AOS 2 – Deep learning

Lecture 05: Attention Neural Networks

Sylvain Rousseau

What is attention

- Selectively concentrating on a few things
- Lead to breakthroughs on NLP, vision, speech recognition
  - Fist applied to neural networks in Bahdanau, Cho, and Bengio 2016 on top of RNN
  - Transformer architecture in Vaswani et al. 2017 and self-attention
  - In vision in Xu et al. 2016 with attention between images and captions
  - Vision transformer in Dosovitskiy et al. 2021

Attention

- Parametrized transform of a set of n vectors in $\mathbb{R}^p$ into a set of m vectors in $\mathbb{R}^p$

$$V^T = [\mathbf{v}_1, \dots, \mathbf{v}_n] \xrightarrow{\quad\Theta\quad} Y^T = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_m]$$

- Linear transform parametrized by **attention coefficients**

$$\boldsymbol{y}_j = \sum_{k=1}^{n} a_{jk}\mathbf{v}_k \quad \text{for } j = 1, \dots, m \qquad \text{or} \qquad Y = AV \quad \text{with} \quad A \in \mathbb{R}^{m\times n}$$

- Attention score a_{jk} reads: how $\mathbf{v}_k$ is relevant for new vector $\boldsymbol{y}_j$
- Attention is usually calculated from a **softmax on attention scores**

$$a_{jk} = \frac{\exp(s_{jk})}{\sum_{l=1}^{n}\exp(s_{jl})} \qquad A = \operatorname{Softmax}^{\leftrightarrow}(S)$$

- Softmax on rows of score matrix $S \in \mathbb{R}^{m\times n}$

Content attention

$$V^T = [\mathbf{v}_1, \dots, \mathbf{v}_n] \xrightarrow{\quad\Theta = (\theta_1, \dots, \theta_m)\quad} Y^T = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_m]$$

- Contribution of $\mathbf{v}_k$ in output $\boldsymbol{y}_j$ depends on the content $\mathbf{v}_k$ and a parameter θ_j

$$s_{jk} = s_{jk}(\mathbf{v}_k, \theta_j)$$

- For a given j, contribution of $\mathbf{v}_k$ for $\boldsymbol{y}_j$ depends only on the content $\mathbf{v}_k$

Context attention

$$V^T = [\mathbf{v}_1, \dots, \mathbf{v}_n] \xrightarrow{C^T = [\boldsymbol{c}_1, \dots, \boldsymbol{c}_m], \theta_c} Y^T = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_m]$$

- Scores depend on content $\mathbf{v}_k$, context $\boldsymbol{c}_j$ and parameters θ_c

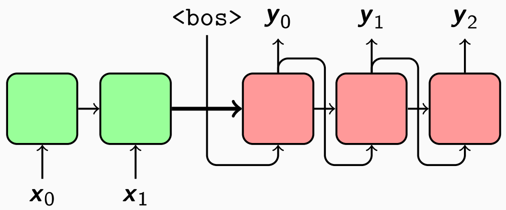
$$s_{jk} = s_{jk}(\mathbf{v}_k, \boldsymbol{c}_j, \theta_c)$$

- $\boldsymbol{c}_j$ is data not a parameter!

Application to sequence to sequence modeling

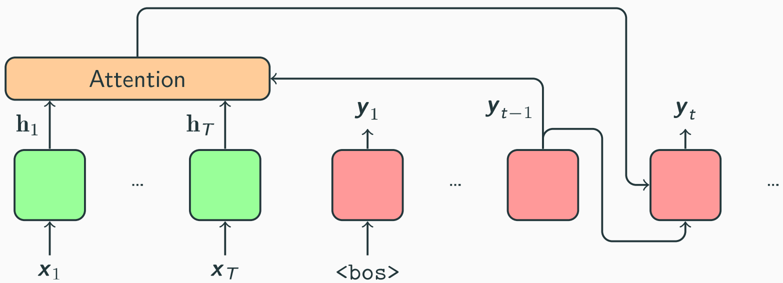
- Machine translation
  Translate a text from one laguage to another
- Video captioning
  Generate a caption describing a video
- Open-ended question answering
  Give a statement as an answer
- Summarization
  Provide a summary of a long text

- Encoder–Decoder architecture
- Input sequence is entirely consumed
- All information is packed in last hidden state: bottleneck problem



