Mathematics for New Technologies in Finance

Solution sheet 9

Background. Associative recall measures a generative model's capability to remember relevant information from past input. This requires storing information in hidden states and is therefore subject to information-theoretic bounds. For transformers, these bounds depend on the size of the context window and also on the choice of attention mechanism. Here, we consider a stylized recall task and ask for a *constructive* solution. The didactical goal is to get familiar with query, key, and value computations.

Definition. Given a sequence of key-value pairs and a key, the associative recall task is to output the value corresponding to this key. Example:

$$\text{A6B8C3B} \to 8$$

Accordingly, a correct input-output pair (x, y) for this example is

$$x = \text{A6B8C3B?} \qquad y = \text{A6B8C3B8}$$

Exercise 9.1 Are transformers able to perfectly solve the associative recall task, for recall sequences with fixed length?

Hint. As the focus is on the attention mechanism, you may for simplicity choose arbitrary functions for token embeddings, position embeddings, and feed-forward networks. Moreover, you may ignore layer normalization. Finally, it might be helpful to note that a transformer block whose attention layer has all-zero weights is just a feed-forward network.

Solution 9.1

Token and Position Embedding. Let us consider

$$\{e_0, \ldots, e_9, e_A, \ldots, e_Z, e_?\}$$

as an orthonormal basis of a real vector space for token and position embeddings. Namely, we define these embeddings such that the input sequence A6B8C3B? becomes

$$X := \begin{pmatrix} e_A & e_6 & e_B & e_8 & e_C & e_3 & e_B & e_? \\ e_8 & e_7 & e_6 & e_5 & e_4 & e_3 & e_2 & e_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}^\top \in \mathbb{R}^{n \times d},$$

where $n = 8$ and $d = 3 \times (10 + 26 + 1) = 111$. The first row will be used for input, the second row for indexing, and the third row for output.

First Transformer Block. This block prepares the second row for alphabetic lookup. For suitably defined weight matrices W_Q, W_K, W_V, we obtain the following queries, keys, and values:

$$XW_Q = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_8 & e_8 & e_6 & e_6 & e_4 & e_4 & e_2 & e_2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}^\top \in \mathbb{R}^{n\times d},$$
$$XW_K = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_8 & e_7 & e_6 & e_5 & e_4 & e_3 & e_2 & e_1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}^\top \in \mathbb{R}^{n\times d},$$
$$XW_V = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_A & e_6 & e_B & e_8 & e_C & e_3 & e_B & e_? \end{pmatrix}^\top \in \mathbb{R}^{n\times d}.$$

Indeed, this can be achieved using query weights W_Q which nullify the first and last rows and replace all e_k with odd k by e_{k+1}, using key weights W_K which merely nullify the first and last rows, and using value weights W_V which copy the first row into the last one and then nullify the first and second rows. The resulting masked attention matrix is

$$A := \rho\big((QW_Q)(KW_K)^\top/\sqrt{d}\big) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \in \mathbb{R}^{n\times n}.$$

The output of the attention head is therefore

$$AXW_V = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_A & e_A & e_B & e_B & e_C & e_C & e_B & e_B \end{pmatrix}^\top \in \mathbb{R}^{n\times d},$$

and the output of the skip connection is

$$AXW_V = \begin{pmatrix} e_A & e_6 & e_B & e_8 & e_C & e_3 & e_B & e_? \\ e_8 & e_7 & e_6 & e_5 & e_4 & e_3 & e_2 & e_1 \\ e_A & e_A & e_B & e_B & e_C & e_C & e_B & e_B \end{pmatrix}^\top \in \mathbb{R}^{n\times d}.$$

Using a feed-forward network with skip connection, we add the third row to the second row whenever the number in the second row is odd. We thus obtain the following output of the first transformer block, which is simultaneously the input for the second transformer block and is again denoted by X:

$$X := \begin{pmatrix} e_A & e_6 & e_B & e_8 & e_C & e_3 & e_B & e_? \\ e_8 & e_7 + e_A & e_6 & e_5 + e_B & e_4 & e_3 + e_C & e_2 & e_1 + e_B \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}^\top \in \mathbb{R}^{n\times d}.$$

