Long-Range Memory Transformers

Akarsh Kumar
Massachusetts Institute of Technology
akarshkumar0101@gmail.com

Abstract

Long-range memory is commonly recognized as one of the key things missing from the current state of the art Transformers in natural language processing. Current approaches to adding memory to transformers do not capture the necessary benefits that memory should provide, such as scalability, abstraction, compositionality, and hierarchy. One of the most notable problems with these all of these approaches is the models' tendency to acquire a recency bias and only process short-range information and ignore long-range signals. We propose a new architecture, the Long-Range Memory Transformer (LRMT), which creates long-range representations by explicitly separating out the gradient information into long-range and short-range components. This allows us to maintain a growing list of long-range memory representations of all contexts the model has seen so far and attend to them as needed in the future. The training of this model is made efficient by only ever maintaining two copies of the network. We open source all code for this project at https://github.com/akarshkumar0101/transformer-memory/tree/master/skip.

1 Introduction

The field of natural language processing has been dominated by the Transformer neural architecture and its attention mechanism (Vaswani et al., 2017). Transformers reached state of the art results on essentially every task, which led to the development of "foundation models": large language models pre-trained on lots of language which can then be adapted for many specific tasks.

Transformers perform extremely well in language modeling but suffer from a very significant problem: they are critically bounded in sequence length by their context size. Increasing their context size is not a scalable approach since it requires $O(n^2)$ time and space compute in the context length.

This paper aims to tackle one of the most important questions in language modeling and AI: how can extremely long-range memory (near 100,000 tokens) naturally emerge within Transformers? If an NLP architecture can naturally model long-range memories it may revolutionize the field. Existing approaches to memory suffer from several problems including lack of scalability or compositionality.

One key hypothesis for this project is that, in Transformers, the representations created by the tokens contain predominantly short-range information. This is because the gradients that update those representations primarily flow from the future words that are nearby. It is well known that information dependence in language falls off with distance, d, at a rate of $1/d$. This means that the gradient information to update word w_t's token comes mostly $w_{t+1}, w_{t+2}, w_{t+3}$. Information from more future words do come, but the gradient signal is much smaller due to the lack of dependence. Thus, the created representations are dominated by short-range gradients. Due to this, we hypothesize that Transformers learn a recency bias and do not fully utilized long-range information. We back this claim up by showing the per-character perplexity does not improve past a certain point, indicating it is not doing long-range information processing, though long-range dependencies in the underlying text exist.

To overcome all of these limitations, we propose a novel architecture, the Long-Range Memory Transformer (LRMT). LRMT explicitly separates out the long-range and short-range gradient information to create "long-range memory tokens" which may not be useful for prediction in this current context, but are forced to be useful arbitrarily far in the future. LRMT's memory representations are not dominated by short-range informa-

tion, but rather enjoy long-range interactions. We show that LRMT does indeed use this memory to achieve lower perplexity in future contexts. However, LRMT still has limitations generalizing to more memory tokens at inference time. We provide advice for future work attempting to fix the limitations and build on our work.

2 Related Work

Lots of work has been proposed to address memory in natural language processing.

Recurrent Networks: Recurrent Neural Networks (RNNs) and more specifically, Long Short-Term Memory Networks (LSTMs) are recurrent networks trained with backpropagation-through-time (Olah, 2015). Although they enjoy, theoretically, an infinite range memory, in practice they suffer from vanishing and exploding gradients. This makes it extremely difficult to do credit assignment properly in extremely long range sequences as the gradient information gets noisy or destroyed. RNNs and LSTMs also suffer from the fact that each time iteration requires an entire copy of the recurrent cell in memory These two factors together make RNNs and backpropagation-through-time not a scalable or practical approach to achieving long-range memory in language models.

Vanilla Transformer: Transformers were shown to be one of the biggest achievements in natural language processing and in AI in general (Vaswani et al., 2017). However, the power of the vanilla Transformer architecture requires $O(n^2)$ time and space in the sequence length. This makes it great for smaller sequence lengths, but is not scalable for a one/two order of magnitude increase in the sequence length, given the current compute bottlenecks. The information flow of a vanilla Transformer is shown in Fig 1.