Attention in Bahdanau, Cho, and Bengio 2016

- Context attention with $n = T$, $m = 1$
- Values are successive hidden states of RNN



Attention matrix

From english to french

- Give context to output correct token
- Attention matrix is aligning sequences

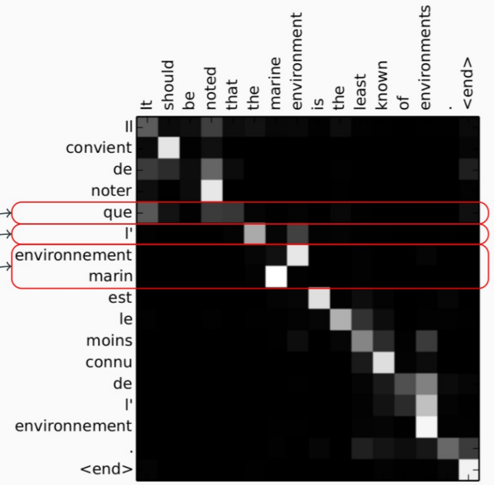


Figure 1: from Bahdanau, Cho, and Bengio 2016

Results and limitations

- BLEU score Models with attention have a better BLEU score
- Limitations
  - Still using RNN
  - One level attention only

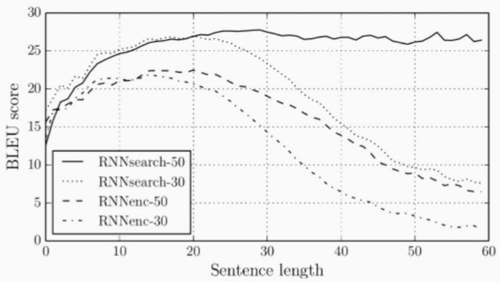


Figure 2: from Bahdanau, Cho, and Bengio 2016

Transformer architecture

Transformer architecture

- Introduced in Vaswani et al. 2017
- Main features are:
  - Encoder-Decoder architecture
  - No RNN, key-value (self-)attention only
  - Multi-layered attention
  - Skip connections
  - Layer normalization
  - Positional encoding

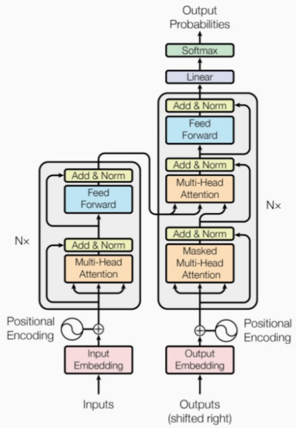


Figure 3: From Vaswani et al. 2017

Key-value attention

- Same as context attention except that:
  - Vector keys $\mathbf{k}_k$ are used instead of $\mathbf{v}_k$ when calculating the score
  - Context vectors that are now called query vectors $\mathbf{q}_j$

$$V^T = [\mathbf{v}_1, \dots, \mathbf{v}_n] \quad \xrightarrow[K^T = [\mathbf{k}_1, \dots, \mathbf{k}_n], \theta_k]{Q^T = [\mathbf{q}_1, \dots, \mathbf{q}_m], \theta_q} \quad Y^T = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_m]$$

- Attention score is then

$$s_{jk} = s_{jk}(\mathbf{k}_k, \mathbf{q}_j, \theta)$$

- Differentiable database

Attention scores

- Additive attention (basically a 2-layers MLP in Bahdanau, Cho, and Bengio 2016)

$$a_{jk} = \boldsymbol{w}_a \cdot \tanh\left(W_q \mathbf{q}_j + W_k \mathbf{k}_k\right)$$

- Bilinear attention in Luong, Pham, and Manning 2015

$$s_{jk} = \mathbf{q}_j W \mathbf{k}_k$$

- Dot-product attention score Luong, Pham, and Manning 2015

$$s_{jk} = \mathbf{q}_j \cdot \mathbf{k}_k$$

- Scaled dot-product attention score in Vaswani et al. 2017

$$s_{jk} = \frac{\mathbf{q}_j \cdot \mathbf{k}_k}{\sqrt{d}}$$

Key-value attention: example

- Values

$$\mathbf{v}_1 = \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix}, \quad \mathbf{v}_2 = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix}, \quad \mathbf{v}_3 = \begin{pmatrix} 0 \\ 5 \\ 1 \end{pmatrix} \quad \text{so that} \quad V = \begin{pmatrix} 2 & 3 & 1 \\ 2 & -1 & 0 \\ 0 & 5 & 1 \end{pmatrix},$$

