Meeting 5: transformers

Outline

- Scaled Dot Product Attention
- Multi-Head Attention
- Encoder-Decoder Architecture
- Universal Declaration of Human Rights Corpus

Sequence neural networks

- A sequence neural network converts a sequence of inputs, $I = [i_1; \dots; i_T]$, into a sequence of outputs, $O = [o_1; \dots; o_T]$, where i_t and o_t are row vectors, and ";" denotes vertical concatenation.
- We always want to re-use some basic network, $o_t = f(i_t)$, with weights that are shared at every time step.

Temporal context

i_t is not enough information to compute o_t. We need to have some context vectors. So instead of $o_t = f(i_t)$, we need $o_t = f(i_t, h_1(t), \ldots, h_h(t))$, where $h_i(t)$ is the i^{th} context vector relevant to making decisions about i_t.

1. Convolutional neural network (CNN): context vectors = neighbors of x_t, e.g., $h_1(t) = i_{t-h/2}$ up through $h_h(t) = i_{t+h/2}$.
2. Recurrent neural network (RNN): $h_i(t)$ = hidden state vector
   - $h_i(t)$ is a vanilla RNN, GRU, or LSTM function of i_t and $h_i(t-1)$
3. Transformer: $h_i(t) = \sum_{\tau=1}^{T} \alpha_i(t,\tau) i_\tau W_i^V$
   - $\alpha_i(t,\tau)$ is a function of i_t and i_τ called the attention
   - So $h_i(t)$ is a weighted linear combination of the linearly transformed values $i_\tau W_i^V$ at time steps τ, for all $1 \leq \tau \leq T$. It's like a CNN, but the attention can select context vectors either near or far away in time.

Attention = convex weighted average of inputs

- Suppose we wanted to pick out some particular input vector as the i^{th} context vector, i.e., $h_i(t) = i_{\tau^*(t)}$ for some $\tau^*(t)$ which is the most relevant context vector. We could do that by using the unit indicator function, $\alpha_i(t,\tau) = \begin{cases} 1 & \tau = \tau^*(t) \\ 0 & \text{otherwise} \end{cases}$,
- but the unit indicator function is not differentiable.
- Instead, we require $\alpha_i(t,\tau) \geq 0$, and $\sum_{\tau=1}^{T} \alpha_i(t,\tau) \ = 1$.
- In other words, $\alpha_i(t,\tau)$ is a softmax function over the input time steps:

$$\alpha_i(t,\tau) = \frac{\exp\big(e_i(t,\tau)\big)}{\sum_{k=1}^{T} \exp\big(e_i(t,k)\big)}$$

- …where $e_i(t,\tau)$ is a measure of the relevance of i_τ.

Excitation: relevance, computed as a scaled dot product

- Attention $\alpha_i(t,\tau)$ is a normalized $[0,1]$ measure of the relevance of i_τ for decisions about i_t. Excitation, $e_i(t,\tau)$, is the unnormalized relevance measure.
- $e_i(t,\tau)$ could be computed by any nonlinear function, but (Vaswani et al., 2017) just use a scaled, masked dot product. Here is the dot product part:

$$d_i(t,\tau) = i_t W_i^Q \left(i_\tau W_i^K\right)^T$$

Where

- $i_t W_i^Q$ is a row vector called the "query;" W_i^Q is the i^{th} query weight matrix.
- $i_\tau W_i^K$ is a row vector called the "key;" W_i^K is the i^{th} key weight matrix
- The term $d_i(t,\tau)$ is not used in the article, but I've separated it out here. I'm using the letter d to mean "dot product."