Transformer-XL: Transformer-XL improves upon the vanilla transformer by caching hidden states from previous contexts with a custom trick to handle the positional encodings (Dai et al., 2019). However, this simply extends the context length by a factor and does not give a reliable way to scale up to extremely long sequence lengths. The information flow of Transformer-XL is shown in Fig 1b.

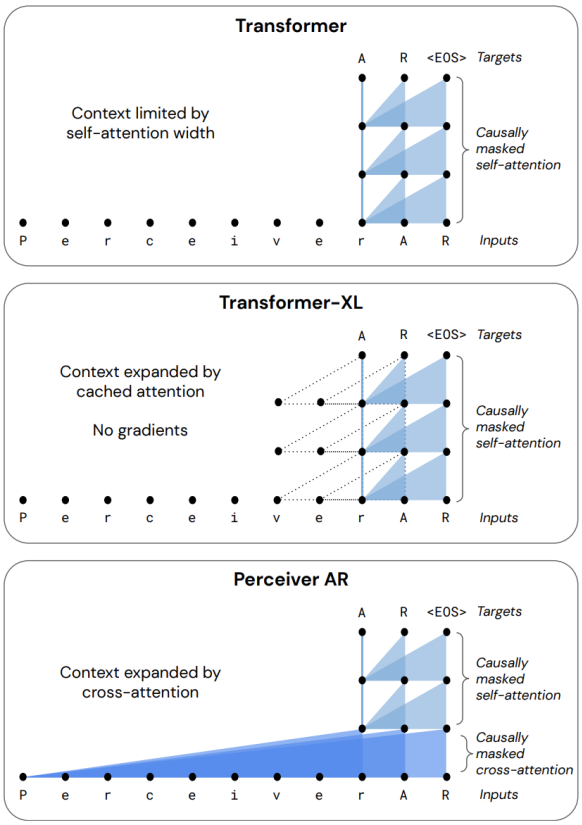

Figure 1: The information flow in the (a) vanilla Transformer, (b) Transformer-XL, and (c) Perceiver-AR. Retrieved from https://arxiv.org/pdf/2202.07765.pdf.

Perceiver-AR: Perceiver-AR performs cross attention with k latent tokens and the the entire sequence of size n in the first layer, making it $O(n)$ in time and space compute, which scales well (Hawthorne et al., 2022). However, the Perceiver-AR only attends to the raw input tokens of the sequence and thus is not able to create a compositional/contextual representations of the sequence in any way for long-range sequences. This is a huge downside and essentially voids all the powerful representations created by a strong Transformer architecture. However, if these representations that are cross-attended to were in some way to be compositional/contextual, this model would prove to be a powerful approach. Our architecture attempts to do exactly this, while keeping it trainable through a clever training proposal. The information flow of Perceiver-AR is shown in Fig 1c.

Compressive Transformer: Compressive Transformer tries to compress past memories by performing a lossless compression with a reconstruction loss (Rae et al., 2019). This leads to the created memories not being abstract/hierarchical enough in order for it to be useful far in the future. Some intelligent *lossy* compression mechanism will be needed for extremely long sequence lengths, as shown by natural intelligence.

3 Methods

In order to solve the memory problems discussed above, we propose the long-range memory Transformer (LRMT) architecture. This architecture explicitly separates out short-range and long-range information in the gradients allowing it to create explicitly short-range and long-range representations.

When processing a context, LRMT performs regular Transformer processing to outputs the predicted next tokens. However, LRMT also outputs a sequence of "long-range" memory tokens, which are representations of the context that are abstract enough to be useful *arbitrarily far in the future*. They are *not constrained to be useful in any way for the processing of the current context*. More specifically, these special representations are not in any way drowned out by the *short-range local high-frequency* information of the current context, but rather capture *long-range global low-frequency* features which should generalize the information for use in the far future. Said another way, this long-range memory need not be useful right now, but must be useful far in the future.