- Keys

$$\mathbf{k}_1 = \begin{pmatrix} 2 \\ 1 \\ -1 \end{pmatrix}, \quad \mathbf{k}_2 = \begin{pmatrix} 0 \\ 3 \\ -1 \end{pmatrix}, \quad \mathbf{k}_3 = \begin{pmatrix} 1 \\ 1 \\ 3 \end{pmatrix} \quad \text{so that} \quad K = \begin{pmatrix} 2 & 1 & -1 \\ 0 & 3 & -1 \\ 1 & 1 & 3 \end{pmatrix},$$

- Same number of keys and values
- Not necessarily same dimension

Key-value attention: example

- Query

$$\mathbf{q}_1 = \begin{pmatrix} 2 \\ -1 \\ 0 \end{pmatrix}, \quad \mathbf{q}_2 = \begin{pmatrix} -2 \\ 1 \\ 4 \end{pmatrix} \quad \text{so that} \quad Q = \begin{pmatrix} 2 & -1 & 0 \\ -2 & 1 & 4 \end{pmatrix},$$

- Any number of query but same dimension as that of keys
- Dot-product attention: $s_{jk} = \langle \mathbf{k}_k, \mathbf{q}_j \rangle$ so that the score matrix is $S = QK^T$

$$Q = \begin{pmatrix} 2 & -1 & 0 \\ -2 & 1 & 4 \end{pmatrix}, \quad K = \begin{pmatrix} 2 & 1 & -1 \\ 0 & 3 & -1 \\ 1 & 1 & 3 \end{pmatrix}, \quad \text{so that} \quad QK^T = \begin{pmatrix} 3 & -3 & 1 \\ -7 & -1 & 11 \end{pmatrix},$$

Key-value attention: example

- Softmax on rows of $S = QK^T$

$$A = \begin{pmatrix} 0.879 & 0.002 & 0.119 \\ 0.0 & 0.0 & 1.0 \end{pmatrix}$$

- From the attention matrix A, $\mathbf{v}_1$ and $\mathbf{v}_3$ are approximately selected

$$X' = AV = \begin{pmatrix} 1.762 & 3.23 & 0.998 \\ 0.0 & 5.0 & 1.0 \end{pmatrix}, \qquad V = \begin{pmatrix} 2 & 3 & 1 \\ 2 & -1 & 0 \\ 0 & 5 & 1 \end{pmatrix}$$

Self-attention

How to calculate context or key and query vectors?
Answer: we use the content itself!

- Context-based self attention: context is calculated from $\mathbf{v}_j$ itself!

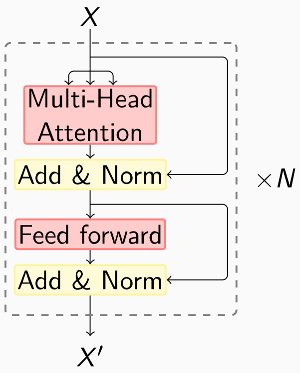
$$\boldsymbol{c}_j = f(\mathbf{v}_j, \theta_j)$$

- Key-value self attention: both key and query vectors are calculated from $\mathbf{v}_j$ itself!

$$\mathbf{k}_j = f^k\left(\mathbf{v}_j, \theta_j^k\right) \qquad \mathbf{q}_j = f^q\left(\mathbf{v}_j, \theta_j^q\right)$$

Transformer encoder block

- Main features
  - Key-value self attention
  - Layer normalization
  - Multi-head attention
  - Skip connections
  - Transformer blocks are stacked



