Practical No. 4

Karol Pustelnik
kp446518@students.mimuw.edu.pl

May 1, 2022

1 Attention exploration

a)

a)

$$c = \sum_{i=1}^{n} v_i \alpha_i$$

$$\alpha_i = \frac{\exp(k_i^T q)}{\sum_{j=1}^{n} \exp(k_j^T q)}$$

To achieve $c \approx v_j$ for some $j \in \{1, \dots, n\}$ it is sufficient to have:

$\forall i \neq j \; \alpha_i \approx 0$ and $\alpha_i \approx 1$ for $i = j$.

for that to hold we must have:

$$\frac{\exp(k_i^T q)}{\sum_{j=1}^{n} \exp(k_j^T q)} \approx 1 \quad \Leftrightarrow$$

$$\exp(k_i^T q) \approx \sum_{j=1}^{n} \exp(k_j^T q) \quad \Leftrightarrow$$

$$\forall i \neq j \; \exp(k_i^T q) \approx 0$$

So $k_j^T q$ must be much greater than any other $k_i^T q$.

$$\Rightarrow \quad k_j^T q >>_2 k_i^T q \quad \forall i$$

b)

b)

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

$$\alpha_a = \frac{1}{2} = \frac{\exp(k_a^T q)}{\sum_{i=1}^{n} \exp(k_i^T q)}$$

$$\alpha_b = \frac{1}{2} = \frac{\exp(k_b^T q)}{\sum_{i=1}^{n} \exp(k_i^T q)}$$

$$\Rightarrow \exp(k_a^T q) >> \exp(k_i^T q) \quad i \neq a, b$$

$$\exp(k_b^T q) >> \exp(k_i^T q) \quad i \neq a, b$$

$$\Rightarrow q = (k_a^T + k_b^T) \cdot c_1, \quad c_1 >> 0$$

Explanations are similiar to a)

c)

c1) $k_i \sim \mathcal{N}(\mu_i, \Sigma_i)$

if $\Sigma_i = \alpha I$ for vanishingly small α, then $k_i \approx \mu_i$ (if $\Sigma_i = 0$, then we would always sample mean)

Similarly to b):

$$q = (u_a + u_b) \cdot C_1, \quad C_1 >> 0$$

c2)

the vector c will be directed closer to the v_a, if $\|k_a\| > \|k_b\|$ or closer to v_b if $\|k_a\| < \|k_b\|$

(the direction of k_a will not change because we add specific component $\frac{1}{2} u_a u_a^T$ that allows changing magnitude, but does not allow big change of direction of k_a.

d)

d)

$$q_1 = u_a \cdot b_1 \quad , \quad b_1, b_2 \geq 0$$

$$q_2 = u_b \cdot b_2$$

$c_1 \approx V_a$ (by the design of q_1)*

$c_2 \approx V_b$ (-11-)

$$c = \frac{1}{2}(c_1 + c_2) \approx \frac{1}{2}(V_a + V_b)$$

So c is the same as in 1).

* Why? Because b_1, b_2 are selected individually for k_a and k_b, so even if $\|k_a\| > \|k_b\|$, then $b_2 > b_1$ will hold to recompensate the difference in norms.

e)

e1)

$$c_2 = \sum_{j=1}^{3} \alpha_{2j} v_j = \sum_{j=1}^{3} \alpha_{2j} x_j$$

$$\alpha_{21} = \frac{\exp(x_1^T x_2)}{\sum_{i=1}^{3} \exp(x_i^T x_2)} = 0$$

$$\alpha_{22} = \frac{\exp(x_2^T x_2)}{\sum_{i=1}^{3} \exp(x_i^T x_2)} \simeq 1$$

$$\alpha_{23} = \frac{\exp(x_3^T x_2)}{\sum_{i=1}^{3} \exp(x_i^T x_2)} \simeq 0$$

So $c_2 = x_2 = u_a$

c_2 will not be able to apprx. u_b even after adding u_d (or u_c) to x_2 because u_i have equal norms and we can't use weights to adjust them.

e 2)

$V_1 = u_b$

$V_1 = V \cdot x_1 = V \cdot (u_a + u_b) = u_b$

observation:

$u_b \cdot u_b^T \cdot u_b = u_b \cdot \|u_b\|^2 \Rightarrow \frac{u_b u_b^T \cdot u_b}{\|u_b\|^2} = u_b$

$u_b \cdot u_b^T \cdot u_a = 0$ (because $u_b \perp u_a$)

So if we take $V = \frac{u_b \cdot u_b^T}{\|u_b\|^2}$

we will have $V \cdot x_1 = u_b$.

What about $V_3 = u_b - u_c$?