Scaling the dot product

- The dot product is a bad measure of relevance, because it can get arbitrarily large (e.g., if W_i^Q and W_i^K converge to large numbers). When $d_i(t,\tau)$ gets large, then the difference between the largest and second-largest also gets large, so the softmax, $\alpha_i(t,\tau)$, approaches a unit indicator function, which makes it hard to compute the gradient (hard to continue learning).
- The usual solution to this problem, in information retrieval, is to use a cosine-similarity ($\frac{i_t W_i^Q \left(i_\tau W_i^K\right)^T}{\left\|i_t W_i^Q\right\| \cdot \left\|i_\tau W_i^K\right\|}$) instead of a dot product. (Vaswani et al., 2017) didn't do that. I don't know why; maybe they found that scaling by d_k worked better.
- The formula (Vaswani et al., 2017) used is just a scaled dot product, which I'll call $c_i(t,\tau)$:

$$c_i(t,\tau) = \frac{d_i(t,\tau)}{\sqrt{d_k}}$$

where d_k is the dimension of the row vector $i_\tau W_i^K$.

- Plausible post-hoc intuitive justification: if each element of each of the vectors $i_t W_i^Q$ and $i_\tau W_i^K$ were unit normal Gaussian, then $d_i(t,\tau)$ would be Gaussian with a standard deviation of $\sqrt{d_k}$.

Masking the scaled dot product

- Sometimes we know future inputs during training, but not during testing. If that happens, we need to train the neural net so it won't depend on future inputs (i.e., make it ***causal***).
- Remember that $h_i(t) = \sum_{\tau=1}^{T} \alpha_i(t,\tau) i_\tau W_i^V$. We can force causality by just setting $\alpha_i(t,\tau) = 0$ whenever $\tau > t$.
- Since $\alpha_i(t,\tau)$ is the output of a softmax, we can set $\alpha_i(t,\tau) = 0$ by setting the softmax input to $e_i(t,\tau) = -\infty$. This is done by masking out future inputs:

$$e_i(t,\tau) = m\big(t-\tau, c_i(t,\tau)\big)$$

where $m(t-\tau, c_i(t,\tau))$ is a "masking function:" :

$$m\big(t-\tau, c_i(t,\tau)\big) = \begin{cases} c_i(t,\tau) & t-\tau \geq 0 \\ -\infty & t-\tau < 0 \end{cases}$$

Scaled dot product attention:

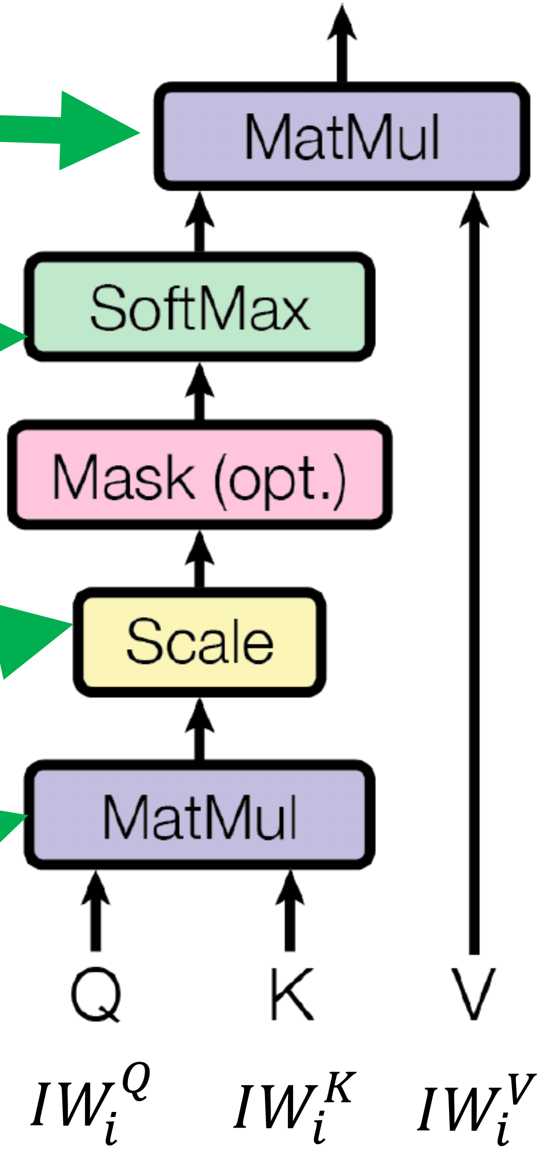
$$h_i(t) = \sum_{\tau=1}^{T} \alpha_i(t,\tau) i_\tau W_i^V$$