To use past memories, LRMT can also be inputted any number of long-range memory tokens from past contexts to aid in the processing of the current context.

3.1 Memory Creation

The vanilla transformer stack consists of a sequence of causal attention blocks. We keep this regular transformer stack, but somewhere in the middle (after a third of the blocks), we extract the representation tokens. These tokens are then fed into *full attention* (non-causal) "memory-creator" blocks. The output of these blocks are the long-range memory tokens. These blocks do not need to be causal since their job is simply to summarize the entire context, in any way possible, as to create memories useful for the future. This pathway is shown by the non-causal attention block in the middle of Fig 2.

3.2 Memory Retreival

A similar mechanism is used for retrieving past memories to aid in the processing of the current context. The regular transformer stack is executed, but somewhere in the middle (after a third of the blocks), we extract the representation tokens. These tokens are then fed into *causal* "memory-retreiver" blocks. The output of these blocks are "memory-retrieval" tokens. Then, a full cross-attention is performed by using queries from the memory-retrieval tokens and the keys and values from the inputted memory tokens. The result is a new set of tokens which have extracted all the useful information (useful for the current context) from the memories. These tokens undergo another causal block of transformations. Then, they are merged back into the main Transformer pathway when the main branch performs cross-attention between the queries of the main pathway and the keys and values of the concatenation of the main pathway and these memory-retrieved tokens. This cross attention is doubly-causal, meaning that it is causal with respect to the main pathway tokens and the memory-retrieved tokens. This special construction of causal and non-causal blocks ensures that information current context next token information is not leaked to previous tokens, while still ensuring that past memories are compressed completely and fully attended to.

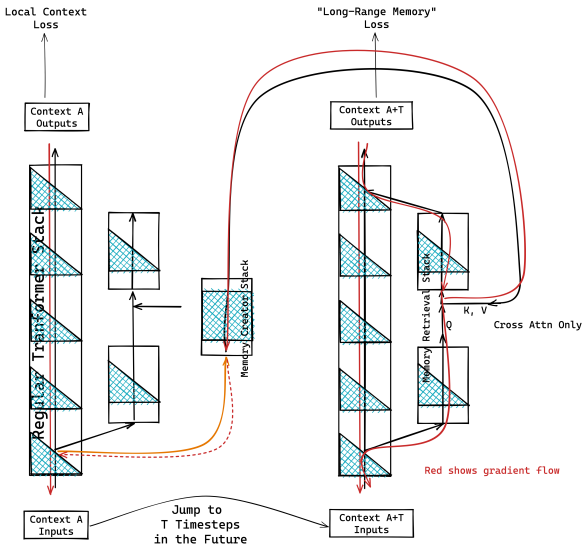

Figure 2: Our proposed model during training time. During training, only two copies of the model are stored in memory. These are fed in different contexts separated by T timesteps, where T is a random number. Model A summarizes the context A into some "long-range" representation tokens which serve as the memory tokens. These memory tokens are fed into the memory input for model B. Model B is allowed to attend to memory tokens in any way as to minimize the cross-entropy loss for context B. This decomposition enforces a special path of long-range gradients to flow to improve the perplexity of context B based on the long-range information in context A. Black lines show the forward pass of information and the red lines show the backward gradient flow of information.

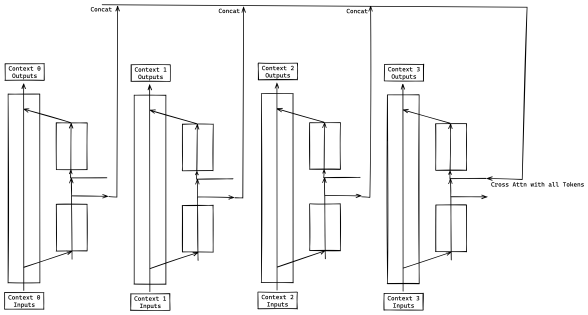

Figure 3: Our proposed model during inference time. During inference, consecutive chunks of contexts are fed into the model. The long-range representation tokens created by each context are constantly concatenated into a memory bank containing all the memories. The model is allowed to attend to all memories from previous contexts when processing the current context. This leads to a $O(n)$ time and space compute.

