Neural Networks Exponentiation-based LLMs' Distilation

Introduction

Inspired by polynomial neural networks (PNNs) and how multihead attention mechanisms change the game for natural language generation models (LLMs), we proposed a exponentiation approach which aims to capture rich features without the computational cost tradionally used in LLMs. Traditional multihead attention is defined by several parallel networks (heads: H_i) computed as:

$$H_i \stackrel{\text{def}}{=} softmax\left(\frac{\mathbf{Q}_i\mathbf{K}_i^T}{\sqrt{d_k}}\right)\cdot\mathbf{V}_i \tag{1}$$

Where:

$$\begin{aligned}\mathbf{X} &\in \mathbb{R}^{m\times n}: \text{ Input matrix}\\ \mathbf{Q}_i=\mathbf{X}\mathbf{W}_{\mathbf{Q}_i} &\in \mathbb{R}^{n\times d_k}: \text{ query matrix}\\ \mathbf{K}_i=\mathbf{X}\mathbf{W}_{\mathbf{K}_i} &\in \mathbb{R}^{n\times d_k}: \text{ key matrix}\\ \mathbf{V}_i=\mathbf{X}\mathbf{W}_{\mathbf{V}_i} &\in \mathbb{R}^{n\times d_v}: \text{ value matrix}\end{aligned} \tag{2}$$

Looking at the expression $(\mathbf{X}\mathbf{W}_{Q_i})\ \left(\mathbf{W}_{K_I}^T\mathbf{X}^T\right)$, it can be noticed that it is an exponentiation operation of the matrix product. Thus, because $H_i\in\mathbb{R}^{n\times d_v}$ and $softmax$ applied to rows transform an $m\times n$ matrix into a $m\times 1$ it follows:

def 1. Let $\mathbf{A}\in\mathbb{R}^{m\times p}$ and $\mathbf{B}\in\mathbb{R}^{p\times n}$ such that $\mathbf{C}=\mathbf{AB}\in\mathbb{R}^{m\times n}$. In order to preserve dimensions, this means that, if $\mathbf{C}\in\mathbb{R}^{m\times n}$ then $\mathbf{C}^n\in\mathbb{R}^{m\times n}$, where $\mathbf{C}^n$ is the n-th power of $\mathbf{C}$. It follows that:

$$\begin{aligned}&\mathbf{C}^0 \stackrel{\text{def}}{=} I_{m\times n}\\ &\mathbf{C}^1 \stackrel{\text{def}}{=} (\mathbf{AB})^1=(\mathbf{AB})\\ &\mathbf{C}^2 \stackrel{\text{def}}{=} \mathbf{C}^1(\mathbf{C}^1)^T=(\mathbf{AB})(\mathbf{B}^T\mathbf{A}^T)\end{aligned} \tag{3}$$

$\therefore$ recursively $(\mathbf{AB})^n \stackrel{\text{def}}{=} \prod_{i=0}^{n} f(\mathbf{AB},n)$ where:

$$f(\mathbf{AB},n)=\begin{cases}I_{m\times n} & , \quad n=0\\ (AB) & , \quad n \bmod 2=0 \text{ and } n\neq 0\\ (AB)^T & , \quad n \bmod 2\neq 0\end{cases} \tag{4}$$

remark 1. The Exponential matrix can be calculated using taylor series expansion as $e^{\mathbf{X}}=I+\sum_{k=1}^{\infty}\frac{\mathbf{X}^k}{k!}$ and can be applied for for non-square matrix product as:

$$e^{\mathbf{XW}}=I_{m\times n}+\sum_{k=1}^{\infty}\frac{\mathbf{X}^k}{k!} \tag{5}$$

for $\mathbf{X}\in\mathbb{R}^{m\times p}$ and $\mathbf{W}\in\mathbb{R}^{p\times n}$. In addition, attention mechanisms are often based on the $softmax$ function, described as $\sigma(\mathbf{z})_i=e^{z_i}/\sum_{j=1}^{k}e^{z_j}$ for some $\mathbf{z}\in\mathbb{R}^k$ where $k>1$. Then, mathematically, H_i is the product of the $softmax$ function of a series of exponential-related operations by a matrix product $(\mathbf{X}\mathbf{W}_{V_i})$, which is what we are looking to do by defining the exponential of a matrix product. Our hypothesis is that, by recurring to a formal definition of the exponential of non-square matrix we can capture the escence of traditional attention mechanisms but, reducing the amount of parameters to required to obtain LLMs comparable results.