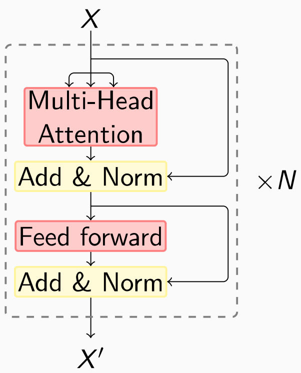
Key-value self-attention

- $X \in \mathbb{R}^{n \times p}$ contains a sequence of n tokens embedded in $\mathbb{R}^p$

$$X^T = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]$$

- Keys, values and queries are created from X,
  - $V = XW_V$ with $W_V \in \mathbb{R}^{p \times d_v}$
  - $Q = XW_Q$ with $W_Q \in \mathbb{R}^{p \times d_k}$
  - $K = XW_K$ with $W_K \in \mathbb{R}^{p \times d_k}$
- Simple key-value self-attention

$$T = \mathrm{Attention}\,(V, K, Q) = \mathrm{Softmax}^{\leftrightarrow}\left(\frac{QK^T}{\sqrt{d_k}}\right) \cdot V$$



Layer normalization from Ba, Kiros, and Hinton 2016

- Standardizing each token representation

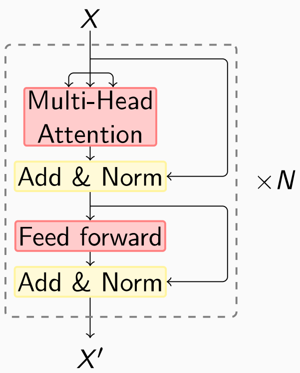
$$\mathrm{LayerNorm}\,(\boldsymbol{x}_t) = \gamma \odot \frac{\boldsymbol{x}_t - \mu_t}{\sigma_t} + \boldsymbol{b}$$

- Rescaling parameters $\gamma, \boldsymbol{b} \in \mathbb{R}^p$
- μ_t the sample mean of $\boldsymbol{x}_t$:

$$\mu_t = \frac{1}{p}\sum_{i=0}^{p-1}(\boldsymbol{x}_t)_i$$

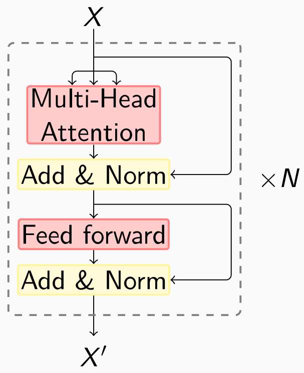
- σ_t the sample standard deviation of $\boldsymbol{x}$:

$$\sigma_t = \sqrt{\frac{1}{p}\sum_{i=0}^{p-1}\left((\boldsymbol{x}_t)_i - \mu_t\right)^2}$$



Feed forward

- Position-wise 2-layers FFN
- Same parameters for each position
- Different parameters between transformer encoder block



Summary: single head transformer block

A transformer block is a map $X \in \mathbb{R}^{n\times p} \mapsto X' \in \mathbb{R}^{n\times p}$

- Keys, values and queries are created from X, $W_Q, W_K \in \mathbb{R}^{p\times d_k}$, $W_V \in \mathbb{R}^{p\times d_v}$
$$Q = XW_Q \in \mathbb{R}^{n\times d_k}, \qquad K = XW_K \in \mathbb{R}^{n\times d_k}, \qquad V = XW_V \in \mathbb{R}^{n\times d_v}$$
- Attention matrix
$$T = \mathrm{Attention}\left(Q, K, V\right) = \mathrm{Softmax}^{\leftrightarrow}\left(\frac{QK^T}{\sqrt{d_k}}\right)\cdot V \in \mathbb{R}^{n\times d_v}$$
- Residual mapping and layer normalization, $W_0 \in \mathbb{R}^{d_v\times p}$
$$U = \mathrm{LayerNorm}\left(TW_0 + X\right) \in \mathbb{R}^{n\times p}$$
- Position-wise FFN, $W_1 \in \mathbb{R}^{p\times d_{ff}}$, $W_2 \in \mathbb{R}^{d_{ff}\times p}$
$$Z = \mathrm{ReLU}\left(UW_1\right)W_2 \in \mathbb{R}^{n\times p}$$
- Residual mapping and layer normalization
$$X' = \mathrm{LayerNorm}\left(Z + U\right) \in \mathbb{R}^{n\times p}$$

