Why Are Positional Encodings Nonessential for Deep Autoregressive Transformers? Revisiting a Petroglyph

Kazuki Irie
Department of Psychology
Harvard University, Cambridge MA, USA
kirie@g.harvard.edu

Abstract

Do autoregressive Transformer language models *require* explicit positional encodings (PEs)? The answer is *'no'* provided they have more than one layer—they can distinguish sequences with permuted tokens without the need for explicit PEs. This follows from the fact that a cascade of (permutation invariant) set processors can collectively exhibit sequence-sensitive behavior in the autoregressive setting. This property has been known since early efforts (contemporary with GPT-2) adopting the Transformer for language modeling (Irie et al., 2019). However, this result does not appear to have been well disseminated, leading to recent rediscoveries. This may be partially due to a sudden growth of the language modeling community *after* the advent of GPT-2/3, but perhaps also due to the lack of a clear explanation in prior work, despite being commonly understood by practitioners in the past. Here we review the long-forgotten explanation why explicit PEs are nonessential for *multi-layer* autoregressive Transformers (in contrast, *one-layer* models require PEs to discern order information of their inputs), as well as the origin of this result, and hope to re-establish it as a common knowledge.

1 Introduction

The field of language modeling has seen new waves of interest after the promising results of GPT-2 (Radford et al., 2019), impressive capabilities of GPT-3 (Brown et al., 2020), and unprecedented versatility of ChatGPT and GPT-4 (Bubeck et al., 2023; Achiam et al., 2023), manipulating human languages in a way no machine has ever before.

About a decade before the current "Large Language Model era" or LLM-era[1], neural language modeling research had also seen a smaller but significant growth after Tomáš Mikolov's breakthrough results with recurrent neural network language models (Mikolov et al., 2010, 2011). This had made neural language modeling (Nakamura and Shikano, 1989; Elman, 1989; Schmidhuber and Heil, 1994; Bengio et al., 2000) a popular research topic, particularly among speech recognition and machine translation researchers as these two fields used to be the major application areas at the time when language models were not yet a standalone system— they were merely a component in a larger system with a specialized application (Jelinek et al., 1975; Brown et al., 1988), except in certain visionary work (Sutskever et al., 2011).

When the Transformer encoder-decoder architecture was shown to be successful for machine translation (Vaswani et al., 2017), several works investigated its application to build conventional (i.e., autoregressive) *language models* using the decoder, e.g., Liu et al. (2018); Radford et al. (2018); Al-Rfou et al. (2019); Dai et al. (2019); Baevski and Auli (2019), or non-autoregressive *models of language* using the encoder, e.g., Devlin et al. (2019); producing many methods and practical knowledge for optimizing Transformers to language modeling, concurrently to GPT-2 (Radford et al., 2019).

While the recent surge of interest in language modeling has been very exciting for the field, it has also led to some discontinuities, e.g., certain common knowledge and results from pre-LLM studies appear to have been lost amid this rapid growth.

This short review focuses on one of such results, namely the property that *multi-layer autoregressive Transformer language models can process sequences without explicit positional encodings* (Irie et al., 2019). In fact, it is often argued that positional encodings are necessary for Transformers,

[1]Here by "LLM-era" we roughly refer to the time after GPT-3. The term "petroplyph" in the title is a hyperbole with a double meaning: first, it highlights that results from the pre-LLM era are now largely regarded as "prehistoric" and they are overlooked; second, more specifically to the positional encoding result discussed here, figures similar to Figure 1 were frequently sketched in old notes and during whiteboard discussions from that time, but such a figure was not included in the 4-page Interspeech paper (Irie et al., 2019).

Findings of the Association for Computational Linguistics: ACL 2025, pages 551–559
July 27 - August 1, 2025