$\therefore$ We can extend the exponential of a matrix product to avoid explicit exponential matrix calculation by limiting the expansion to $n\leq\infty$:

$$e^{\mathbf{XW}}\approx I_{m\times n}+\sum_{k=1}^{n}\frac{(\mathbf{XW})^k}{k!} \tag{6}$$

and combining infinite number of feature dimensions using a radial basis function representation parameterized by $\sigma=1$. Then, we can define the attention like mechanism as:

$$\phi(\mathbf{X}, \mathbf{W}) \stackrel{\text{def}}{=} \exp\left(-\frac{(\mathbf{XW})^2}{2}\right) \tag{7}$$

This allows us to capture attention-like attention mechanisms ny reducing the number of parameters required, and computational cost.

$$H_i = \phi(\mathbf{X}, \mathbf{W}) \tag{8}$$

The output dimension of $\phi(\mathbf{X}, \mathbf{W}) \in \mathbb{R}^{m \times n}$ matches the matrix product dimensions, analogous to concatenating multiple attention heads in traditional multi-head attention. To better emulate standard attention mechanisms and maintain model dimensions, we can incorporate a linear projection layer:

$$H = \phi(\mathbf{X}, \mathbf{W}) \cdot \mathbf{W}_O \quad \text{where} \quad \mathbf{W}_O \in \mathbb{R}^{n \times d_{model}} \tag{9}$$

Analyzing the derivative with respect to the input reveals important gradient properties:

$$\frac{\partial \phi}{\partial(\mathbf{XW})} = -(\mathbf{XW}) \cdot e^{\left(-\frac{(\mathbf{XW})^2}{2}\right)} \tag{10}$$

The gradient magnitude is bounded by $|\frac{\partial \phi}{\partial(\mathbf{XW})}| \leq \frac{1}{\sqrt{e}}$, preventing exploding gradients. The gradient magnitude bound of $\frac{1}{\sqrt{e}}$ is concluded by analyzing the critical points of the derivative function. We begin with the defined radial basis function:

$$\phi(z) = e^{\left(-\frac{z^2}{2}\right)}$$

where $z = \mathbf{XW}$ is a scalar argument for the purpose of this derivation. The derivative with respect to z is:

$$\frac{d\phi}{dz} = -z \cdot e^{\left(-\frac{z^2}{2}\right)}$$

To find the maximum magnitude of this derivative, we consider the absolute value and find its maximum:

$$g(z) = \left|\frac{d\phi}{dz}\right| = |z| \cdot e^{\left(-\frac{z^2}{2}\right)}$$

Since $g(z)$ is an even function, we can find the maximum for $z \geq 0$:

$$g(z) = z \cdot e^{\left(-\frac{z^2}{2}\right)}$$

We find the critical point by taking the derivative and setting it to zero:

$$g'(z) = e^{\left(-\frac{z^2}{2}\right)} - z^2 \cdot e^{\left(-\frac{z^2}{2}\right)} = (1 - z^2) \cdot e^{\left(-\frac{z^2}{2}\right)} = 0$$

This equation holds when $1 - z^2 = 0$, which gives $z = \pm 1$. Substituting $z = 1$ back into $g(z)$ yields the maximum value:

$$g(1) = (1) \cdot e^{\left(-\frac{(1)^2}{2}\right)} = e^{\left(-\frac{1}{2}\right)} = \frac{1}{\sqrt{e}}$$

Therefore, the magnitude of the gradient is bounded by $\frac{1}{\sqrt{e}}$:

$$\left|\frac{d\phi}{dz}\right| \leq \frac{1}{\sqrt{e}} \tag{11}$$

However, for very large $|\mathbf{XW}|$ values, the gradient can vanish. To mitigate this, we can make σ a learnable parameter:

$$\phi(\mathbf{X}, \mathbf{W}) \stackrel{\text{def}}{=} e^{\left(-\frac{(\mathbf{XW})^2}{2\sigma^2}\right)} \tag{12}$$

This allows the model to automatically adjust the sensitivity and gradient scale, balancing between preventing vanishing gradients for large inputs and maintaining sufficient gradient signal for optimization.