3.3 Training Time

During training, we take two copies of LRMT and feed them contexts A and B, separated by T tokens. Then, we feed in the long-range memory output from context A into the memory input pathway for context B to help predict B's context. By randomly varying T during training, we enforce the model create long-range memories that are useful *arbitrarily far into the future*. Fig 2 provides an overview of the training procedure. The flow of gradients in red shows that only long-range information is ever transmitted through the long-range memory creation and retriever stacks.

3.4 Inference Time

During inference time, we consecutively go through the contexts and create long-range memories for each context and concatenate all of them into a growing list of memory tokens. By inputting this list of previous memory tokens for each context, we allow the model to access all previous memories via the memory-retrieval cross-attention mechanism. This list grows with the sequence length and thus results in a $O(n)$ time and space complexity in the sequence length.

4 Results

4.1 Data and Compute

We gathered natural language data from books on Project Gutenberg (https://www.gutenberg.org/). This gave an corpus of stories that was split into a training and testing set.

We conduct all experiments on a character-level model to 1) artificially increase the amount of data, 2) ensure long-range dependencies exist, and 3) ensure large amounts of compositionality is required in the memories.

All experiments for this project were done on four Titan Xp graphics cards on a single machine provided by the Embodied Intelligence Lab's vision GPU cluster.

We implemented the vanilla Transformer and LRMT from scratch in PyTorch, building off of the GPT-2 architecture. Everything from the multi-head attention mechanism and MLP blocks to the positional encodings were built from scratch. We chose to do this to gain a full understanding of the model its weaknesses. Building off of existing codebases was not possible due to the drastic changes required in LRMT.

4.2 Hyperparamters

A learning rate sweep was initially performed to find the optimal learning rate of $3e{-}4$, which was then used to train all models. Various parameters for batch size and sequence length were also tested.

4.3 Evaluation

We evaluate all models on three per-character perplexity metrics. The first metric is per-character perplexity of characters within words. This metric captures the performance of the model on short-range information. The second metric is per-character perplexity of the first characters of common words. This metric captures the performance of the model on medium-range information. The third metric is per-character perplexity of the first characters of more rare words. Since this metric has previously been found to empirically correlate with long-range dependencies in language, this metric captures the performance of the model on long-range information. These metrics are also tracked across the token position in the context to see how the model scales up given more information in the context.

4.4 Does the vanilla Transformer Prioritize Short-Range Information?

We investigate whether or not the vanilla Transformer creates representations that are useful for long-range information processing. One way to test this is to train a very large model with a large context size and see the perplexity plotted against the token position in the context. If this perplexity keeps getting lower throughout the context, it is successfully utilizing all the information from the context, even the long-range information from the beginning of the context. We know that the character sequences consist of words which contain sentence and paragraph level dependencies. If the model were properly modeling these dependencies, we should see the perplexity of the first character of the words to decrease over the range of 100-200 words, and definitely within a context length of 64 characters. However, Fig 4 shows that the vanilla transformer only gets better at perplexity for the first 10 tokens, then plateaus. This holds true for all the frequency bins, hinting that it is not doing long-range processing and is instead creating only short-range representations useful for prediction less than 10 tokens in the future. We hypothesize that this happens because the the representations' weak long-range gradients is dominated and drowned out by the much stronger signal from the short-range gradients.

