Machine Learning for Molecular Physics WiSe 24/25

Exercise 05 Nov 19, 2024

1 Theoretical background

Attention mechanism equation:

$$\text{Attention (Q, K, W)} = \text{softmax}\left[\frac{QW^Q(KW^K)^T}{\sqrt{d_{model}}}\right]VW^V.$$

K, Q, V all have same dimensionality and K, Q, V $\epsilon\ \mathbb{R}^{[n_{batch} * \ d_{seq} * \ d_{model}]}$, where n_batch is the batch size, d_seq the sequence length and d_model the feature dimension. Also, $w^k, w^Q, w^V \in \mathbb{R}^{d\_model \, * \, d\_model}$.

Attention mechanism:

Attention mechanism focuses on specific parts of input when generating output. Instead of treating all inputs equally, model assigns different weights to diff inputs, depending on their relevance.

1. What is the meaning of all appearing quantities and how are they used in the attention mechanism?

   - **Input values: Q, K, V matrices** represent queries, keys and values.
     - **Queries** represent the part of the input seeking information
     - **Keys** act like labels to help the query find relevant information.
     - **Values** contain the actual data or features that query retrieve based on the keys.
     - **All the 3 matrices are derived from the input sequences.**
   - **Weights W_Q/K/V**
     - **W_Q/k/V** are the learnable weight matrices that transform the input into Q, K, V spaces.
   - **Dot product** ($QW^Q(KW^K)^T$) computes a similarity score between each query and all keys, the higher value shows that the query and key are more aligned or relevant.
   - **Scaling** ($\sqrt{d_{model}}$) are used to scale the dot product to avoid reaching larger values.
   - **V (weighted values of vector sum)**, where weights are the attention scores from softmax, softmax converts the similarity scores into a probability distribution and the sum of weights across keys for a given query is equal to 1.

2. What is the attention map (also known as attention weights)? What is its dimension? How can we interpret it?

   Attention Map/ Attention weights:
   - A matrix representing how much each query attends to every key. If the sequence has d_seq elements, the attention map has dimensions (d_seq, d_seq).
   - Every row corresponds to a query, n column to a key. Each value shows the relevance of the corresponding key for query.

3. What is self-attention? How does it relate to Eq.?

   - Self-attention is a specific case where Q, K and V are derived from input sequence. In this case each position in sequence is related to other positions,
   - e.g., For each position in a sequence, we compute how much attention to pay to each other position using Q and K, retrieve relevant information from all other position using V.

- It relates the relationship between sequence elements regardless of their distance and doesn't rely on fixed size windows like convolutions or RNNs.

4. The attention update in Eq. is sometimes referred to as a fully-connected graph neural network. Why?
   - **Fully connected graph:**
     - In an attention mechanism, every position/node can attend to every other position/node forming a fully connected graph/grid. Here, edge weights are attention scores derived from $QW^Q(KW^K)^T$product.
     - For node updates, each node aggregates the information from all other nodes using weighted values V like a gnn.

2 Implementation

2.2 Attention block

Why do we use a residual update?

We use the following residual update in forward pass within the class SelfAttentionBlock as follows:

x 🡐 x + Attention (x, x, x)

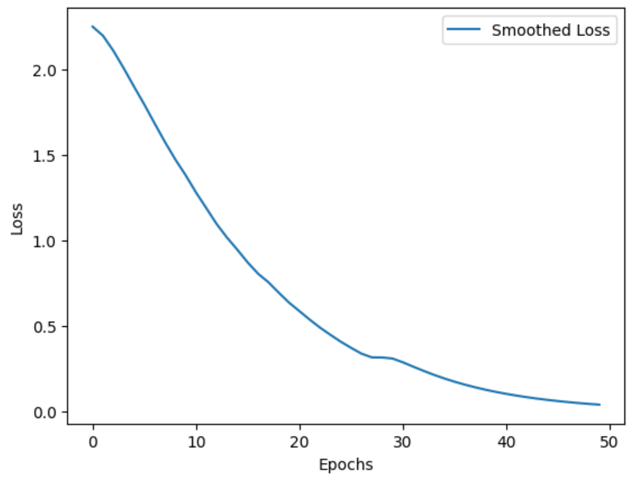
x 🡐 x + MLP(x)

Firstly, it stabilises the training and prevents the model from learning unstable updates to the input and improves the gradient flow to propagate directly to earlier layers without creating other deep networks or even being vanished and preserves the original input features are always part of the models' representations.

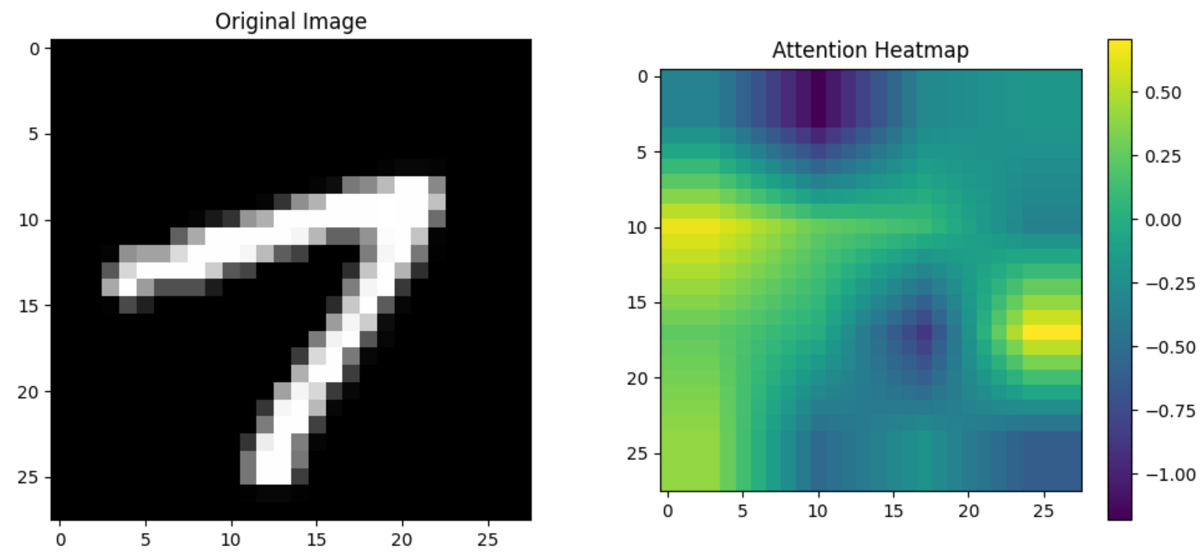
3 Training

3.2 Fitting

We are using a model of visual transformer with 5 layers and a dimension of 64. In my model parameters, I considered dimensionality (d_model) = 64, with a learning rate (lr) of 1×10−3. For convergence it could have been better to take higher epochs, but I considered no of epochs (NUM_EPOCHS) as 50 while training with lr = 1×10−3 and it took a lot of time for training, approximately 15 minutes. The total trainable parameters were 140618 and accuracy was about 71.51% which is significantly low.



The attention map obtained for the test image as follows:


Original Image
Attention Heatmap