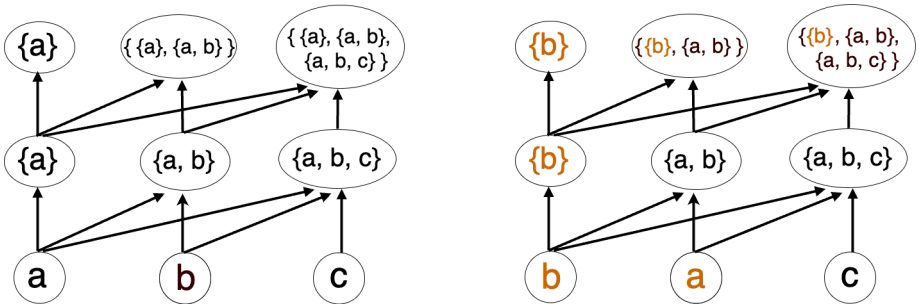

Figure 1: *A cascade of set processors can behave as a sequence processor.* An illustration of autoregressive Transformers without explicit positional encodings for two input sequences: (a, b, c) and its permutation (b, a, c). The color in the right diagram highlights the differences in terms of context seen by each layer at each position (expressed as a set). With two (or more) layers, as soon as an input is different at one position (here at the first position), autoregressive Transformers see different contexts for all later positions at the top layer. In contrast, *one-layer* models can not distinguish between these two input sequences (in this example, as soon as the second token is fed) even though they are strictly speaking not permutation invariant (since the context at the first position is different). Here we only swap the two first tokens; with a more "complex" permutation, one-layer models may see different contexts at more positions but in all cases, they systematically fail to do so in the last step.

because the self-attention operation (Vaswani et al., 2017; Cheng et al., 2016; Parikh et al., 2016; Lin et al., 2017) is permutation invariant. There is a flaw in this deduction: it overlooks the behavior of multi-layer (i.e., cascaded) self-attention in the autoregressive setting. Here we provide a simple explanation of this result (Figure 1 and Sec. 3), which, although known to language modeling practitioners of the pre-LLM era, was never published (to the best of our knowledge). We also refer back to early work on this property (Sec. 5).

2 Background: Self-Attention

Following the original definition (Vaswani et al., 2017), one Transformer "layer" consists of two sub-layers: a self-attention layer and a feedforward block. Given that a typical feedforward block processes information at each position/time step exclusively, the self-attention layer is the only sequence processing component of the Transformer layer.

Autoregressive Self-Attention. Let d and T denote positive integers. An autoregressive self-attention layer transforms an input sequence $\{\boldsymbol{x}_t\}_{t=1}^T, \boldsymbol{x}_t \in \mathbb{R}^d$ to an output sequence $\{\boldsymbol{y}_t\}_{t=1}^T, \boldsymbol{y}_t \in \mathbb{R}^d$ as follows:

$$\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t = \boldsymbol{W}_q\boldsymbol{x}_t, \boldsymbol{W}_k\boldsymbol{x}_t, \boldsymbol{W}_v\boldsymbol{x}_t \quad (1)$$

$$\boldsymbol{K}_t = \left[\boldsymbol{K}_{t-1}, \boldsymbol{k}_t\right] \in \mathbb{R}^{d\times t} \quad (2)$$

$$\boldsymbol{V}_t = \left[\boldsymbol{V}_{t-1}, \boldsymbol{v}_t\right] \in \mathbb{R}^{d\times t} \quad (3)$$

$$\boldsymbol{y}_t = \boldsymbol{V}_t\mathrm{softmax}(\boldsymbol{K}_t^\top \boldsymbol{q}_t) \quad (4)$$

where $\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v \in \mathbb{R}^{d\times d}$ are trainable weight matrices, $[\boldsymbol{A}, \boldsymbol{a}]$ denotes the concatenation of vector $\boldsymbol{a}$ to matrix $\boldsymbol{A}$ which increments the time dimension ($\boldsymbol{K}_0$ and $\boldsymbol{V}_0$ are initially empty), and $\mathrm{softmax}$ is along the time dimension. We omit the $1/\sqrt{d}$ scaling inside $\mathrm{softmax}$, as well as the output projection, which are irrelevant for our discussion.

While the equations above accurately describe the model conceptually, self-attention is also often expressed in the following *matrix form* that better reflects the possibility to parallelize computation over the time axis during training. By denoting the input as $\boldsymbol{X} = [x_1, ..., x_T] \in \mathbb{R}^{d\times T}$ ($\boldsymbol{X}_i = x_i$) and the output as $\boldsymbol{Y} = [y_1, ..., y_T] \in \mathbb{R}^{d\times T}$, it yields:

$$\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} = \boldsymbol{W}_q\boldsymbol{X}, \boldsymbol{W}_k\boldsymbol{X}, \boldsymbol{W}_v\boldsymbol{X} \quad (5)$$

$$\boldsymbol{Y} = \boldsymbol{V}\mathrm{softmax}(\boldsymbol{M} \odot (\boldsymbol{K}^\top \boldsymbol{Q})) \quad (6)$$

where $\boldsymbol{M} \in \mathbb{R}^{T\times T}$ is the so-called *attention mask*. For these equations to be equivalent to Eqs. 1-4 above, i.e., for autoregressive self-attention, $\boldsymbol{M}$ is set to be the upper triangular matrix, i.e., $\boldsymbol{M}_{i,j} = 1$ if $i \leq j$ and $\boldsymbol{M}_{i,j} = -\infty$ otherwise.

