Practical No. 4

Mateusz Kwiatkowski
mk457176@students.mimuw.edu.pl

May 1, 2023

1 Part 2

a)

We know that $c = \sum_{i=1}^{n} v_i \alpha_i$, so if $\alpha_j \approx 1$, we can be sure that $c \approx v_j$. For α_j to be close to one it must occur $exp(k_j^T q) \approx \sum_{i=1}^{n} exp(k_i^T q)$, so $k_j^T q$ is much larger than $k_i^T q$ for all $i \neq j$. This means that q is similar to k_j and not similar to the other keys.

b)

$$c \approx \frac{v_a + v_b}{2} \Rightarrow \alpha_a = \alpha_b \approx \frac{1}{2} \text{ and } \alpha_i \approx 0 \text{ for } i \notin \{a, b\}$$

$$\alpha_a = \alpha_b \Rightarrow exp(k_a^T q) = exp(k_b^T q) \Rightarrow k_a^T q = k_b^T q \Rightarrow (k_a - k_b)^T q = 0$$

This means that $(k_a - k_b) \perp q$, so q can be equal to $n(k_a + k_b)$, where n is real number, because $n(k_a - k_b)^T (k_a + k_b) = n||k_a|| - n||k_b|| = 0$

$$exp(k_i^T q) = exp(n k_i^T (k_a + k_b)) = exp(0) = 1 \text{ for } i \notin \{a, b\}$$

$$exp(k_b^T q) = exp(k_a^T q) = exp(n k_a^T (k_a + k_b)) = exp(n)$$

n must be a very large number, because then $\alpha_a \approx \frac{1}{2}$.
Answer: $q = n(k_a + k_b)$

c)

This query will be equal to $n(\mu_a + \mu_b)$, where n is a very large number, because for vanishingly small α: $k_i \approx \mu_i$.

$$exp(k_i^T q) = exp(n k_i^T (\mu_a + \mu_b)) = exp(n k_i^T \mu_a + n k_i^T \mu_b)) \approx exp(0) = 1 \text{ for } i \notin \{a, b\}$$

$$exp(k_b^T q) = exp(k_a^T q) = exp(n k_a^T (\mu_a + \mu_b)) = exp(n k_a^T \mu_a + n k_a^T \mu_b)) \approx exp(n)$$

In the second case, we have $\Sigma_a \approx \frac{1}{2}(\mu_a \mu_a)^T$, so the values in the covariance matrix can be large, which means that assumption that $k_a \approx \mu_a$ no longer applies. Very often we will have the following situation:

$$k_a q \neq \mu_a q = n$$

We can therefore see that $\alpha_a \neq \alpha_b$, so $q = v_a \alpha_a + v_b \alpha_b$ is a weighted average, but this weights are not equal.

d)

We can take $q_1 = n\mu_a$ and $q_2 = n\mu_b$, where n is a large real number, because then we got very high values of $\alpha_{1,a}$ and $\alpha_{2,b}$, so $c_1 = v_a$, $c_2 = v_b$, which means that $c = \frac{1}{2}(v_a + v_b)$. We can also see that in this case large values in the covariance matrix are not a problem for us. From the content of the task, we can assume that $q_1^T k_a > 0$, so we can make this value very high, by increasing n. It will make $alpha_{1,a}$ large, so $c_1 \approx v_a$. Similarly we can make that $c_2 \approx v_b$. Our result c will be close to the average of v_a and v_b.