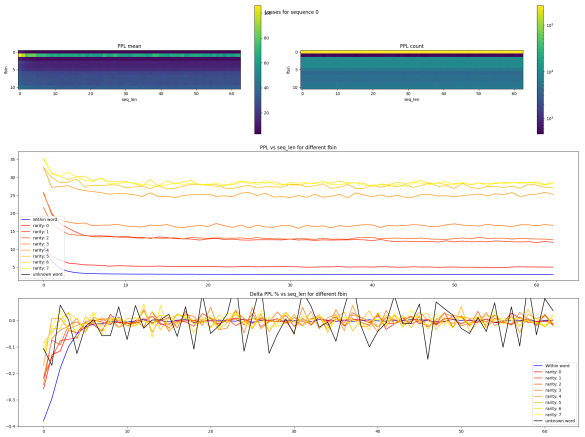

Figure 4: The perplexity statistics for a large vanilla Transformer with context length 64. The second row shows the average perplexity for within-word and common to rare tokens plotted against the position in the context. The third row shows the percentage change of the perplexity from the previous position to the current position. This plot demonstrates how the perplexity decreases given more information in the context. The regular transformer does not get better after 10 characters of past data, meaning it does not do long-range information processing.

4.5 Does Long-Range Memory Help?

Fig 5 shows that the LRMT does indeed outperform the vanilla Transformer in perplexity. The second sequence processed with LRMT, using memory from the first sequence, does have lower

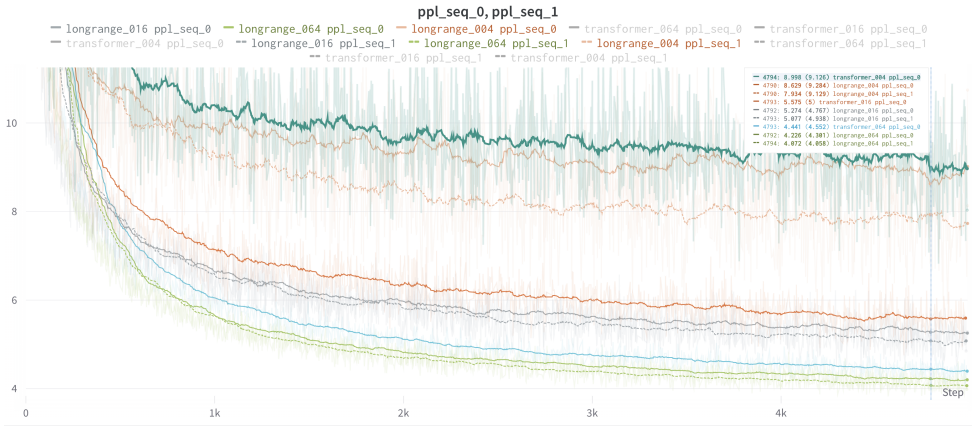

Figure 5: The perplexity of the vanilla Transformer and LRMT with various context lengths over the course of training. seq_0 indicates the first context shown to the vanilla Transformer and LRMT. seq_1 indicates the second context shown to the vanilla Transformer and LRMT. LRMT outperforms the vanilla Transformer in the second context when given the first context as memory, meaning that it is properly utilizing the memory.

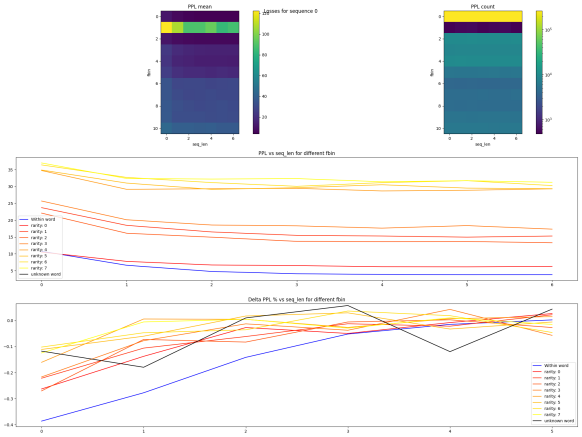

Figure 6: Perplexity statistics for the first context shown to LRMT.