We also denote by $\boldsymbol{Y} = \mathrm{SelfAttn}(\boldsymbol{X}, \boldsymbol{M})$ the overall self-attention operation given input $\boldsymbol{X}$ and mask $\boldsymbol{M}$, grouping Eqs. 5-6.

Non-Autoregressive Self-Attention. The same equations (Eqs. 5-6) can also express *non-autoregressive* self-attention by removing the mask $\boldsymbol{M}$, i.e., by setting $\boldsymbol{M}_{i,j} = 1$ for all $i, j \in \{1, ..., T\}$. We denote such $\boldsymbol{M}$ as $\boldsymbol{M} = \mathbb{1}$.

Positional Encodings. When positional encodings (Gehring et al., 2017; Vaswani et al., 2017) are used, a vector representing discrete position t

is added to input token $\boldsymbol{x}_t$. The exact choice of PE design is irrelevant to our discussion.

3 Main Results on Positional Encodings

The goal of this short review is to provide a summary of results on the (non)necessity of positional encodings for different Transformer model variations with comprehensible explanations, and to discuss the original references (Sec. 5).

Definitions. We first define two key properties:

(1) A sequence processor $f : \mathbb{R}^{d\times T} \to \mathbb{R}^{d\times T}$ is said to be *permutation invariant* when for any input $\boldsymbol{X} \in \mathbb{R}^{d\times T}$, and its arbitrary permutation along the time/token axis $\boldsymbol{X'} \in \mathbb{R}^{d\times T}$, $f(\boldsymbol{X}) = f(\boldsymbol{X'})$.

(2) f is *fully position-sensitive* when for any inputs $\boldsymbol{X}, \boldsymbol{X'} \in \mathbb{R}^{d\times T}$, if $\boldsymbol{X}_i \neq \boldsymbol{X}'_i$, then $f(\boldsymbol{X})_j \neq f(\boldsymbol{X'})_j$ for all $j \in \{i, ..., T\}$; meaning that as soon as one input is different at position i, f produces different outputs at all the "future" positions $j \geq i$.

The necessity of using explicit PEs is tied to the model's capability to distinguish between permutated sequences[2], which can be characterized using the definitions above. (1) If a sequence processor f is *permutation invariant*, positional encodings are needed. (2) If f is *fully position-sensitive*, positional encodings are not needed. (3, Remark) As we will see, strictly speaking, being *permutation non-invariant* alone is not enough to conclude on the non-necessity of positional encodings (as all positions matters); therefore, we additonally introduce the property of being *fully position-sensitive*.

We present the main results in the form of question/answer pairs as follows.

Question 1 (Back to basics)**.** *Why are positional encodings **needed** for **non-autoregressive** Transformers?*

This is because non-autoregressive self-attention is *permutation invariant*, i.e., for any input $\boldsymbol{X}$, and its arbitrary permutation along the time/token axis $\boldsymbol{X'}$, $\text{SelfAttn}(\boldsymbol{X}, \mathbb{1}) = \text{SelfAttn}(\boldsymbol{X'}, \mathbb{1})$.

We can straightforwardly check this by directly looking at Eqs. 5-6. Without the mask, i.e., $\boldsymbol{M} = \mathbb{1}$, keys and queries from all positions interact regardless of their positions to yield attention scores (i.e., $\text{softmax}(\boldsymbol{K}^\top \boldsymbol{Q}) \in \mathbb{R}^{T\times T}$ in Eq. 6), which are used to compute weighted average, which is commutative, of values.

[2]Once it is clear that the model can distinguish between permuted sequences, there is no reason to introduce extra explicit PEs. A common wisdom is to let the model learn to use the positional signals on its own. For example, it is rather unnatural to add extra PEs to recurrent neural networks (RNNs).

Question 2 (Knowledge Bias)**.** *Why are positional encodings believed to be crucial for Transformers by "default" in the first place?*

This is partly because the standalone self-attention is a permutation-invariant set operation (this is the case, even in the autoregressive case, if we ignore the edge case of the first token; as illustrated in Figure 1).