$$\alpha_i(t,\tau) = \frac{\exp\big(e_i(t,\tau)\big)}{\sum_{k=1}^{T} \exp\big(e_i(t,k)\big)}$$

$$e_i(t,\tau) = m\big(t-\tau, c_i(t,\tau)\big)$$

$$c_i(t,\tau) = \frac{d_i(t,\tau)}{\sqrt{d_k}}$$

$$d_i(t,\tau) = i_t W_i^Q \left(i_\tau W_i^K\right)^T$$



Outline

- Scaled Dot Product Attention
- Multi-Head Attention
- Encoder-Decoder Architecture
- Universal Declaration of Human Rights Corpus

Temporal context

- Now that you've found the most relevant context vectors ("head") $h_i(t)$, for $1 \leq i \leq h$, what do you do with them?
- Solution, for now: concatenate all the $h_1(t), \ldots, h_h(t)$ into a long vector, then multiply it by a weight matrix W^O:

$$x_t = [h_1(t), \ldots, h_h(t)]W^O$$

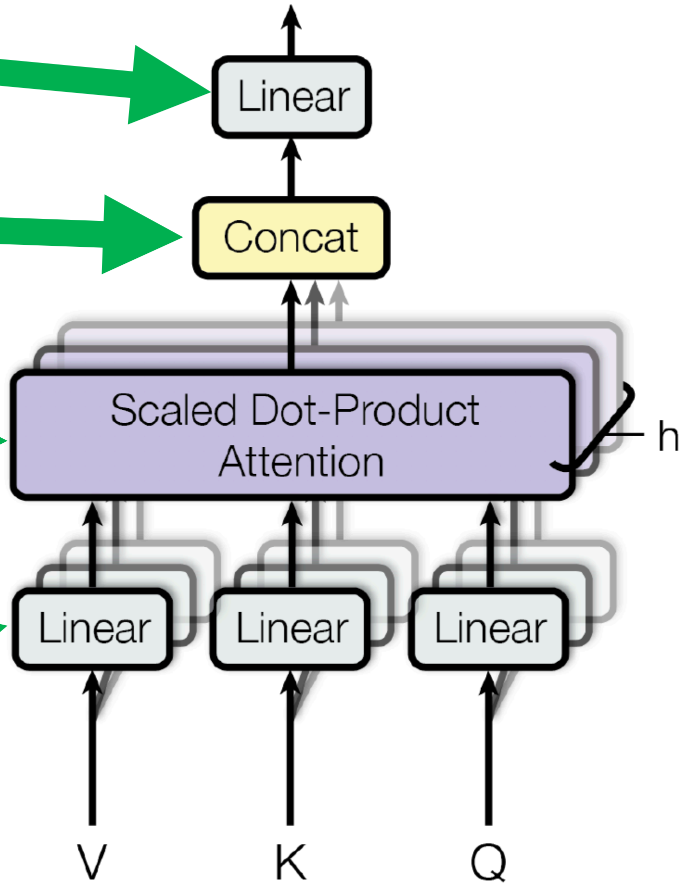
Multi-headed attention

$$X = [H_1, \dots, H_h]W^O$$

$$[H_1, \dots, H_h]$$

$$H_i = \underset{columns}{\operatorname{softmax}}\left(\frac{IW_i^Q\left(IW_i^K\right)^T}{\sqrt{d_k}}\right)IW_i^V$$

