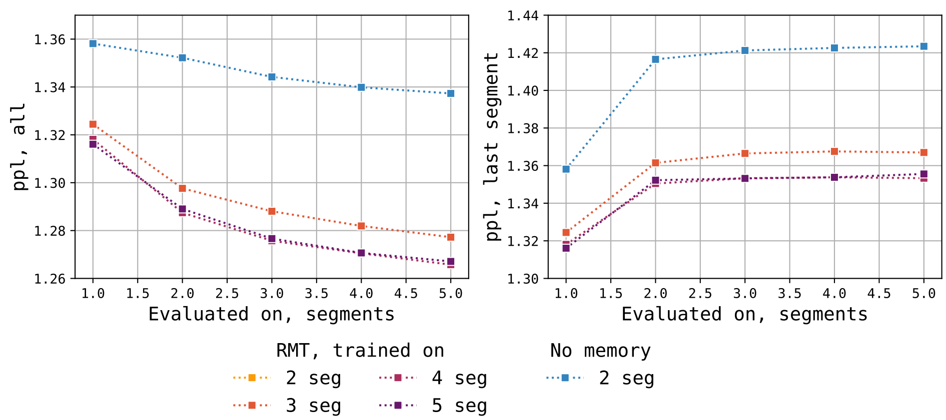

Figure 10: **Lemmas memorization for a theorem proving.** Evaluation of the RMT model and backbone model without memory. Two metrics are calculated: perplexity on all tokens of the sequence (left) and perplexity on the last segment of the sequence (right). RMT model shows better quality.

To assess the performance of the RMT model, we compare it with GPTNeo without memory trained on a sequences of 2 segments (first segment always contains the theorem statement and the second contains the proof). GPTNeo undergoes fine-tuning using the same number of tokens as RMT with 2 segments. Figure 10 shows the results of the RMT model. The RMT model improves perplexity compared to the memory-less model. However, training with 4 or more segments does not enhance predictions for longer sequences. According to how the sequence is constructed and split into segments, we hypothesize that the model is more concentrated on learning to remember the beginning of the last lemma in the previous segment to predict its end in the subsequent segment. The effect of detecting and memorizing relevant lemmas and utilizing them in proof generation is less notable. We believe that the results can be improved by more careful loss construction and data preparation.

8 Conclusions

The problem of long inputs in Transformers has been extensively studied since the introduction of this architecture. Our research has presented a series of significant advancements in augmenting and training of Transformer language models. The work expands the conventional capabilities of these models through the integration of token-based memory storage and segment-level recurrence using recurrent memory (RMT). This mechanism propels the abilities of both encoder-only and decoder-only pre-trained Transformers, revealing an unprecedented level of scalability.

We have shown that by employing the RMT approach, even models pre-trained on shorter sequences can be effectively adapted to manage tasks involving significantly longer sequences. This demonstrates that the input length originally designed for the model does not necessarily restrict its potential capabilities, thus offering a new perspective on the adaptability of Transformer models.

Our work further uncovered the remarkable adaptability of the trained RMT models in extrapolating to tasks of varying lengths. The results obtained showcased the RMT's ability to handle sequences exceeding 1 million tokens. Importantly, the computational requirements scaled linearly, thereby maintaining computational efficiency even as task length drastically increased. This is a substantial contribution that could lead to broader applications and improved performance in handling large-scale data. Through an analysis of attention patterns, we provided insight into the operations RMT engages to manipulate memory.

Overall, our research contributes significantly to the understanding and enhancement of pre-trained Transformer language models. It offers a promising direction for future work, particularly in terms of handling longer sequences and improving the adaptability of these models.

References

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Philip Pham, Anirudh Ravula, and Sumit Sanghai. Etc: Encoding long and structured data in transformers, 2020.

Joshua Ainslie, Tao Lei, Michiel de Jong, Santiago Ontañón, Siddhartha Brahma, Yury Zemlyanskiy, David Uthus, Mandy Guo, James Lee-Thorp, Yi Tay, Yun-Hsuan Sung, and Sumit Sanghai. Colt5: Faster long-range transformers with conditional computation, 2023.

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

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew R Gormley. Unlimiformer: Long-range transformers with unlimited length input. *arXiv preprint arXiv:2305.01625*, 2023.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021. URL `https://doi.org/10.5281/zenodo.5297715`.

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 11079–11091. Curran Associates, Inc., 2022. URL `https://proceedings.neurips.cc/paper_files/paper/2022/file/47e288629a6996a17ce50b90a056a0e1-Paper-Conference.pdf`.

Mikhail S Burtsev, Yuri Kuratov, Anton Peganov, and Grigory V Sapunov. Memory transformer. *arXiv preprint arXiv:2006.11527*, 2020.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4012. URL `https://aclanthology.org/W14-4012`.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL `https://aclanthology.org/P19-1285`.