Also, the explanation for the necessity of positional encodings in the original paper was the fact that the "*model contains no recurrence and no convolution*" (Vaswani et al., 2017). As we'll see below, this explanation is incomplete, but if one assumed this to be true, it would imply that the autoregressive self-attention also requires positional encodings (we emphasize that this is not true).

Question 3. *Why are positional encodings **nonessential** for **multi-layer autoregressive** Transformers?*

This is because multi-layer autoregressive Transformers are *fully position-sensitive*.

A simple method to check this is to compare the model outputs when we feed two sequences that are permutations of each other. While we could also provide a mathematical proof here, this can better be visualized as in Figure 1: even for this extreme case where we feed two sequences that only differ from each other by a permutation of their two first tokens, the multi-layer model sees different contexts at all positions at the top layer. Essentially, we can build a sequence processor by cascading multiple set processors.

Question 4. *Does the multi-layer autoregressive Transformer language models effectively learn to use positional signals in practice?*

For this question, we refer to Irie et al. (2019) which demonstrated good general performance of multi-layer autoregressive Transformer language models without PEs and provided visualization of attention weights (see figures in Irie et al. (2019)). They reported that, interestingly, the first attention layer mainly attends to the new input, while the second layer uniformly attends to the context. Uniform attention in early layers is intuitively good as it allows the model to grasp all the available context, which is crucial to distinguish similar sequences (as illustrated in Figure 1).

Finally, being non-essential does not imply that some sophisticated extra positional encodings may not improve Transformer language models, we discuss corresponding references in Sec. 5.

Question 5. *Why are positional encodings **needed** for **one-layer autoregressive** self-attention?*

This is also well illustrated in Figure 1 by looking at the first layer. Depending on the specific permutation, one-layer model's outputs at some positions are sensitive to the input permutation, but the output at the last position (when the entire sequence is seen) is the same for any permutations; implying that they are not *fully position-sensitive*.

4 An Intriguing Linear Transformer Case

Here we discuss an *intriguing* case of linear Transformers (Katharopoulos et al., 2020; Schmidhuber, 1992; Schlag et al., 2021). One representative example of linear Transformers can be obtained by simply removing $\mathrm{softmax}$ in Eq. 4. The resulting model can be equivalently expressed as the following fast weight programmer (see Appendix A):

$$\boldsymbol{q}_t, \boldsymbol{k}_t, \boldsymbol{v}_t = \boldsymbol{W}_q \boldsymbol{x}_t, \boldsymbol{W}_k \boldsymbol{x}_t, \boldsymbol{W}_v \boldsymbol{x}_t \tag{1}$$

$$\boldsymbol{W}_t = \boldsymbol{W}_{t-1} + \boldsymbol{v}_t \otimes \boldsymbol{k}_t \tag{7}$$

$$\boldsymbol{y}_t = \boldsymbol{W}_t \boldsymbol{q}_t \tag{8}$$

where $\otimes$ denotes outer product and $\boldsymbol{W}_0 = 0$. Because of the state update rule of Eq. 7 giving an impression of "recurrence" (as in the title of Katharopoulos et al. (2020)), it may not be immediately clear if this model requires PEs. In reality, this is not "true recurrence" but a degenerated one with a recurrent transition matrix reduced to identity (for further discussions, we refer to Irie et al. (2021, 2023); Merrill et al. (2024)). Since this model is equivalent to the autoregressive self-attention layer discussed in Sec. 3, it inherits the same properties, i.e., PEs are needed for one-layer models, while they are nonessential for multi-layer models.

5 Literature Review

To the best of our knowledge, Shen et al. (2018) were the first to use non-symetric "attention masks" (Eq. 6) to encode positional information for neural networks whose sequence processing ability solely relies on the attention mechanism. While the above work does not specifically discuss autoregressive language models (LMs), their insights about masking could have been directly extended to answer the question whether positional encodings are needed for autoregressive Transformer LMs.

Irie et al. (2019) empirically demonstrated that multi-layer autoregressive Transformer LMs perform well without positional encodings. To be more specific, the corresponding ablation study was conducted for 12, 24, and 42 layer models; while other deeper models (up to 112 layers) were also trained without PEs. This result was later rediscovered/confirmed by Haviv et al. (2022)[3] and Kazemnejad et al. (2023). Irie et al. (2019) argued that the autoregressive setting itself encodes the positional information due to increasing context over time, which directly connects to Shen et al. (2018)'s argument (while Irie et al. (2019) failed to cite it). Also, while they specifically state that the results are valid for the *multi-layer* models, they did not explicitly discuss the *one-layer* case (see Footnote 1 for further comments). This no-positional encoding scheme has been immediately adopted in other works on speech recognition; e.g., Zeyer et al. (2019) removed PEs from the decoder of their encoder-decoder speech recognizer. A popular open-source speech toolkit, ESPnet (Watanabe et al., 2018) also integrated Transformer LMs without PEs as part of their standard recipe.