Compute the linear transforms IW_i^Q, IW_i^K, and IW_i^V



Outline

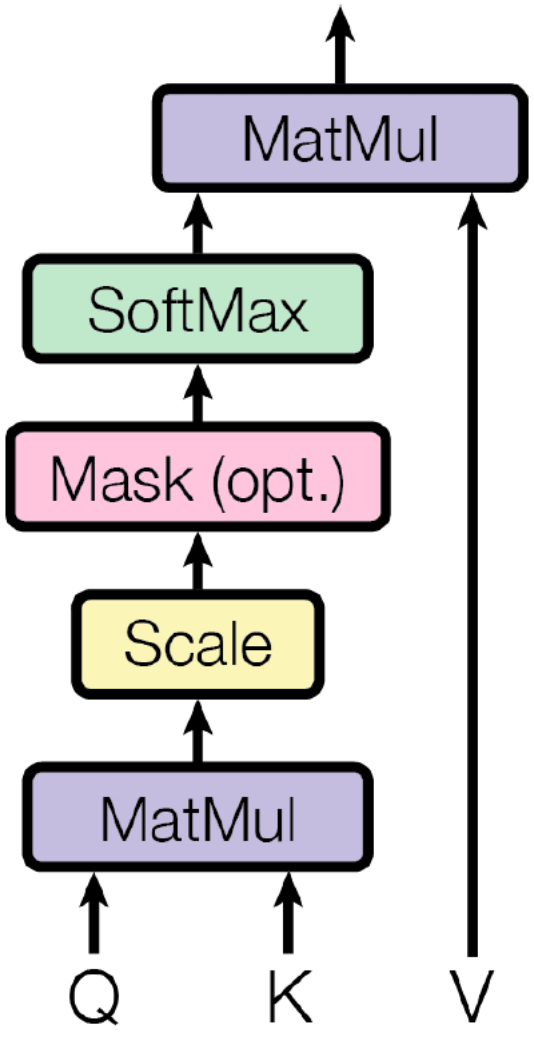
- Scaled Dot Product Attention
- Multi-Head Attention
- Encoder-Decoder Architecture
- Universal Declaration of Human Rights Corpus

Scaled dot product attention more generally

The query, key, and value need not come from the same input signal. They might come from three different input signals, $Q = [q_1; \dots; q_U]$, $K = [k_1; \dots; k_T]$, and $V = [v_1; \dots; v_T]$. K and V need to have the same number of rows, but Q doesn't. In that case, attention becomes