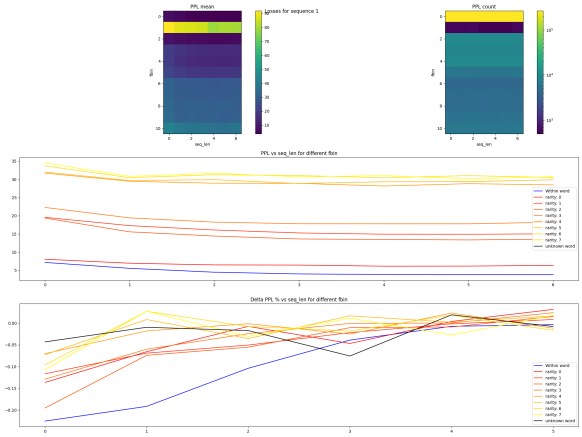

Figure 7: Perplexity statistics for the second context shown to LRMT. This outperforms the first context in essentially frequency range, meaning LRMT is using the memory from the first context properly.

perplexity than the first sequence processed by LRMT, meaning the memory mechanism is working. Interestingly, LRMT is able to create representations than even help perplexity within the first sequence, compared to the vanilla Transformer.

4.6 Does training with two contexts generalize to inference with N contexts?

We now investigate whether it is okay to train using only two copies of the network, while chaining together N copies during inference time. Fig 6, Fig 7, and Fig 8 show the perplexity statistics for the first, second, and fourth contexts processed by LRMT during inference. Each context is given all the previous contexts' memory tokens as memory inputs. The second context does indeed show better performance than the first, since the memory mechanism does indeed work. However, the fourth context shows the same performance as the first, which is worse than the second. Thus, attending to a different number of tokens during training and inference time is not valid. One option to mitigate this issue would be to incorporate the ideas from ALiBi to help extrapolative generalization in the attention mechanism to larger sequence lengths (Press et al., 2022).

4.7 Issues Encountered

We encountered two issues during the training of LRMT.

Long-range gradient is near zero. Fig 9 shows that the magnitude of the gradients for the memory-creator and memory-retrieval blocks were one or two orders of magnitude less than the

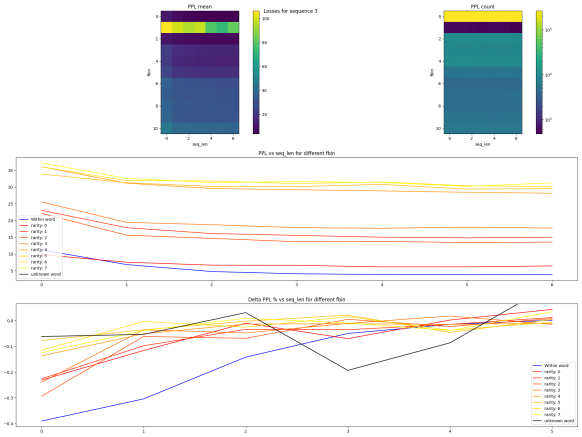

Figure 8: Perplexity statistics for the fourth context shown to LRMT. This degrades performance back to the first context, meaning LRMT struggles to utilize memory when given memory from the past three contexts. This is due to the fact that LRMT is trained using only one context's memory, and at inference is using the past N contexts' memory.

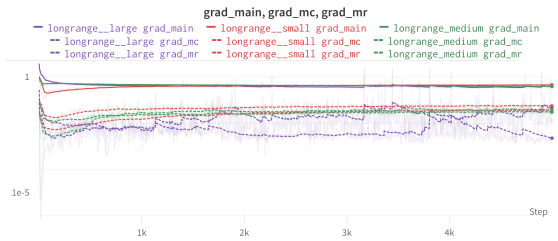

Figure 9: The gradient magnitude in the main blocks, memory-creator blocks, and memory-retriever blocks in LRMT. The memory-creator and memory-retriever blocks suffer from one or two orders of magnitude smaller gradients, which is a problem during training.

main blocks. This issue was partially solved by decreasing T, thus increasing the dependency between the two contexts in the language. Another solution left for future work may be to scale up the gradient and increase the batch size drastically to get accurate long-range gradients.

Transformer learns to ignore long-range pathway tokens. We found that the transformer consistently learned to ignore the long-range pathway memory-retrieved tokens, and only used them sometimes. To what extent this is a problem is unknown, since maybe memory is supposed to be only sparsely activated and attended to. However, if future work finds this to be a significant problem, one fix is to enforce that the transformer must half-way attend to context tokens as well as half-way attend to long-range tokens. Another more natural fix may be to use ALiBi positional encodings ensuring that the transformer cannot use position to discriminate against long-range tokens, and must use content based information (Press et al., 2022).

