Practical No. 4

Tomasz Siłkowski
ts407106@students.mimuw.edu.pl

29 kwietnia 2023

2 Attention Exploration

a) Attention input for identity

For attention to approximately copy one of the value vectors v_i, $k_i^T q$ would have to significantly greater than $k_j^T q$ for any $j \neq i$.

b) Query input for attention to return average

$$q = C(k_a + k_b), \text{where } C \gg 0$$

c) Drawbacks of single-headed attention

1.

$$q = C(\mu_a + \mu_b), \text{where } C \gg 0$$

Because covariance matrices for all keys k are identity matrix multiplied by a vanishingly small constant, we know that keys (random variables) k_a and k_b are not correlated, therefore independent. That means μ_a and μ_b are their best estimators and are also perpendicular to each other, they are the best guess for requested value.

2. To restate the problem, we have $k_a \sim \mathcal{N}(\mu_a, \alpha I + \frac{1}{2}(\mu_a \mu_a^T))$ where α is vanishingly small.
Let $q = C(\mu_a + \mu_b)$ (same as part 1) and let k_b be the key pointing in the same direction as k_a but with different norm. This means:

$$k_i^T q \approx \begin{cases} \varepsilon_a C & \text{for } i = a, \text{ where } \varepsilon_a \sim \mathcal{N}(1, \frac{1}{2}) \\ \varepsilon_b C & \text{for } i = b, \text{ where } \varepsilon_b \sim \mathcal{N}(1, \frac{1}{2}) \\ 0 & \text{otherwise} \end{cases}$$

Because of that, when calculating c:

$$c = \frac{\exp(\varepsilon_a C)}{\exp(\varepsilon_a C) + \exp(\varepsilon_b C)} v_a + \frac{\exp(\varepsilon_b C)}{\exp(\varepsilon_a C) + \exp(\varepsilon_b C)} v_b =$$
$$= \frac{1}{\exp((\varepsilon_a - \varepsilon_b)C)} v_a + \frac{1}{\exp((\varepsilon_b - \varepsilon_a)C)} v_b.$$

Because ε_a and ε_b come from the same distribution, it is equally likely that c will be closer to v_a as to v_b. This means c will be closer to whichever v_i has bigger $|k_i|$ for $i \in \{a, b\}$.

d) Benefits of multi-headed attention

1.

$$q_1 = C_1 \mu_a, \text{where } C_1 \gg 0$$
$$q_2 = C_2 \mu_b, \text{where } C_2 \gg 0$$

2.

$$k_a^T q_1 = C_1 \varepsilon_a$$
$$k_b^T q_2 = C_2 \varepsilon_b$$

Then:

$$c_1 = v_a$$
$$c_2 = v_b$$
$$c = \frac{1}{2}(c_1 + c_2) \approx \frac{1}{2}(v_a + v_b)$$

Basing my judgement on these calculations, I expect output c to be close to the average of v_a and v_b.

e) Key-Query-Value self-attention intuition

1. c_2 approximates vector u_a.
   It's impossible to approximate u_b with c_2 by adding either u_c or u_d to x_2. Any of these vectors will increase in value equally along with u_b. That means there's no way for u_b to dominate the combination.

2.

$$V = u_b u_b^T \circ \frac{1}{||u_b||_2^2} - u_c u_c^T \circ \frac{1}{||u_c||_2^2} = (u_b u_b^T - u_c u_c^T) \circ \frac{1}{\beta^2}$$
$$K = I$$
$$Q = u_d u_a^T \circ \frac{1}{||u_a||_2^2} - u_c u_d^T \circ \frac{1}{||u_d||_2^2} = (u_d u_a^T - u_c u_d^T) \circ \frac{1}{\gamma^2}$$

And to prove that these values work:

$$\begin{aligned} v &= [u_b, 0, u_b - u_c] \\ q &= [u_c, u_d, 0] \\ \forall_{i \in \{1,2,3\}} k_i &= x_i \end{aligned}$$

This implies:

$$\begin{aligned} \alpha_1 &\approx [0, 0, 1] \\ \alpha_2 &\approx [1, 0, 0] \\ c_1 &\approx v_3 = u_b - u_c \\ c_2 &\approx b_1 = u_b \end{aligned}$$

3 Pretraining Transformer-based Generative Model

d) Model with only finetuning

After 75 epochs of finetuning previously untrained model, in evaluation it achieved a score of 0.6% (3 out of 500 correct).
For comparison, naive model (outputting only "London" as an answer) achieved a score of 5.0% (25 out of 500 correct).

f) Fully trained model on CharCorruptedDataset

After 650 epochs of pretraining and 10 epochs of finetuning using Dataset with randomly masked substrings, the model achieved a score of 23.0% (115 out of 500 correct) in evaluation.

g) More computationally efficient transformer variants

1. **Linformer**
   Linformer architecture allows to reduce computational complexity of transformer to $O(n)$, where n is the length of the sequence. It achieves that by approximating attention dot product matrix with a matrix of lower rank, calculated using SVD. Architecture is made even more efficient by substituting SVD with a simple linear projection to reduce the dimensionality.

2. **BigBird**
   BigBird model impreoves efficiency of transformers by limiting attention to number of pairs proportional to length of sequence (therefore of complexity $O(n)$. Specific pairs that are used are: random, sliding window (neighbours of token) and limited global attention (attention with few, globally selected tokens).

3. **Why is BigBird as efficient as full-attention mechanisms?**
   Because mnost of BigBird's selected pairs of attention are universally significant: neighbours, tokens significant due to their position. Additionally included random pairs of attention, allow most of the context to be perceived and therefore the model can learn to weigh them as more or less improtant.

4 Pretrained Models as Source of Knowledge

a) Success of pretrained vs. failure of non-pretrained

At the beginning of fine-tuning, pretrained model has already learned universal relations in data. That extra knowledge allowed, after short fine-tuning, proved useful in comparison to noise that non-pretrained model knew of in the same position.

b) User-facing systems implications

1. False information (halucinating): such models will confidently assert falsities
2. Biased input leads to biased output: these models will be unable to critically assess biases and stereotypes present in the data

c) Generating not yet learned data

When the model will be prompted with a question for a birth place of a person that it wasn't trained for, it will still attempt to answer. Most likely it will chose an answer for a person with similar name or from a general area, from where people with such names come from. This will showcase model's biases and apply it to given prompt.