Multi-head transformer block

- X is first split into H pieces
$$X = [X_1, \dots, X_H], \quad X_i \in \mathbb{R}^{n\times p_h} \text{ with } H\cdot p_h = p$$
- Keys, values and queries are created from each X_h, $W_Q^{(h)}, W_K^{(h)} \in \mathbb{R}^{p_h\times d_{k_h}}$, $W_V^{(h)} \in \mathbb{R}^{p_h\times d_{v_h}}$
$$Q^{(h)} = X_h W_Q^{(h)} \in \mathbb{R}^{n\times d_{k_h}}, \qquad K^{(h)} = X_h W_K^{(h)} \in \mathbb{R}^{n\times d_{k_h}}, \qquad V^{(h)} = X_h W_V^{(h)} \in \mathbb{R}^{n\times d_{v_h}}$$
- Multi-head attention matrix
$$T^{(h)} = \mathrm{Attention}\left(Q^{(h)}, K^{(h)}, V^{(h)}\right) = \mathrm{Softmax}^{\leftrightarrow}\left(\frac{Q^{(h)}\left(K^{(h)}\right)^T}{\sqrt{d_{k_h}}}\right)\cdot V^{(h)} \in \mathbb{R}^{n\times d_{v_h}}$$
$$T = \left[T^{(1)}, \dots, T^{(H)}\right] \in \mathbb{R}^{n\times d_v} \quad \text{with } d_v = H\cdot d_{v_h}$$

Positional encoding

- Equivariant to permutation of samples:
  - If X gives Y, $X_\sigma = P_\sigma X$ gives $Y_\sigma = P_\sigma X$
  - We can check that $X'_\sigma = P_\sigma X'$
- Temporal information is not taken into account
- Two different stategies:
  - Learnable positional encoding
    - Enlarge embedding with parameters tied to position
    - Problem at test time with unseen positions during train time
  - Non-learnable positional encoding
    - No extra parameters
    - Defined for all length of sequence even if it has not been seen is the train set.

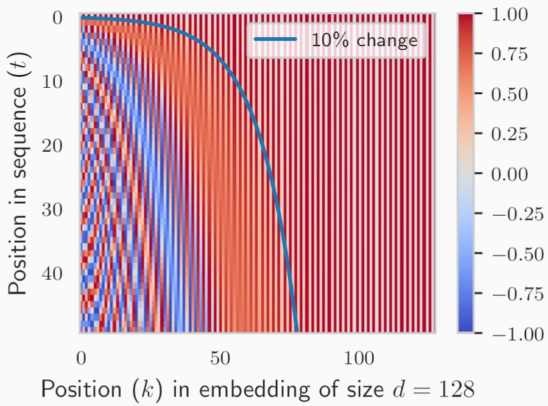
Positional encoding

- $\boldsymbol{e}_1, \ldots \boldsymbol{e}_n$ is a sequence of vectors in $\mathbb{R}^d$
- positional encoding $\mathbf{p}_t \in \mathbb{R}^d$ of $\boldsymbol{e}_t$ is

$$p_t^k = \begin{cases} \sin 2\pi t / T_{2i} & \text{if } k = 2i \\ \cos 2\pi t / T_{2i} & \text{if } k = 2i+1 \end{cases}$$

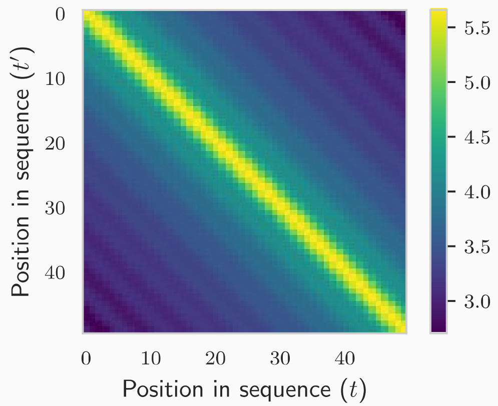
where $T_k = 2\pi \cdot 10000^{k/d}$

- Input after positional encoding is $\boldsymbol{e}_t + \mathbf{p}_t$ instead of $\boldsymbol{e}_t$