$V_3 = V \cdot x_3 = V \cdot (u_c + u_b)$

observation:

$-u_c u_c^T \cdot u_c = -u_c \cdot \|u_c\|^2$

$\Rightarrow \frac{-u_c u_c^T \cdot u_c}{\|u_c\|^2} = -u_c$

and $-u_c u_c^T \cdot u_b = 0$ because $u_c \perp u_b$

So if we choose $V = \frac{u_b u_b^T}{\|u_b\|} - \frac{u_c \cdot u_c^T}{\|u_c\|^2}$

$= \frac{u_b u_b^T - u_c u_c^T}{B^2}$

we will have $v_1 = u_b$, $v_3 = u_b - u_c$

$c_2 \approx u_b$ How to achieve it ?

$$c_2 = \alpha_{21} v_1 + \alpha_{22} v_2 + \alpha_{23} v_3$$

$$\Rightarrow \alpha_{21} \approx 1, \quad \alpha_{22} \approx 0, \quad \alpha_{23} \approx 0$$

$$\alpha_{22} = \frac{\exp(k_2^T q_2)}{\exp(k_1^T q_2) + \exp(k_2^T q_2) + \exp(k_3^T q_2)}$$

in order for α_{22} to be ≈ 0

$k_2^T q_2$ has to be 0. To achieve that, we can just simply set $k_1 = x_1$, $k_2 = x_2$, $k_3 = x_3$ $q_2 = u_d$.

$$\alpha_{22} = \frac{1}{\exp(B) + 2} \approx 0$$ because $B \gg 0$

Simimarly, we need $\alpha_{23} \approx 0$

$$\alpha_{23} = \frac{\exp(k_3^T q_2)}{\exp(k_1^T q_2) + \exp(k_2^T q_2) + \exp(k_2^T q_2)}$$

$\exp(k_3^T q_2)$ -> already = 1

$\exp(k_1^T q_2) = B$

$$\approx 0$$

Now we need to make sure $\alpha_{21} \approx 1$

$$\alpha_{21} = \frac{\exp(k_1^T q_2)}{\exp(k_1^T q_2) + \exp(k_2^T q_2) + \exp(k_3^T q_2)}$$

$$= \frac{\exp(B)}{\exp(B) + 1 + 1} \approx 1$$

$c_1 \approx u_b - u_c$ How to achieve it ?

$$c_1 = \alpha_{11} v_1 + \alpha_{12} v_2 + \alpha_{13} v_3$$

we need $c_1 \approx u_b - u_c$

=> α_{11}, α_{12} has to be ≈ 0

and $\alpha_{13} \approx 1$.

$$\alpha_{11} = \frac{\exp(k_1^T q_1)}{\exp(k_1^T q_1) + \exp(k_2^T q_1) + \exp(k_3^T q_1)}$$

$$\alpha_{12} = \frac{\exp(k_2^T q_1)}{\exp(k_1^T q_1) + \exp(k_2^T q_1) + \exp(k_3^T q_1)}$$

for $k_1^T q_1 = 0$ and $k_2^T q_1 = 0$ we can only set $q_1 = u_c$.

We then get:

$$\alpha_{11} = \frac{1}{\exp(B) + 2} \approx 0$$

$$\alpha_{12} = \frac{1}{\exp(B) + 2} \approx 0$$

Now let's make sure $\alpha_{13} \approx 1$

$$\alpha_{13} = \frac{\exp(k_3^T q_1)}{\exp(k_1^T q_1) + \exp(k_2^T q_1) + \exp(k_3^T q_1)}$$

$$= \frac{\exp(B)}{2 + \exp(B)} \approx 1$$

To summarize:

In order to have $c_1 \approx u_b - u_c$ and $c_2 \approx u_b$

we need:

$$q_1 = u_c, \quad q_2 = u_d, \quad k_i = x_i, \; i \in \{1,2,3\}$$

So what should matrices Q and K look like?

K is simple: $K = I$.

Now Q.

observation:

$$u_c u_d u_d^T = u_c \|u_d\|^2 \Rightarrow \frac{u_c u_d u_d^T}{\|u_d\|^2} = u_c$$

$$u_d u_a u_a^T = u_d \|u_a\|^2 \Rightarrow \frac{u_d u_a u_a^T}{\|u_a\|^2} = u_d$$

So Q must be:

$$Q = \frac{u_c u_d^T}{\|u_d\|^2} + \frac{u_d u_a^T}{\|u_a\|^2} = \frac{u_c u_d^T + u_d u_a^T}{\beta^2}$$

To summarize completly we need:

$$V = \frac{u_b u_b^T - u_c u_c^T}{\beta^2}$$

$$K = I$$

$$Q = \frac{u_c u_d^T + u_d u_a^T}{\beta^2}$$

2 Pretraining Transformer-based Generative Model

d)

accuracy on dev: 1.2% accuracy on london baseline: 25%

f)

accuracy on dev: 16.2%

g)

Question 1: Explain in no more than 3 sentences the idea behind Linformer.

