Preliminary Research Idea (7): "Elastic" Attention Mechanism and Sinkhorn Computation

Professor Xu Yida
Machine Learning Knowledge Disseminator

In attention mechanisms, after computing QK^T, the next step is typically row-wise normalization: Row-Wise Softmax. This often leads to mode collapse, where only a few tokens receive most of the attention, while others receive nearly zero. However, if we also normalize the columns, we can obtain a Doubly Stochastic Matrix (Sinkhorn), which implies that the total attention a token emits (to other tokens) and receives (from other tokens) is the same (even if individual allocations differ). This can be achieved by solving the following equation:

Let $C = QK^T \in \mathbb{R}^{N \times N}$ be the cost matrix (or negative score matrix). We seek a matrix $P \in \mathbb{R}^{N \times N}$ that minimizes the total cost minus an entropy term:

$$\min_{P \geq 0} \underbrace{\sum_{i,j} P_{ij} C_{ij}}_{\text{Total Cost}} - \underbrace{\epsilon H(P)}_{\text{Entropy}} \tag{1}$$

Needless to say, everyone knows that entropy $H(P) = -\sum_{i,j} P_{ij} \log P_{ij}$. By modifying the constraint set and the temperature parameter ϵ, we can obtain three different cases.

The Unified Optimization Problem

Case 1: Row Softmax (Independent Selection)

- **Constraints:** Row sums must be 1. Columns are unconstrained.

$$\sum_j P_{ij} = 1, \quad \forall i \tag{2}$$

- **Temperature:** $\epsilon > 0$ (finite smoothing).

Since columns are unconstrained, the optimization problem decouples across rows. The Lagrangian dual gives the standard Softmax function:

$$P_{ij} = \frac{\exp(-C_{ij}/\epsilon)}{\sum_k \exp(-C_{ik}/\epsilon)} \tag{3}$$

Case 2: Doubly Stochastic Matrix (Sinkhorn)

- **Constraints:** Row sums and column sums must both be 1 (Birkhoff Polytope).

$$\begin{aligned} \sum_j P_{ij} = 1 \quad \forall i \\ \sum_i P_{ij} = 1 \quad \forall j \end{aligned} \tag{4}$$

- **Temperature:** $\epsilon > 0$ (finite smoothing).

Due to the presence of column constraints, the choice for each row is coupled. The form of this solution is a matrix scaling problem, which can be solved using the Sinkhorn-Knopp algorithm. The optimal transport scheme is:

$$P_{ij} = u_i \exp(-C_{ij}/\epsilon) v_j \tag{5}$$

where u and v are non-negative scaling vectors, used to ensure that the sums of rows and columns are 1.

Case 3: Permutation Matrix (Hard Assignment)

- **Constraints:** Row sums and column sums must both be 1.
- **Temperature:** $\epsilon \to 0$ (zero smoothing).

When the temperature approaches zero, the entropy term disappears. This problem converges to a Linear Assignment Problem (i.e., minimum cost perfect matching). The probability mass hardens to binary:

$$P = \lim_{\epsilon \to 0} \text{Sinkhorn}(C, \epsilon) \tag{6}$$

Mathematically, $P_{ij} \in \{0, 1\}$ represents a specific permutation (i.e., tokens swapping positions). This is the result solved by the Hungarian Algorithm.

Proposed Modified Iterative Update Rule

My idea is whether it's possible to devise a modified iterative update rule. Standard Sinkhorn is $v \leftarrow 1/(K^T u)$, while in our "elastic" version, the update rule becomes a power function form.

Let $K_{ij} = \exp(-C_{ij}/\epsilon)$. We maintain two scaling vectors u (rows) and v (columns).

Approximate Algorithm Flow:

1. **Initialization:** $u^{(0)} = \mathbf{1}, v^{(0)} = \mathbf{1}$

2. Iterative Update:

Row Update (Row Update - Maintaining Hard Constraint):

We need to ensure that the row sums of Attention are always 1 (this is a basic requirement for Transformer, ensuring output within the convex hull).

$$u^{(t)} = \frac{1}{K v^{(t-1)}} \tag{7}$$

(Note: This is equivalent to the standard Softmax operation)

Column Update (Column Update - Adaptive Soft Constraint):

Here we introduce λ. Instead of forcing column sums to be 1, we "partially" correct them based on the gap between the current column sums and 1.

$$v^{(t)} = \left(\frac{1}{K^T u^{(t)}} \right)^{\lambda} \odot (v^{(t-1)})^{1-\lambda} \tag{8}$$

Calculate Final Matrix:

$$P_{ij} = u_i K_{ij} v_j \tag{9}$$

Additionally, anneal $\epsilon \to 0$ during the optimization process. Or more creative modifications.