Why Do Decoder-only LLMs Need Positional Encoding?

Su Jianlin

September 1, 2024

As is well known, current mainstream LLMs are all Decoder-only models based on Causal Attention (we have also discussed this in "Why are current LLMs all Decoder-only architectures?"). Regarding Causal Attention, many studies have shown that it can achieve non-trivial results without additional positional encoding (referred to as NoPE). However, the fact remains that mainstream Decoder-only LLMs still incorporate additional positional encodings, such as RoPE, ALiBi, etc.

So the question arises: if it is said that positional encoding is not strictly necessary, why do mainstream LLMs still include it? Isn't it better to "keep things simple"? In this article, we will provide the author's perspective from three angles:

1. What is the role of positional encoding for Attention?
2. How does Causal Attention with NoPE implement positional encoding?
3. What are the shortcomings of positional encoding implemented via NoPE?

1 Positional Encoding

In this section, let us first consider the first question: the significance of positional encoding for the Attention mechanism.

During the era when BERT was popular, many works on positional encoding were proposed. I summarized some of them in "Transformer Positional Encodings that Make Researchers Rack Their Brains". Later, in "The Road to Transformer Upgrade: 1. Tracing the Source of Sinusoidal Positional Encoding", we attempted to understand positional encoding from a perspective closer to its principles and obtained a theoretical explanation for the earliest Sinusoidal positional encoding, which directly inspired the subsequent RoPE.

Simply put, the most fundamental role of positional encoding is to **break the permutation invariance of At** What is permutation invariance? In the BERT era, we mainly used bidirectional Attention, whose basic form is:

$$\boldsymbol{y}_n = \boldsymbol{f}(\boldsymbol{q}_n; \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_L) = \frac{\sum_{m=1}^{L} e^{\boldsymbol{q}_n \cdot \boldsymbol{k}_m} \boldsymbol{v}_m}{\sum_{m=1}^{L} e^{\boldsymbol{q}_n \cdot \boldsymbol{k}_m}}, \quad \boldsymbol{k}_n/\boldsymbol{v}_n = \boldsymbol{x}_n \boldsymbol{W}_{k/v} + \boldsymbol{b}_{k/v} \tag{1}$$

Suppose $\sigma_1, \sigma_2, \cdots, \sigma_L$ is any permutation of $\{1, 2, \cdots, L\}$. Permutation invariance means:

$$\boldsymbol{y}_n = \boldsymbol{f}(\boldsymbol{q}_n; \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_L) = \boldsymbol{f}(\boldsymbol{q}_n; \boldsymbol{x}_{\sigma_1}, \boldsymbol{x}_{\sigma_2}, \cdots, \boldsymbol{x}_{\sigma_L}) \tag{2}$$

In plain terms, $\boldsymbol{y}_n$ is independent of the order of the key-value pairs. This does not align with the characteristics of natural language, so we must find a way to break this invariance. To use a database analogy, Attention without positional encoding is like a database without timestamps; retrieval results only depend on the query. Positional encoding is equivalent to tagging database items with timestamps in order, allowing retrieval results to also depend on the order of items.

2 Prior Knowledge

Another role of positional encoding is to incorporate prior knowledge into Attention or to give Attention the ability to learn these prior properties.

For example, the Sinusoidal positional encoding mentioned earlier is an absolute positional encoding generated directly from trigonometric functions, where the similarity between two adjacent position vectors is higher. This implies a prior that adjacent tokens should have similar embeddings. The positional encoding used by BERT is also an absolute positional encoding, but it is randomly initialized and learned as parameters. This means it does not assume similarity but allows the model to learn this property if it deems it necessary.

More popular is relative positional encoding, whose prior assumption is that "relative position is more important than absolute position." Early relative positional encodings usually included a truncation (where relative positions beyond a certain value are treated as the same). The assumption here is that "long-distance relative positions do not need to be as accurate." T5's positional encoding went a step further by processing relative positions in a logarithmic bucket format, achieving the effect of "the further the relative position, the more blurred it becomes." Furthermore, some relative positional encodings directly add priors to the importance of tokens; for instance, ALiBi implies the assumption that, on average, further tokens are less important (long-range decay).