Second Transformer Block. The second transformer block performs the lookup operation. For suitably defined weight matrices W_Q, W_K, W_V, different from the above ones with the same name, we obtain the following queries, keys, and values:

$$XW_Q = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_8 & e_7 + e_A & e_6 & e_5 + e_B & e_4 & e_3 + e_C & e_2 & e_B \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}^\top \in \mathbb{R}^{n\times d},$$

$$XW_K = XW_Q,$$

$$XW_V = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_A & e_6 & e_B & e_8 & e_C & e_3 & e_B & 0 \end{pmatrix}^\top \in \mathbb{R}^{n\times d}.$$

Indeed, this can be achieved using equal query and key weights $W_Q = W_K$ which nullify the first row, last row, and all vectors e_1, and using value weights W_V which copy the first into the last row, delete all vectors $e_?$, and nullify the first and second rows. The resulting masked attention matrix is

$$A := \rho\big((QW_Q)(KW_K)^\top/\sqrt{d}\big) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \lambda & 0 & 0 & 0 & 1-\lambda \end{pmatrix} \in \mathbb{R}^{n\times n},$$

where $\lambda = \frac{e^{1/\sqrt{d}}}{e^{1/\sqrt{d}} + e^{2/\sqrt{d}}} \in (0,1)$. The output of the attention head is therefore

$$AXW_V = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_A & e_6 & e_B & e_8 & e_C & e_3 & e_B & \lambda e_8 \end{pmatrix}^\top \in \mathbb{R}^{n\times d},$$

and the output of the skip connection is

$$AXW_V = \begin{pmatrix} e_A & e_6 & e_B & e_8 & e_C & e_3 & e_B & e_? \\ e_8 & e_7 + e_A & e_6 & e_5 + e_B & e_4 & e_3 + e_C & e_2 & e_1 + e_B \\ e_A & e_6 & e_B & e_8 & e_C & e_3 & e_B & \lambda e_8 \end{pmatrix}^\top \in \mathbb{R}^{n\times d}.$$

A suitable feed-forward network with skip connection implements the identity operation. Then, the output of the second transformer block, and the input for the transformer's output layers, is the above matrix.

Output Layers. A linear read-out extracts the third row and enforces deterministic output by setting the so-called temperature to zero. This amounts to multiplication by ∞ or some very high floating point number, resulting in

$$\infty \times \big(e_A \quad e_6 \quad e_B \quad e_8 \quad e_C \quad e_3 \quad e_B \quad e_8\big)^\top \in \bar{\mathbb{R}}^{n\times 37}.$$

These are logits for a categorical distribution which generates the desired deterministic output.

Exercise 9.2 (Memory of Signature transforms) Notice the analogy to the first example:

(a) Given a C^1 trajectory $u : [0,1] \to \mathbb{R}^2$. How many and which signature components do you need to know to reconstruct the area swept over by the vector from $u(0)$ to $u(t)$ on the interval $[0,1]$.

(b) Let us assume that the first component is time, i.e. $u^1(t) = t$ for $t \in [0,1]$. Which signature components do you need to know (possibly countably many) to reconstruct $u^2(t) - u^2(0)$ perfectly for every $t \in [0,1]$. If you can only store finitely many signature components: which ones to choose to do an approximate reconstruction and what would be a criterion for the approximation quality (Notice that u is C^1 and that one can make a Fourier expansion of $u^2(t) - u(0)$).

Solution 9.2

(a) We need $Sig_{12}(u)$ and $Sig_{21}(u)$.

(b) First expand $\dot{u}^2$ with Fourier expansion, then using Taylor expansion for the coefficients.

$$\sum_{u=0}^{\infty} \int exp(i\Delta 2\pi)\dot{u}^2(s)ds = \sum_{k=0}^{\infty} \frac{1}{k!}(i2\pi e)^k \int_0^1 \frac{\Delta^k}{k!} du^2(s) \tag{1}$$

There are other criterions can be used for the approximation quality depending on the data. In this problem, we use squared L2 norm for error estimation.