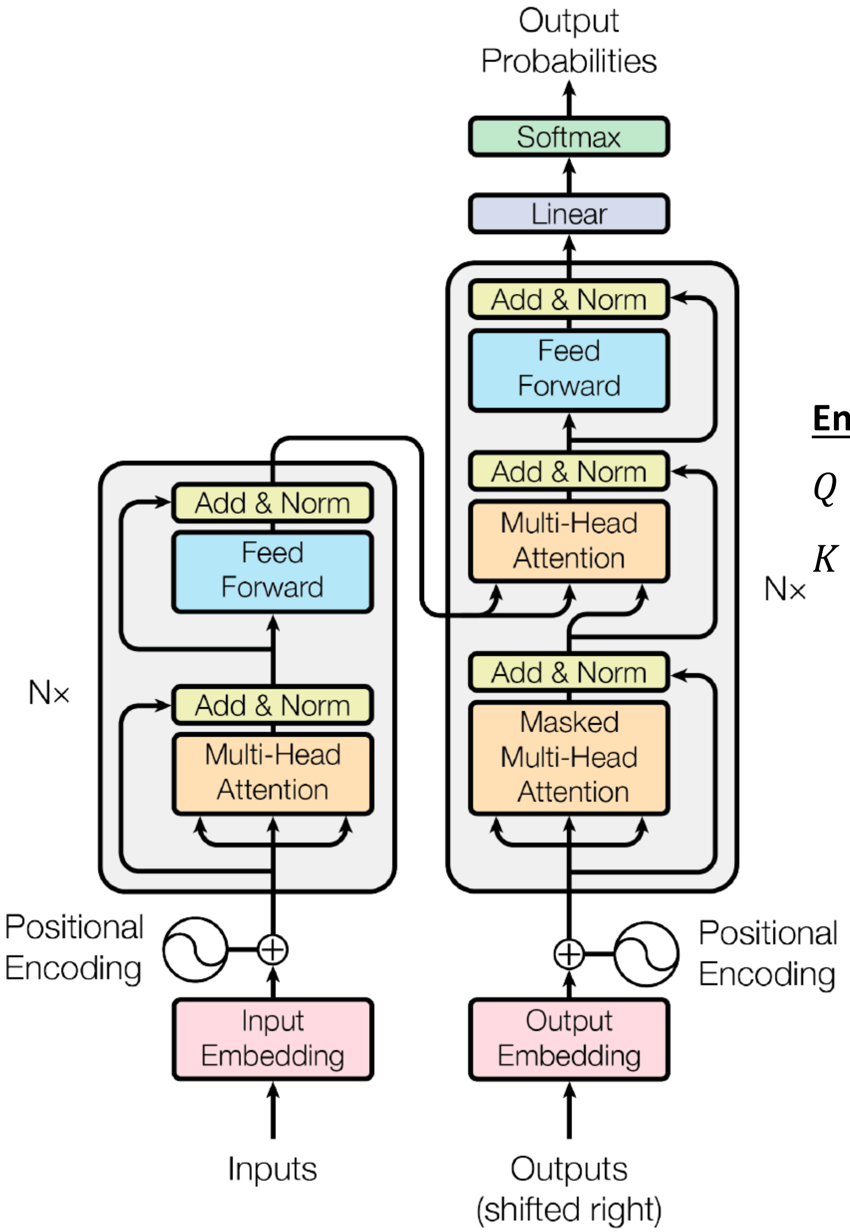
$$H_i = \underset{columns}{\operatorname{softmax}}\left(\frac{QW_i^Q\left(KW_i^K\right)^T}{\sqrt{d_k}}\right)VW_i^V$$

We call it "self-attention" when Q and K are the same matrix, otherwise it's just "attention."



Three uses of attention in the Transformer

The transformer uses the same attention module in three different places.



Encoder-decoder attention:

Q =outputs of decoder self-attention,

$K = V$ =encoder outputs

Encoder self-attention:

Q =inputs, K =inputs, V =inputs

Decoder self-attention:

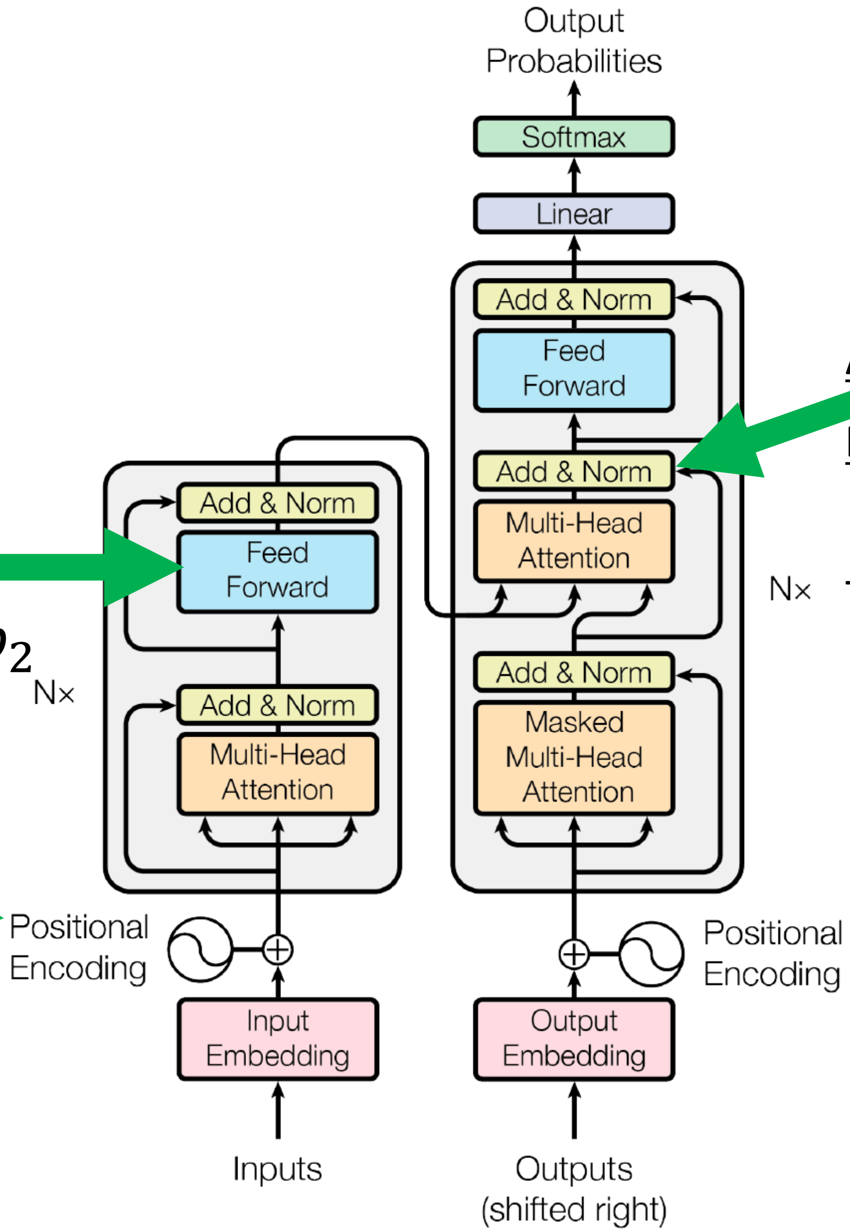
Q =outputs, K =outputs, and V =outputs

(masked so it's causal)

(shifted by one so that $q_t = k_t = v_t$ is the vector embedding of the reference class label from time $t-1$)

Other components in the Transformer

Here are the other components mentioned in the article:



Feed Forward:

$$o_t = \max(0, x_t W_1 + b_1) W_2 + b_2$$

Positional Encoding:

$$I = I + P$$

$$P_{t,2k} = \sin\left(\frac{t}{10000^{2k/\mathrm{len}(i_t)}}\right)$$

$$P_{t,2k+1} = \cos\left(\frac{t}{10000^{2k/\mathrm{len}(i_t)}}\right)$$

Add & Norm:

Residual connection & LayerNorm:

$$x_t = x_t + i_t$$

Then

$$x_{t,k} = \frac{x_{t,k} - \mu_t}{\sigma_t}$$

$$\mu_t \equiv \frac{1}{\mathrm{len}(x_t)} \sum_{k=1}^{\mathrm{len}(x_t)} x_{t,k}$$

$$\sigma_t^2 \equiv \frac{1}{\mathrm{len}(x_t)} \sum_{k=1}^{\mathrm{len}(x_t)} \left(x_{t,k} - \mu_t\right)^2$$

Overview of training

- Each training example is a mel-spectrogram, and a phone transcription.
- Encoder self-attention: $q_t = k_t$= $v_t = t^{\mathrm{th}}$ mel spectral vector
- Decoder self-attention: $q_u = k_u$= $v_u =$ vector embedding of the u^{th} phone symbol (let's call the u^{th} phone symbol φ_u), starting with q_0=embedding of the symbol before the beginning of the phone transcription (silence), and ending with q_{U-1}, the second-to-last phone symbol.
- Loss function: cross-entropy, computed w.r.t. reference phone symbols q_1 through q_U, i.e., the actual phone transcription

$$\mathcal{L} = -\sum_{u=1}^{U} \ln p(\varphi_u)$$

Outline

- Scaled Dot Product Attention
- Multi-Head Attention
- Encoder-Decoder Architecture
- Universal Declaration of Human Rights Corpus

Universal Declaration of Human Rights Corpus