Models such as RNNs and CNNs essentially integrate the prior that "closer tokens are more important" into their architecture, allowing them to function without positional encoding and reducing complexity to linear. However, priors are human-made and biased—or more bluntly, not accurate enough. Currently, it seems the goal of LLMs is to surpass humans rather than just imitate them. This explains why mainstream architectures use Attention: because the architectural priors are fewer, meaning there are fewer human biases and misconceptions, thus the ceiling is higher.

3 Unidirectional Attention

After understanding the role of positional encoding, let us consider how NoPE works, or to what extent it can achieve the roles of positional encoding mentioned above.

As stated in the previous two sections, bidirectional Attention possesses permutation invariance and thus requires positional encoding to break it. Therefore, NoPE is not suitable for bidirectional Attention. Its prerequisite is unidirectional Attention, or Causal Attention:

$$\boldsymbol{y}_n = \boldsymbol{f}(\boldsymbol{q}_n; \boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_L) = \frac{\sum_{m=1}^{n} e^{\boldsymbol{q}_n\cdot \boldsymbol{k}_m}\boldsymbol{v}_m}{\sum_{m=1}^{n} e^{\boldsymbol{q}_n\cdot \boldsymbol{k}_m}}, \quad \boldsymbol{k}_n/\boldsymbol{v}_n = \boldsymbol{x}_n\boldsymbol{W}_{k/v} + \boldsymbol{b}_{k/v} \tag{3}$$

The difference between this and the bidirectional Attention in Equation (1) is simply that the upper limit of the summation is changed from L to n. From this, it can be seen that it is similar to a cumsum, and the result depends on the order of $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_L$. In other words, it does not possess permutation invariance by itself. Therefore, the combination of "Causal + NoPE" does not, in principle, require positional encoding to achieve non-trivial results (non-trivial meaning the performance is in the same league as those with positional encoding).

The paper that first pointed out this conclusion should be "Transformer Language Models without Positional Encodings Still Learn Positional Information". Of course, this mainly refers to the author being the first to announce this conclusion in a formal "experiment + paper" manner. In fact, to my knowledge, this conclusion was already taken for granted by many before this paper. Additionally, subsequent works like "The Impact of Positional Encoding on Length Generalization in Transformers" and "Length Generalization of Causal Transformers without Position Encoding" explored the length generalization capabilities of NoPE.

4 Identifying Position via Variance

Furthermore, through what mechanism does "Causal + NoPE" identify positional information? We can gain insight through a minimalist example.

Intuitively, $\boldsymbol{y}_n$ defined in Equation (3) is the (weighted) average of n vectors $\boldsymbol{v}$, $\boldsymbol{y}_{n+1}$ is the (weighted) average of $n+1$ vectors $\boldsymbol{v}$, and so on. We can first try the simplest case—a uniform distribution—by considering the following Attention matrix:

$$A = \begin{pmatrix} 1 & & & & & \\ \frac{1}{2} & \frac{1}{2} & & & & \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & & & \\ \vdots & \vdots & \vdots & \ddots & & \\ \frac{1}{n} & \frac{1}{n} & \cdots & \cdots & \frac{1}{n} & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \tag{4}$$

Under this assumption, we have:

$$\boldsymbol{y}_n = \frac{1}{n}\sum_{m=1}^{n} \boldsymbol{v}_m \tag{5}$$

Then, we assume that each component of each $\boldsymbol{v}$ is independently and identically sampled from the same distribution with "mean 0 and variance σ^2." Under this assumption, we can calculate the mean and variance of $\boldsymbol{y}_n$:

$$\frac{1}{d}\sum_{i=1}^{d} \boldsymbol{y}_{n,i} \approx \mathbb{E}[\boldsymbol{y}_{n,i}] = \mathbb{E}\left[\frac{1}{n}\sum_{m=1}^{n} \boldsymbol{v}_{n,i}\right] = \frac{1}{n}\sum_{m=1}^{n} \mathbb{E}\left[\boldsymbol{v}_{n,i}\right] = 0 \tag{6}$$

$$\frac{1}{d}\sum_{i=1}^{d} \boldsymbol{y}_{n,i}^2 \approx \mathbb{E}[\boldsymbol{y}_{n,i}^2] = \mathbb{E}\left[\left(\frac{1}{n}\sum_{m=1}^{n} \boldsymbol{v}_{n,i}\right)^2\right] = \frac{1}{n^2}\sum_{m=1}^{n} \mathbb{E}\left[\boldsymbol{v}_{n,i}^2\right] = \frac{\sigma^2}{n} \tag{7}$$