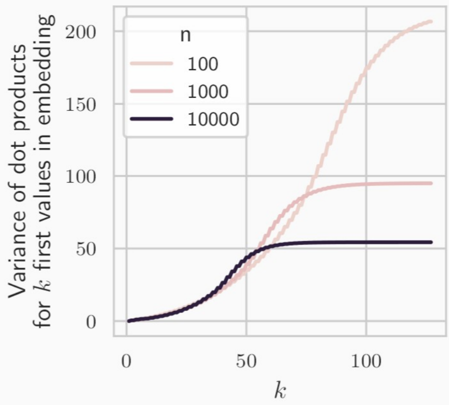
Translation invariance

- $\langle p_t, p_{t'} \rangle$ only depends on $|t - t'|$
- $$\langle p_t, p_{t'} \rangle = \sum_{k \text{ even}} \sin\left(\frac{2\pi t}{T_k}\right) \sin\left(\frac{2\pi t'}{T_k}\right) + \cos\left(\frac{2\pi t}{T_k}\right) \cos\left(\frac{2\pi t'}{T_k}\right) = \sum_{k \text{ even}} \cos\left(\frac{2\pi (t - t')}{T_k}\right) = \sum_{k \text{ even}} \cos\left(\frac{2\pi\, |t - t'|}{T_k}\right)$$
- $\mathbf{p}_{t+k}$ is a rotation of $\mathbf{p}_t$ that only depends on k



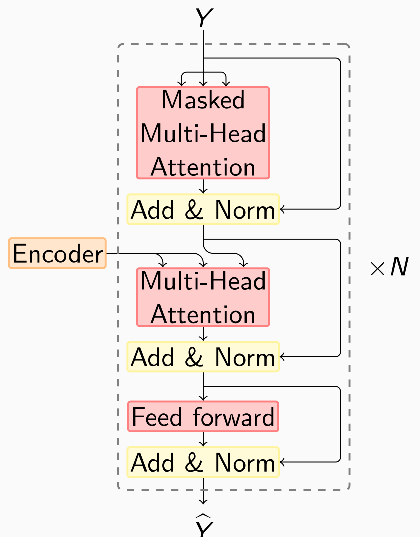
Variance of positional encoding

- Plot of $\mathrm{Var}_{|t-t'| \sim \mathcal{U}(50)} \left\langle \mathbf{p}_t^{1\ldots k}, \mathbf{p}_{t'}^{1\ldots k} \right\rangle$ w.r.t k
- Position information is encoded in first $\simeq 50$ elements with $d = 128$



Transformer decoder block

- Main features
  - Masked key-value self attention
  - Layer normalization
  - Skip connections
  - Transformer blocks are stacked
- Two main differences
  - Attention matrix is masked out to prevent receiving attention from future
  - Keys and values come from the encoder and decoder make queries on these



Masked multi-head attention

- What is the loss governing the whole architecture?

Source	<bos>	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	$\dots$	$\boldsymbol{y}_{m-2}$	$\boldsymbol{y}_{m-1}$
	$\downarrow$	$\downarrow$	$\downarrow$	$\dots$	$\downarrow$	$\downarrow$
Prediction	$\widehat{\boldsymbol{y}}_1$	$\widehat{\boldsymbol{y}}_2$	$\widehat{\boldsymbol{y}}_3$	$\dots$	$\widehat{\boldsymbol{y}}_{m-1}$	$\widehat{\boldsymbol{y}}_m$
Target	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	$\boldsymbol{y}_3$	$\dots$	$\boldsymbol{y}_{m-1}$	<eos>

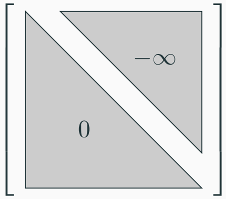
- Loss on couples $(\widehat{\boldsymbol{y}}_1, \boldsymbol{y}_1), \dots, (\widehat{\boldsymbol{y}}_{m-1}, \boldsymbol{y}_{m-1})$ and $(\widehat{\boldsymbol{y}}_m, \text{<eos>})$
- But $\widehat{\boldsymbol{y}}_1$ should not be able to receive attention from $\boldsymbol{y}_2$!
- Masked attention prevents tokens from receiving attention from future ones

Masked multi-head attention

- How to prevent tokens from receiving attention from future ones?
- Change attention scores $S = QK^T$ so that $\mathrm{Softmax}$ ignore them
  - $\mathrm{Softmax}(2, 2, -\infty) = (1/2, 1/2, 0)$
  - $\mathrm{Softmax}(1, 1, 1, -\infty) = (1/3, 1/3, 1/3, 0)$

$$\mathrm{Attention}(Q, K, V) = \mathrm{Softmax}^{\leftrightarrow}\left(\frac{QK^T + M}{\sqrt{d_k}}\right) \cdot V \in \mathbb{R}^{m \times d_v}$$

$$M_{jk} = \begin{cases} -\infty & \text{if } j < k \\ 0 & \text{otherwise} \end{cases} \qquad M =$$