Leonardo de Moura, Soonho Kong, Jeremy Avigad, Floris Van Doorn, and Jakob von Raumer. The lean theorem prover (system description). In *Automated Deduction-CADE-25: 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings 25*, pages 378–388. Springer, 2015.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, 2019. URL `https://aclweb.org/anthology/papers/N/N19/N19-1423/`.

SiYu Ding, Junyuan Shang, Shuohuan Wang, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. ERNIE-Doc: A retrospective long-document modeling transformer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2914–2927, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.227. URL `https://aclanthology.org/2021.acl-long.227`.

Angela Fan, Thibaut Lavril, Edouard Grave, Armand Joulin, and Sainbayar Sukhbaatar. Addressing some limitations of transformers with feedback memory. *arXiv preprint arXiv:2002.09402*, 2020.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626):471–476, October 2016. ISSN 00280836. URL `http://dx.doi.org/10.1038/nature20101`.

Edward Grefenstette, Karl Moritz Hermann, Mustafa Suleyman, and Phil Blunsom. Learning to transduce with unbounded memory, 2015.

Caglar Gulcehre, Sarath Chandar, Kyunghyun Cho, and Yoshua Bengio. Dynamic neural turing machine with soft and hard addressing schemes. *arXiv preprint arXiv:1607.00036*, 2016.

Caglar Gulcehre, Sarath Chandar, and Yoshua Bengio. Memory augmented neural networks with wormhole connections. *arXiv preprint arXiv:1701.08718*, 2017.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. LongT5: Efficient text-to-text transformer for long sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 724–736, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.55. URL `https://aclanthology.org/2022.findings-naacl.55`.

Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. Star-transformer. 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 1315–1325, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1133. URL `https://aclanthology.org/N19-1133`.

Ankit Gupta and Jonathan Berant. Gmat: Global memory augmentation for transformers. *arXiv preprint arXiv:2006.03274*, 2020.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `https://doi.org/10.1162/neco.1997.9.8.1735`.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 30016–30030. Curran Associates, Inc., 2022. URL `https://proceedings.neurips.cc/paper_files/paper/2022/file/c1e2faff6f588870935f114ebe04a3e5-Paper-Conference.pdf`.

DeLesley Hutchins, Imanol Schlag, Yuhuai Wu, Ethan Dyer, and Behnam Neyshabur. Block-recurrent transformers. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=uloenYmLCAo`.

Armand Joulin and Tomas Mikolov. Inferring algorithmic patterns with stack-augmented recurrent nets, 2015.

Jie Lei, Liwei Wang, Yelong Shen, Dong Yu, Tamara L. Berg, and Mohit Bansal. Mart: Memory-augmented recurrent transformer for coherent video paragraph captioning, 2020.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=Bkg6RiCqY7`.

The mathlib Community. The lean mathematical library. In *Proceedings of the 9th ACM SIGPLAN International Conference on Certified Programs and Proofs*. ACM, jan 2020. doi: 10.1145/3372885.3373824. URL `https://doi.org/10.1145%2F3372885.3373824`.

Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

Yuanliang Meng and Anna Rumshisky. Context-aware neural model for temporal information extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 527–536, 2018.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=Byj72udxe`.

OpenAI. Gpt-4 technical report, 2023.

Richard Yuanzhe Pang, Alicia Parrish, Nitish Joshi, Nikita Nangia, Jason Phang, Angelica Chen, Vishakh Padmakumar, Johnny Ma, Jana Thompson, He He, and Samuel Bowman. QuALITY: Question answering with long input texts, yes! In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5336–5358, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.391. URL https://aclanthology.org/2022.naacl-main.391.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Jack W Rae, Jonathan J Hunt, Tim Harley, Ivo Danihelka, Andrew Senior, Greg Wayne, Alex Graves, and Timothy P Lillicrap. Scaling memory-augmented neural networks with sparse reads and writes, 2016.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SylKikSYDH.

C Stephen. Kleene. representation of events in nerve nets and finite automata. *Automata studies*, 1956.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks, 2015.

Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10): 1550–1560, 1990.

Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1410.3916.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomás Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1502.05698.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45, 2020.

Qingyang Wu, Zhenzhong Lan, Kun Qian, Jing Gu, Alborz Geramifard, and Zhou Yu. Memformer: A memory-augmented transformer for sequence modeling. In *Findings of the Association for Computational Linguistics: AACL-IJCNLP 2022*, pages 308–318, Online only, November 2022a. Association for Computational Linguistics. URL https://aclanthology.org/2022.findings-aacl.29.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2022b. URL https://openreview.net/forum?id=TrjbxzRcnf-.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/c8512d142a2d849725f31a9a7a361ab9-Paper.pdf.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068, 2022.