e)

$$k_1^T q_2 = x_1^T x_2 = (u_d + u_b)^T u_a = 0$$

$$k_2^T q_2 = x_2^T x_2 = u_a^T u_a = \beta$$

$$k_3^T q_2 = x_3^T x_2 = (u_c + u_b)^T u_a = 0$$

$$\alpha_{21} = \frac{1}{2 + exp(\beta)} \approx 0$$

$$\alpha_{22} = \frac{exp(\beta)}{2 + exp(\beta)} \approx 1$$

$$\alpha_{23} = \frac{1}{2 + exp(\beta)} \approx 0$$

$$c_2 \approx v_2 = x_2 = u_a$$

It's impossible for c_2 to approximate u_b, because even after adding u_d or u_c, the value of the $k_2^T q_2 = x_2^T x_2$ will be large, so the weight α_{22} won't be close to zero. This means that u_a will be a significant component of c_2, thus c_2 is not an approximation of u_b.
Now we must construct V matrix, i-th row of this matrix should be equal to $u_{b,i}/\beta u_b^T - u_{c,i}/\beta u_c^T$, where $u_{b,i}$ is a i-th element of vector u_b.

$$v_1 = V x_1 = V u_d + V u_b = V u_b = u_b$$

$$v_3 = V x_3 = V u_c + V u_b = -(u_c^T u_c) u_c/\beta + (u_b^T u_b) u_b/\beta = -u_c + u_b$$

2 Part 3

c+d)

The model without pretraining answered only 5 questions correctly, giving only 1% accuracy. This is a very negative conclusion, as better results would be obtained by answering "London" every time (5% accuracy). All the logs are saved in my notebook Practical4.ipynb

e+f)

The model achieved an accuracy of 13.8%, a significant improvement on the previous experiment. This may not be a very satisfactory result yet, but I managed to beat the target of 10%.

g)

Question 1

Linformer addresses the issue of the quadratic complexity by approximating the self-attention matrix with a low-rank matrix, leading to a linear complexity in the sequence length. This approximation is achieved by factoring the full self-attention matrix into two matrices using singular value decomposition. By using an efficient matrix multiplication algorithm, Linformer achieves comparable performance to the standard self-attention with significantly reduced computational costs.

Question 2

The key idea behind BigBird is to use a sparse attention mechanism, where only a subset of the input sequence is attended to in each step. This allows the model to efficiently process long sequences without requiring excessive computational resources. Additionally, BigBird introduces global and local attention patterns that capture both long-range and short-range dependencies in the input sequence.

Question 3

BigBird first proved sparse attention mechanism defined by any graph containing star graph is a universal approximator. This approach can also be used to simulate a Turing machine.

h)

Question 1

This research uses the FAVOR+ algorithm (Fast Attention Via Positive Orthogonal Random Features), in which the A matrix is replaced by two matrices (Q' and K'^T), reducing complexity. The authors of the paper proved that after the changes they made, the time complexity is equal to $O(Ldr)$, where L is the length of the sequence, d is the dimension of

the latent space and r is dimensionality of feature map codomain, so it is linear with respect to every parameter.

Question 2

When using trigonometric functions as random feature maps, the resulting dimension values can be negative, leading to unstable behavior. In regions where the kernel scores are close to zero, using estimators with large variance can cause the approximation to become even more unstable. The authors provide theoretical explanations showing that the variance of the estimator of such an approximated regular attention matrix is large when the approximated values tend towards zero.

Question 3

The authors of this paper use an unbiased estimator that returns only positive values: $\phi(x) = exp(w^T x - \frac{1}{2}x^T x)$

Question 4

Sampling orthogonal random features reduces the correlation between the random features, which can help to reduce overfitting and improve the generalization performance of the model. It is also more efficient computationally.

3 Part 4

a)

During pre-training, the model was able to take information about the data and its structure. This was very helpful in the fine-tuning phase, as it meant that the model did not have to learn from scratch and could use the knowledge gained during the long pre-training.

b)

A big problem can be bias in the data that will lead to the model replicating stereotypes. In such cases, we may end up with a situation where the user is unable to verify which of the generated data are true and which are the result of incorrectly selected data.

c)

The model in these situations will not know the correct answer, so it will try to guess it based on observed examples. He may look at people with the same name or surname. He may also look for people whose name sounds similar. Doing this will probably not give the correct result, but it is very likely that at least the country will match. Using models that work in this way can be a cause for concern, as it can be used to identify the nationality of people about whom we know nothing.