- M is setting scores to $-\infty$ or leaving them unchanged
- We only allow queries (j) on past keys (k): $k \leqslant j$

Single-head decoder block I

A decoder transformer block is a map $Y \in \mathbb{R}^{m \times p} \mapsto \widehat{Y} \in \mathbb{R}^{m \times p}$

- Keys, values and queries are created from Y, $W_{V_1} \in \mathbb{R}^{p \times d_v}$, $W_{Q_1}, W_{K_1} \in \mathbb{R}^{p \times d_k}$
  $$Q_1 = YW_{Q_1} \in \mathbb{R}^{m \times d_k}, \qquad K_1 = YW_{K_1} \in \mathbb{R}^{m \times d_k}, \qquad V_1 = YW_{V_1} \in \mathbb{R}^{m \times d_v}$$
- Masked attention matrix
  $$T_1 = \mathrm{Attention}(Q_1, K_1, V_1) = \mathrm{Softmax}^{\leftrightarrow}\left(\frac{Q_1 K_1^T + M}{\sqrt{d_k}}\right) \cdot V_1 \in \mathbb{R}^{m \times d_v}$$
- Residual mapping and layer normalization, $W_0 \in \mathbb{R}^{d_v \times p}$
  $$U_1 = \mathrm{LayerNorm}(T_1 W_0 + Y) \in \mathbb{R}^{m \times p}$$
- Query of decoder on keys and values of the encoder
  $$Q_2 = U_1 W_{Q_2} \in \mathbb{R}^{m \times d_k}, \qquad K_2 = X' W_{K_2} \in \mathbb{R}^{n \times d_k}, \qquad V_2 = X' W_{V_2} \in \mathbb{R}^{n \times d_v}$$
- Cross-attention with attention matrix $A \in \mathbb{R}^{m \times n}$
  $$T_2 = \mathrm{Attention}(Q_2, K_2, V_2) = \mathrm{Softmax}^{\leftrightarrow}\left(\frac{Q_2 K_2^T}{\sqrt{d_k}}\right) \cdot V_2 \in \mathbb{R}^{m \times d_v}$$

Single-head decoder block II

- Residual mapping and layer normalization, $W_1 \in \mathbb{R}^{d_v \times p}$
  $$U_2 = \mathrm{LayerNorm}(T_2 W_1 + U_1) \in \mathbb{R}^{m \times p}$$
- Position-wise FFN, $W_2 \in \mathbb{R}^{p \times d_{\mathrm{ff}}}$, $W_3 \in \mathbb{R}^{d_{\mathrm{ff}} \times p}$
  $$Z = \mathrm{ReLU}(U_2 W_2) W_3 \in \mathbb{R}^{m \times p}$$
- Residual mapping and layer normalization,
  $$\widehat{Y} = \mathrm{LayerNorm}(Z + U_2) \in \mathbb{R}^{m \times p}$$
- Set of m distributions on tokens, $W \in \mathbb{R}^{p \times N}$
  $$\mathrm{Softmax}^{\leftrightarrow}\left(\widehat{Y} W\right)$$

References i

[1] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. "Effective approaches to attention-based neural machine translation." 2015. arXiv: 1508.04025.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." 2016. arXiv: 1607.06450.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate." May 19, 2016. arXiv: 1409.0473 [cs, stat]. URL: http://arxiv.org/abs/1409.0473 (visited on 10/14/2021).

[4] Kelvin Xu et al. "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention." Apr. 19, 2016. arXiv: 1502.03044 [cs]. URL: http://arxiv.org/abs/1502.03044 (visited on 12/15/2021).

[5] Ashish Vaswani et al. "Attention is all you need." In: *Advances in neural information processing systems*. 2017, pp. 5998–6008. arXiv: 1706.03762.

[6] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." June 3, 2021. arXiv: 2010.11929 [cs]. URL: http://arxiv.org/abs/2010.11929 (visited on 12/15/2021).