The second equation is actually the "MS (Mean Square)" in RMS Norm. It can be seen that it is related to the position n. Since the mean is zero, the MS is also equivalent to the variance. From this, we conclude that "Causal + NoPE" actually hides positional information within the variance of the components of $\boldsymbol{y}$, or equivalently, within the ℓ_2 norm of $\boldsymbol{y}$. Of course, readers might question the assumptions of this conclusion. Indeed, these two assumptions at most apply to the model at initialization, but they are sufficient to "grasp" the principle of how NoPE identifies position: **the intuitive difference between each $\boldsymbol{y}_n$ is the number of $\boldsymbol{v}_m$ being averaged, and the most direct variable resulting from different quantities of averaging is the variance.**

The same conclusion appears in the paper "Latent Positional Information is in the Self-Attention Variance of Transformer Language Models Without Positional Encodings", where the authors performed further verification on pre-trained NoPE models, confirming the universality of this conclusion.

5 Shortcomings

Let us summarize the results so far: First, in the first two sections, we summarized the two roles of positional encoding—the primary role is to break the permutation invariance of Attention, and the secondary is to inject certain priors. Then we showed that Causal Attention itself does not possess permutation invariance, so it "in principle" does not need positional encoding (NoPE). Finally, we found that NoPE primarily expresses positional information through the variance of the hidden state vectors.

Now back to the question in the title: why do Decoder-only models based on Causal Attention usually still add positional encoding? The answer is what we just mentioned—Causal Attention "in principle" does not need positional encoding. "In principle" usually means "it can get by, but it's not good enough." Simply put, while NoPE is okay, adding positional encoding is better.

Why is that? This brings us back to "NoPE expressing positional information through vector variance." This is equivalent to saying that $\boldsymbol{y}_n$ is obtained by multiplying some vector $\boldsymbol{z}_n$ (which lacks positional information) by some scalar function $p(n)$ related to position n. This implies:

1. NoPE implements something similar to multiplicative absolute positional encoding, and it only compresses positional information into a single scalar. Therefore, this is a very weak form of positional encoding.

2. A single scalar can represent limited information. When the input length increases, the positional encoding becomes increasingly compact and difficult to distinguish. For example, in the minimalist case where $p(n) \sim \frac{1}{\sqrt{n}}$, when n is large enough, $\frac{1}{\sqrt{n}}$ and $\frac{1}{\sqrt{n+1}}$ are almost indistinguishable, meaning it cannot distinguish between position n and $n+1$.

3. The mainstream view is that relative positional encoding is more suitable for natural language. Since NoPE implements absolute positional encoding, its efficiency is naturally inferior to supplementing the model with additional relative positional encoding.

4. NoPE neither adds priors such as long-range decay to the model nor seems to give the model the ability to learn such priors. When the input length is large enough, problems like attention dispersion may occur.

In summary, NoPE may suffer from insufficient positional resolution, lower efficiency, and attention dispersion for long texts. Therefore, even for Decoder-only models, we still need to supplement them with additional positional encoding (especially relative positional encoding) to improve upon these various shortcomings.

Of course, these analyses are mainly directed at Single-Head Attention. In fact, even if the positional information for each head is only a scalar, with the support of Multi-Head and Multi-Layer structures, the total positional information becomes a fairly substantial vector. Thus, NoPE is not actually that bad; it's just that adding positional encoding makes things better, as it allows the LLM itself to focus more on overall reasoning capabilities rather than spending effort on reproducing capabilities that positional encoding can already provide.

6 Summary

Although some works have shown that Decoder-only models without positional encoding can achieve decent results, mainstream LLMs still incorporate additional positional encoding. This article has attempted to provide an interpretation of this phenomenon.