Lee et al. (2019) discussed permutation invariance of non-autoregressive self-attention; Tsai et al. (2019) extended this discussion and showed that autoregressive self-attention is not permutation invariant. However, as discussed above, permutation invariance alone is not enough to conclude on the necessity of PEs (as shown in Sec. 3, one-layer autoregressive models are not *permutation invariant* but also not *fully position-sensitive* and require PEs).

Length Generalization. Empirically, whether removal of PEs yields performance improvements depends on the specific setting. Irie et al. (2019) reported general performance gain by removing absolute/sinusoidal PEs for Transformer LMs trained on books, for various numbers of layers. In constrast, Haviv et al. (2022) and Scao et al. (2022) reported slight degradation.

Nevertheless, one of the common benefits of removing PEs is the improved length generalization. Bhattamishra et al. (2020) showed that Transformers without PEs can generalize on certain formal languages with test sequences that are longer than the training ones. Kazemnejad et al. (2023) showed that LMs without PEs yields the best length generalization performance on reasoning-related tasks compared to sophisticated positional encoding methods. Schlag et al. (2021) successfully trained deep linear Transformers without PEs (Sec. 4) by

[3] Many recent papers inaccurately attribute the origin of this result (see Sec. 7 for examples and further discussions).

carrying context across training batches to enable them to process arbitrarily long sequences.

Regarding length generalization of Transformers with *non*-autoregressive self-attention, we refer to, e.g., Csordás et al. (2021, 2022).

Finally, the main scope of our discussion is the (non)necessity of PEs at the conceptual level. In practice, it is often useful to augment Transformers with a certain type of PEs, especially, relative PEs (Su et al., 2024; Shaw et al., 2018; Dai et al., 2019) to improve their performance; designing practically useful positional encodings for Transformers remains an ongoing research. The perspective developed in this work also opens up the question whether explicit PEs could also enhance other sequence processors, e.g., modern linear RNNs (Bradbury et al., 2017; Lei et al., 2018; Li et al., 2018; Balduzzi and Ghifary, 2016; Gu and Dao, 2024; Qin et al., 2023) even when they are inherently capable of encoding positions.

6 Conclusion

We provide a didactic explanation of why positional encodings are nonessential for multi-layer autoregressive Transformers—an explanation that was well-known among the pre-LLM language modeling practitioners but has not been formally published. We also review the literature related to this result in the hope of correcting potential misconceptions and enhancing our collective knowledge.

7 Excursion: Metascience Perspectives

Beyond the technical scope of this work, it is also interesting, through the lens of metascience, to observe how easily the misconception about the origin of the "no-positional encoding" result has propagated in the current machine learning community (c.f. the references in Sec. 5). Many recent papers refer to this no-PE result as "recent" findings. Here are some example quotes for illustration:

- Flamingo: a Visual Language Model for Few-Shot Learning (Alayrac et al., 2022): *"recent work has shown that such disambiguation is still possible implicitly through the causal attention mechanism [36] (Haviv et al. 2022)."*
- Transformers Learn Shortcuts to Automata (Liu et al., 2022): *"Note that removing positional encoding does not mean having no position information, since the use of the causal mask implicitly encodes the position, which is also noted in Bhattamishra et al. (2020) and concurrent work by Haviv et al. (2022)."*
- Challenges and Applications of Large Language Models (Kaddour et al., 2023): *"Surprisingly, Haviv et al. [192] find that causal LLMs without positional encodings are competitive compared to models with positional encodings and accredit this success to the causal attention mask leaking positional information into the model."*
- Code Llama: Open Foundation Models for Code (Roziere et al., 2023): *"Recent work suggests that causal models do not require an explicit encoding of position information (Haviv et al., 2022; Kazemnejad et al., 2023)"*
- A Phase Transition between Positional and Semantic Learning in a Solvable Model of Dot-Product Attention (Cui et al., 2024): *"While some transformers can leverage implicit positional information through causal masks in training (Haviv et al., 2022; Sinha et al., 2022; Kazemnejad et al., 2023)"*

We speculate that this is partially due to the influence of social media advertising the version-1 preprint. More concretely, neither of the recent papers on no-PEs, Haviv et al. (2022) and Kazemnejad et al. (2023), referred to the pre-LLM work that discussed the same result (Sec. 5) in their first version (arXiv v1). Once such versions are widely advertised (to a relatively new LLM community), it seems too late to include the earlier work in a later version after a few months, unless the discovered priority issue and the correction for potential misinformation are equally advertised.

