cluster-attention

Tristan Peat

September 2024

1 Introduction

Transformers have become ubiquitous in deep learning [3], with many attempts to improve the original transformer block structure beyond quadratic space and time complexity. However, these modifications largely retain the core self-attention mechanism, a scaled dot product with inherent quadratic costs.

A key strength of transformers is their ability to capture relationships between inputs by entangling them in the latent space. In the self-attention mechanism, each object x_i has a query $Q(x_i)$, which it uses to check compatibility with the key $K(x_j)$ of another object x_j. Compatibility between x_i and x_j is determined by the inner product $\langle Q(x_i), K(x_j) \rangle$. A high inner product indicates a strong match, leading x_i to retrieve x_j's value $V(x_j)$. The representation u_i is then formed by summing up values of objects weighted by their compatibility with x_i.

However, it has been shown that the self-attention mechanism, when used in isolation, converges to a rank-1 solution—where all rows become identical due to the softmax operation—in cubic time [1]. This limitation has inspired fine-tuning methods such as LoRA, which improve efficiency by leveraging low-rank adaptations. Skip connections and feedforward networks are crucial in preventing this rank collapse and enhancing the overall representational power of transformers.

Furthermore, under specific conditions, self-attention has been proven to converge to well-defined geometric structures, suggesting an underlying clustering behavior in the attention process [2]. These studies consider transformer parameters, such as weight matrices and LayerNorm coefficients γ, β from LayerNorm) as time dependent and analyze how these structures evolve as $t \to \infty$.

2 Hypothesis

I hypothesize that over time gradient descent and self-attention work together to push similar objects in $x \in \mathbb{R}^{n \times d}$ together, thereby forming clusters and that there is alternative architecture component that can more effectively find these clusters without repeated $O(n^2)$ operations.

For seamless integration with other models and specialized output heads in deep networks, it would be ideal if this component could be easily swapped in place of the transformer block—or specifically, the self-attention mechanism—without compromising compatibility.

3 Math

Note: This section is intended to outline the equations I might use. I am currently unsure how to begin implementing this approach.

A transformer block is a parameterized function class $f_\theta : \mathbb{R}^{n\times d} \to \mathbb{R}^{n\times d}$. If $x \in \mathbb{R}^{n\times d}$, then $f_\theta(x) = z$, where

$$Q^{(h)}(x_i) = W_{h,q}^T x_i, \quad K^{(h)}(x_i) = W_{h,k}^T x_i, \quad V^{(h)}(x_i) = W_{h,v}^T x_i, \quad W_{h,q}, W_{h,k}, W_{h,v} \in \mathbb{R}^{d\times k} \tag{1}$$

$$\alpha_{i,j}^{(h)} = \text{softmax}_j\left(\frac{\langle Q^{(h)}(x_i), K^{(h)}(x_j)\rangle}{\sqrt{k}}\right) \tag{2}$$

$$u_i' = \sum_{h=1}^{H} W_{c,h}^T \sum_{j=1}^{n} \alpha_{i,j}^{(h)} V^{(h)}(x_j), \quad W_{c,h} \in \mathbb{R}^{k\times d} \tag{3}$$

$$u_i = \text{LayerNorm}(x_i + u_i'; \gamma_1, \beta_1), \quad \gamma_1, \beta_1 \in \mathbb{R}^d \tag{4}$$

$$z_i' = W_2^T \text{ReLU}(W_1^T u_i), \quad W_1 \in \mathbb{R}^{d\times m}, \quad W_2 \in \mathbb{R}^{m\times d} \tag{5}$$

$$z_i = \text{LayerNorm}(u_i + z_i'; \gamma_2, \beta_2), \quad \gamma_2, \beta_2 \in \mathbb{R}^d \tag{6}$$

I might parameterize the time dependence as follows:

$$Q^{(h)}(x_i, t) = W_{h,q}(t)^T x_i, \quad K^{(h)}(x_i, t) = W_{h,k}(t)^T x_i, \quad V^{(h)}(x_i, t) = W_{h,v}(t)^T x_i$$

$$\alpha_{i,j}^{(h)}(t) = \text{softmax}_j\left(\frac{\langle Q^{(h)}(x_i, t), K^{(h)}(x_j, t)\rangle}{\sqrt{k}}\right) \tag{2'}$$

I would like to study $\alpha_{i,j}^{(h)}(t)$ and $W_{h,q}(t), W_{h,k}(t), W_{h,v}(t)$ as $t \to \infty$ to deeper explore the clustering phenomenon:

$$\lim_{t\to\infty} \alpha_{i,j}^{(h)}(t) = \text{softmax}_j\left(\frac{\langle Q^{(h)}(x_i, t), K^{(h)}(x_j, t)\rangle}{\sqrt{k}}\right)$$

I'd also like to show as $t \to \infty$,

$$\|\alpha_{i,j}^{(h)}(t) - \alpha_{i,j}^{(h)}(\infty)\| \le \epsilon, \quad \forall\, i, j, h, \text{ for some } \epsilon > 0$$

4 Open Questions

1. How can I simplify the problem for mathematical convenience without destroying the transfer-ability to an industry setting?

2. What mathematical tools are available to prove the existence of clusters?

3. If clusters are proven to exist, how do I then prove how many clusters there are?

4. Incorporating a clustering algorithm into an existing PyTorch model is challenging because clustering algorithms assign labels to inputs, while PyTorch requires a differentiable forward and backward pass compatible with its computational graph. How should I address this issue?

5. If I do design a new component that takes in $x \in \mathbb{R}^{n \times d}$ and outputs $y \in \mathbb{R}^{n \times d}$ which has clustered similar objects in x closer together, how can I quantify how much worse this approach will be than the transformer block?

References

[1] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth, 2023.

[2] Borjan Geshkovski, Cyril Letrouit, Yury Polyanskiy, and Philippe Rigollet. The emergence of clusters in self-attention dynamics, 2024.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.