4.8 Discussion

Overall, LRMT does show promise in its ability to create long-range memory representations. However, more thorough testing of the true effectiveness of LRMT compared to vanilla Transformers is still needed, such as in word-level modeling. More study is also needed on the true properties of the long-range memories as compared to the vanilla Transformer's representations. Which representations are good for which kinds of tasks and at which timescales? Despite the effectiveness of this approach still being up in the air, we believe this general paradigm of separating long-range and short-range information will be critical in future work. We outline potential directions going forward in the next section.

5 Future Work

One potentially interesting direction for future work is to benchmark the performance of the long-range model and the previous models on a "Which sentence came first?" task. Essentially, does the network learn representations for sentences, such that, when we put a classification head on top of these representations for two separate sentences, it can correctly tell which sentence came first? If the representations can answer this question, it is very

likely it is containing long-range information in a sense.

Another interesting direction forward will be to experiment with ALiBi positional encodings (Press et al., 2022) as outlined in the previous section.

6 Conclusion

We proposed the Long-Range Memory Transformer (LRMT), a novel approach to adding memory to the transformer which bypasses several current issues of existing memory extension models. We believe that the idea of explicitly separating out short-range and long-range graident information will be a vital step forward to ensure the proper creation of long-range memory. Without this mechanism, the long-range information is dominated and drowned out by the short-range information leading to the creation of short-sighted representations. LRMT is shown to outperform the regular vanilla transformer and properly utilize its memory for achieving lower perplexity on rare words. However, LRMT still requires many improvements to be practically useful and scale to larger inference time memory lengths.

7 Impact Statement

This work furthers the state of the art in large language models and thus all limitations and harms caused by them is applicable to this work as well. Specifically, large language models are known to exacerbate harmful biases existing the datasets they were trained on (Bender et al., 2021). For example, if the natural language data contains text that is derogatory to certain (often miniority) groups, this model will also suffer from those harmful biases. The proposed model, LRMT, may actually make this problem worse if the memory tokens contain biased and harmful information. Further investigation is needed to understand how the memory and language models in general can be trained to avoid these failure cases. Further investigation is also needed on examining the datasets used for training to mitigate the amount of bias that can leak into these models.

Acknowledgements

We thank Alexander Huth, Shailee Jain, and Yoon Kim for useful discussions on this project.

References

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? . In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. Association for Computing Machinery, New York, NY, USA, FAccT '21, pages 610–623. https://doi.org/10.1145/3442188.3445922.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. ArXiv:1901.02860 [cs, stat]. https://doi.org/10.48550/arXiv.1901.02860.

Curtis Hawthorne, Andrew Jaegle, Cătălina Cangea, Sebastian Borgeaud, Charlie Nash, Mateusz Malinowski, Sander Dieleman, Oriol Vinyals, Matthew Botvinick, Ian Simon, Hannah Sheahan, Neil Zeghidour, Jean-Baptiste Alayrac, João Carreira, and Jesse Engel. 2022. General-purpose, long-context autoregressive modeling with Perceiver AR. Technical Report arXiv:2202.07765, arXiv. ArXiv:2202.07765 [cs, eess] type: article. http://arxiv.org/abs/2202.07765.

Chris Olah. 2015. Understanding LSTM Networks – colah's blog. https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation. Technical Report arXiv:2108.12409, arXiv. ArXiv:2108.12409 [cs] type: article. http://arxiv.org/abs/2108.12409.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, and Timothy P. Lillicrap. 2019. Compressive Transformers for Long-Range Sequence Modelling. Technical Report arXiv:1911.05507, arXiv. ArXiv:1911.05507 [cs, stat] type: article. https://doi.org/10.48550/arXiv.1911.05507.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. ArXiv:1706.03762 [cs]. https://doi.org/10.48550/arXiv.1706.03762.