A contrastive/positive example is the case of "recurrent dropout": early versions of Zaremba et al. (2014) did not cite prior work by Pham et al. (2014); but after discovering Pham et al. (2014), Zaremba et al. (2014) updated the paper with the following statement in Page 1: *"Independently of our work, Pham et al. (2013) developed the very same RNN regularization method and applied it to handwriting recognition. We rediscovered this method and demonstrated strong empirical results over a wide range of problems."* (and there used to be a note on their website encouraging people to cite Pham et al. (2014) instead of theirs).

These data points may contribute to future work in metascience and cognitive psychology of scientific referencing.

Limitations

While our literature review reflects the authors' best efforts and knowledge, we acknowledge the possibility that prior work addressing the nonessentiality of positional encodings in autoregressive Transformers may exist.

Also, here we only focused on the specific topic of positional encodings for autoregressive Transformer language models. There are other similar cases, including the discussion on methods to manage/reduce the size of key-value memory storage (the so-called "KV-cache") in autoregressive Transformers (c.f., Irie et al. (2020) and Liu et al. (2023); Ge et al. (2024)); or the strategy to build mixture-of-experts language models by pre-training component/expert language models independently in parallel (c.f., Irie et al. (2018) and Li et al. (2022); Sukhbaatar et al. (2024)). Further discussion is beyond the scope of this work.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 technical report. *Preprint arXiv:2303.08774*.

Mark A. Aizerman, Emmanuil M. Braverman, and Lev I. Rozonoer. 1964. Theoretical foundations of potential function method in pattern recognition. *Automation and Remote Control*, 25(6):917–936.

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. 2019. Character-level language modeling with deeper self-attention. In *Proc. Conference on Artificial Intelligence (AAAI)*, pages 3159–3166, Honolulu, HI, USA.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Proc. Advances in Neural Information Processing Systems (NeurIPS)*.

Jimmy Ba, Geoffrey E Hinton, Volodymyr Mnih, Joel Z Leibo, and Catalin Ionescu. 2016. Using fast weights to attend to the recent past. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 4331–4339, Barcelona, Spain.

Alexei Baevski and Michael Auli. 2019. Adaptive input representations for neural language modeling. In *Int. Conf. on Learning Representations (ICLR)*, New Orleans, LA, USA.

David Balduzzi and Muhammad Ghifary. 2016. Strongly-typed recurrent neural networks. In *Proc. Int. Conf. on Machine Learning (ICML)*, New York City, NY, USA.

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 932–938, Denver, CO, USA.

Satwik Bhattamishra, Kabir Ahuja, and Navin Goyal. 2020. On the ability and limitations of transformers to recognize formal languages. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7096–7116, Virtual only.

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2017. Quasi-recurrent neural networks. In *Int. Conf. on Learning Representations (ICLR)*, Toulon, France.

Peter F. Brown, John Cocke, Stephan A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Rossin. 1988. A statistical approach to language translation. In *Int. Conf. on Computational Linguistics*, pages 71–76, Buffalo, NY, USA.

Tom B Brown et al. 2020. Language models are few-shot learners. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, Virtual only.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. *Preprint arXiv:2303.12712*.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 551–561, Austin, TX, USA.

Róbert Csordás, Kazuki Irie, and Jürgen Schmidhuber. 2021. The devil is in the detail: Simple tricks improve systematic generalization of transformers. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, Punta Cana, Dominican Republic.

Róbert Csordás, Kazuki Irie, and Jürgen Schmidhuber. 2022. The Neural Data Router: Adaptive control flow in Transformers improves systematic generalization. In *Int. Conf. on Learning Representations (ICLR)*, Virtual only.

Hugo Cui, Freya Behrens, Florent Krzakala, and Lenka Zdeborová. 2024. A phase transition between positional and semantic learning in a solvable model of dot-product attention. *Preprint arXiv:2402.03902*.

Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proc. Association for Computational Linguistics (ACL)*, pages 2978–2988, Florence, Italy.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proc. North American Chapter of the Association for Computational Linguistics on Human Language Technologies (NAACL-HLT)*, pages 4171–4186, Minneapolis, MN, USA.

Jeffrey L Elman. 1989. Structured representations and connectionist models. In *Proc. Conference of Cognitive Science Society (CogSci)*, pages 17–25, Ann Arbor, MI, USA.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. 2024. Model tells you what to discard: Adaptive kv cache compression for llms. In *Int. Conf. on Learning Representations (ICLR)*, Vienna, Austria.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proc. Int. Conf. on Machine Learning (ICML)*, Sydney, Australia.

Albert Gu and Tri Dao. 2024. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling (COLM)*.

Adi Haviv, Ori Ram, Ofir Press, Peter Izsak, and Omer Levy. 2022. Transformer language models without positional encodings still learn positional information. In *Proc. Findings of Conf. on Empirical Methods in Natural Language Processing (EMNLP-Findings)*, pages 1382–1390, Abu Dhabi, United Arab Emirates.

Kazuki Irie, Róbert Csordás, and Jürgen Schmidhuber. 2022. The dual form of neural networks revisited: Connecting test time predictions to training patterns via spotlights of attention. In *Proc. Int. Conf. on Machine Learning (ICML)*, Baltimore, MD, USA.

Kazuki Irie, Róbert Csordás, and Jürgen Schmidhuber. 2023. Practical computational power of linear transformers and their recurrent and self-referential extensions. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, Sentosa, Singapore.

Kazuki Irie, Alexander Gerstenberger, Ralf Schlüter, and Hermann Ney. 2020. How much self-attention do we need? Trading attention for feed-forward layers. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6154–6158.

Kazuki Irie, Shankar Kumar, Michael Nirschl, and Hank Liao. 2018. RADMM: Recurrent adaptive mixture model with applications to domain robust language modeling. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6079–6083, Calgary, Canada.

Kazuki Irie, Imanol Schlag, Róbert Csordás, and Jürgen Schmidhuber. 2021. Going beyond linear transformers with recurrent fast weight programmers. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, Virtual only.

Kazuki Irie, Albert Zeyer, Ralf Schlüter, and Hermann Ney. 2019. Language modeling with deep Transformers. In *Proc. Interspeech*, pages 3905–3909, Graz, Austria.

Frederick Jelinek, Lalit Bahl, and Robert Mercer. 1975. Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, 21(3):250–256.

Jean Kaddour, Joshua Harris, Maximilian Mozes, Herbie Bradley, Roberta Raileanu, and Robert McHardy. 2023. Challenges and applications of large language models. *Preprint arXiv:2307.10169*.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proc. Int. Conf. on Machine Learning (ICML)*, Virtual only.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2023. The impact of positional encoding on length generalization in transformers. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, New Orleans, LA, USA.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proc. Int. Conf. on Machine Learning (ICML)*, pages 3744–3753, Long Beach, CA.

Tao Lei, Yu Zhang, Sida I. Wang, Hui Dai, and Yoav Artzi. 2018. Simple recurrent units for highly parallelizable recurrence. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4470–4481, Brussels, Belgium.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A Smith, and Luke Zettlemoyer. 2022. Branch-train-merge: Embarrassingly parallel training of expert language models. *Preprint arXiv:2208.03306*.

Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao. 2018. Independently recurrent neural network (IndRNN): Building a longer and deeper RNN. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5457–5466, Salt Lake City, UT, USA.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *Int. Conf. on Learning Representations (ICLR)*, Toulon, France.

Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. 2022. Transformers learn shortcuts to automata. *Preprint arXiv:2210.10749*.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Łukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. In *Int. Conf. on Learning Representations (ICLR)*, Vancouver, Canada.

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. 2023. Scissorhands: Exploiting the persistence of importance hypothesis for LLM KV cache compression at test time. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, New Orleans, LA, USA.

William Merrill, Jackson Petty, and Ashish Sabharwal. 2024. The illusion of state in state-space models. In *Proc. Int. Conf. on Machine Learning (ICML)*, Vienna, Austria.

Tomás Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Proc. Interspeech*, pages 1045–1048, Makuhari, Japan.

Tomas Mikolov, Stefan Kombrink, Lukas Burget, Jan H Cernocky, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5528–5531, Prague, Czech Republic.

Masami Nakamura and Kiyohiro Shikano. 1989. A study of english word category prediction based on neural networks. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 731–734, Glasglow, UK.

Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2249–2255, Austin, TX, USA.

Vu Pham, Théodore Bluche, Christopher Kermorvant, and Jérôme Louradour. 2014. Dropout improves recurrent neural networks for handwriting recognition. In *Proc. IEEE International conference on frontiers in handwriting recognition*.

Zhen Qin, Songlin Yang, and Yiran Zhong. 2023. Hierarchically gated recurrent neural network for sequence modeling. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, New Orleans, LA, USA.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. [Available Online] : https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. [Available Online] : https://openai.com/index/better-language-models/.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. 2023. Code Llama: Open foundation models for code. *Preprint arXiv:2308.12950*.

Teven Le Scao, Thomas Wang, Daniel Hesslow, Stas Bekman, M. Saiful Bari, Stella Biderman, Hady Elsahar, Niklas Muennighoff, Jason Phang, Ofir Press, Colin Raffel, Victor Sanh, Sheng Shen, Lintang Sutawika, Jaesung Tae, Zheng Xin Yong, Julien Launay, and Iz Beltagy. 2022. What language model to train if you have one million GPU hours? In *Proc. Findings of Conf. on Empirical Methods in Natural Language Processing (EMNLP-Findings)*, pages 765–782, Abu Dhabi, UAE.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. 2021. Linear Transformers are secretly fast weight programmers. In *Proc. Int. Conf. on Machine Learning (ICML)*, Virtual only.

Jürgen Schmidhuber. 1992. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139.

Jürgen Schmidhuber and Stefan Heil. 1994. Predictive coding with neural nets: Application to text compression. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 1047–1054, Denver, CO, USA.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proc. North American Chapter of the Association for Computational Linguistics on Human Language Technologies (NAACL-HLT)*, pages 464–468, New Orleans, Louisiana, USA.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2018. DiSAN: Directional self-attention network for RNN/CNN-free language understanding. In *Proc. AAAI Conf. on Artificial Intelligence*, pages 5446–5455, New Orleans, LA, USA.

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

Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Rozière, Jacob Kahn, Daniel Li, Wen-tau Yih, Jason Weston, et al. 2024. Branch-train-MiX: Mixing expert llms into a mixture-of-experts llm. *Preprint arXiv:2403.07816*.

Ilya Sutskever, James Martens, and Geoffrey E. Hinton. 2011. Generating text with recurrent neural networks. In *Proc. Int. Conf. on Machine Learning (ICML)*, pages 1017–1024, Bellevue, WA, USA.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4344–4353, Hong Kong, China.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proc. Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, Long Beach, CA, USA.

Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. 2018. ESPnet: End-to-end speech processing toolkit. In *Proc. Interspeech*, pages 2207–2211.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. In *Int. Conf. on Learning Representations (ICLR)*, San Diego, CA, USA.

Albert Zeyer, Parnia Bahar, Kazuki Irie, Ralf Schlüter, and Hermann Ney. 2019. A comparison of transformer and LSTM encoder decoder models for asr. In *Proc. IEEE Automatic Speech Recog. and Understanding Workshop (ASRU)*, Sentosa, Singapore.

A Reminder: Derivation of the dual form of linear Transformers

Here we briefly review the derivation (Katharopoulos et al., 2020; Schlag et al., 2021; Ba et al., 2016) connecting the fast weight programmer of Sec. 4 and its attention form (Eqs. 1-4). Starting from Eq. 8, and by using the definition of $\boldsymbol{W}_t$ from Eq. 7, we obtain:

$$\boldsymbol{y}_t = \boldsymbol{W}_t \boldsymbol{q}_t \tag{9}$$

$$= \left(\sum_{\tau=1}^{t} \boldsymbol{v}_\tau \otimes \boldsymbol{k}_\tau \right) \boldsymbol{q}_t \tag{10}$$

$$= \sum_{\tau=1}^{t} \boldsymbol{v}_\tau \boldsymbol{k}_\tau^\top \boldsymbol{q}_t \tag{11}$$

$$= \boldsymbol{V}_t \boldsymbol{K}_t^\top \boldsymbol{q}_t \tag{12}$$

where the definitions of $\boldsymbol{K}_t$ and $\boldsymbol{V}_t$ are as in Sec. 2.

The last equation is effectively Eq. 4 without $\mathrm{softmax}$.

Note that this relation is analogous to the famous *duality* that connects the perceptron to kernel machines (Aizerman et al., 1964; Irie et al., 2022).