The original transformer architecture presented in "Attention is all you need" uses $O(n^2)$ time and space with respect to the sequence length when calculating self-attention. Authors of Linformer show, that the self-attention matrix is often low-rank (thus most of the dimensions are unnecessary) and can be approximated, saving memory and time. Specifically, they add a linear projection layer that serves to reduce dimensionality.

Question 2: Explain in no more than 3 sentences the idea behind BigBird.

Authors of Big Bird replace self-attention with sparse attention mechanism that is linear in the number of tokens. The sparse attention consists of three types of attention:

- random attention - intuitively: each token attends to r random tokens
- window attention - intuitively: each token attends to itself but also to w its nearest neighbors
- global attention - intuitively: select g tokens that are attended to every other

Numbers r, w and g are hyperparameters.

Question 3: In what respects Big Bird is as powerful and expressive as full-attention mechanisms? Explain in up to 3 sentences.

In the paper authors of the Big Bird show that sparse-attention is enough to simulate any Turing Machine and we do not need full-attention. They also show that sparse-attention mechanism used as a encoder is Universal Approximation of sequence to sequence functions.

*h)

Question 1: Explain in less than 3 sentences high-level idea of attention approximation from the paper to speed up computations? What are expected speedups?

Authors try to approximate softmax attention kernels to reduce time and space complexity. To achieve this, they use a novel approach FAVOR+ (Fast Attention Via positive Orthogonal Random features approach). They basically construct an unbiased estimator to the Attention matrix that is arbitrarily good in linear time.

Question 2: What problem might arise when the the softmax-kernel which defines regular attention matrix is approximated naively by trigonometric functions?

The problem arises when you try to apply random feature maps with potentially negative dimension values. It leads to problems with training because kernel scores close to 0 are approximated by large variance estimators. It is partly because trigonometric functions are sometimes negative, and softmax is always positive.

Question 3: How does authors of the paper avoid the problem from question 2? What approximation can help here?

They propose novel method called Positive Random Features (PRF). Instead of using triginometric functions (that sometimes can be negative and that leads to problems) they use $f_1(u) = exp(u)$ and $f_2(u) = exp(-u)$

Question 4: What are we achieving by sampling orthogonal random features?

We can reduce the variance of softmax kernel estimators for any dimensionality d.

3 Pretrained Models as Source of Knowledge

a)

Briefly explain why the pretrained (vanilla) model was able to achieve an accuracy of above 10%, whereas the non-pretrained model was not.

The fact that pretrained transformers work better than non-pretrained transformers is generally true not only for NLP tasks. The pretraininig was on data which contained information about people and their birthplaces so the model "remembered" it. Moreover I think the pretraining helped the model to understand the English language, so when it was given new data, it did not have to learn basic language structure from zero and was able to focus specifically on making good birth-place predictions. Another explanation is from the paper "Why Does Unsupervised Pre-training Help Deep Learning?". Authors suggest that unsupervised pre-training adds robustness to a deep architecture. Their results show that pre-trained networks give consistently better generalization. They also claim that pre-trained models learn qualitatively different features (if networks are visualized in the weight space) compared to networks without pre-training.

b)

Take a look at some of the correct predictions of the pretrain+finetuned vanilla model, as well as some of the errors. We think you'll find that it's impossible to tell, just looking at the output, whether the model retrieved the correct birth place, or made up an incorrect birth place. Consider the implications of this for user-facing systems that involve pretrained NLP components. Come up with two reasons why this indeterminacy of model behavior may cause concern for such applications.

The model invented the place "Akraine" that looks real so one may take this as the correct birthplace. It also predicted wrong birthplaces even though they were real. When

someone sees the output from the model, he will be biased towards it. Moreover, many wrong predictions can make people working with such a model think that it is really bad, even though it is still better (on average) than random. Eventually, they can stop using it and lose trust in AI.

c)

If your model didn't see a person's name at pretraining time, and that person was not seen at fine-tuning time either, it is not possible for it to have "learned" where they lived. Yet, your model will produce something as a predicted birth place for that person's name if asked. Concisely describe a strategy your model might take for predicting a birth place for that person's name, and one reason why this should cause concern for the use of such applications.

I think the model tries to predict the same birthplace as for the known person with similiar name. It is concerning because it shows that model tries to correlate names with birthplaces.