CS-568 Deep Learning

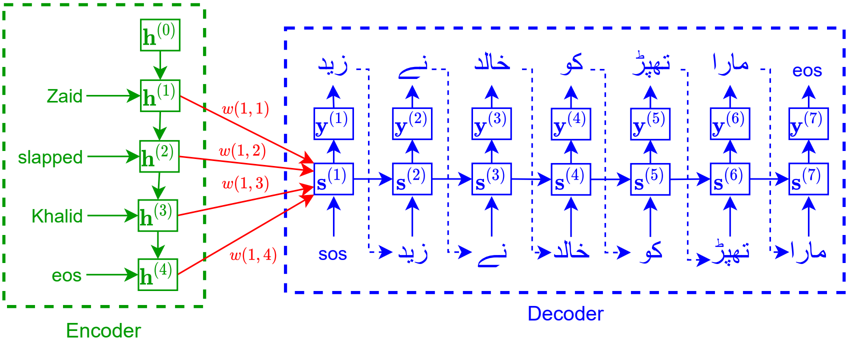
Nazar Khan

PUCIT

Transformers

Previously: Decoder with attention

- An attention-based decoder decides which part of the input encoding to focus on.



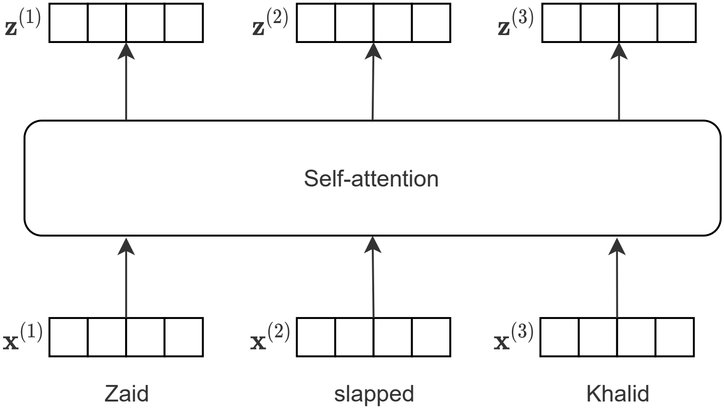
- Decoding emphasizes different parts of the encodings.
- In this lecture, we will study how *encodings can be computed with attention* as well.

This lecture: Encoder with attention aka Transformer[1]

- A sequence-to-sequence model without convolution and without recurrence.
- Recurrence is a sequential process – cannot be parallelized.
- Transformer contains parallelizable modules and can therefore be trained faster.
- Transformers achieve state-of-the-art performance on sequence modelling tasks.

[1] Ashish Vaswani et al. 'Attention is All You Need'. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS'17. Long Beach, California, USA, 2017, 6000–6010.

Self-attention



We will assume 512-dimensional input embeddings $\mathbf{x}^{(t)}$ as well as 512-dimensional encodings $\mathbf{z}^{(t)}$.

Self-attention

1. Place embeddings of all words in a matrix $E \in \mathbb{R}^{512 \times T^{\text{in}}}$.
2. Consider 3 *learnable* matrices $W_Q, W_K \in \mathbb{R}^{64 \times 512}$ and $W_V \in \mathbb{R}^{512 \times 512}$ and apply linear transformations

$$Q = W_Q E \in \mathbb{R}^{64 \times T^{\text{in}}}$$
$$K = W_K E \in \mathbb{R}^{64 \times T^{\text{in}}}$$
$$V = W_V E \in \mathbb{R}^{512 \times T^{\text{in}}}$$

   to each word. *Parallelizable in time*.
3. Compute similarity scores between the representations in Q and K.

$$S = \text{row-wise softmax}\left(\frac{Q^T K}{\sqrt{64}}\right) \in \mathbb{R}^{T^{\text{in}} \times T^{\text{in}}}$$

Self-attention

4. Compute the encoding of each word

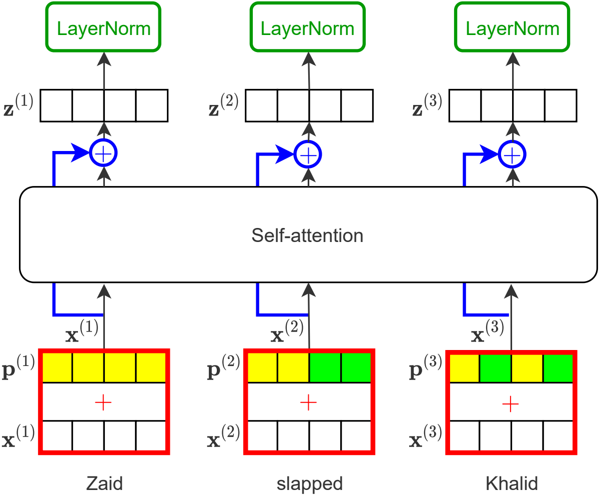
$$Z = VS^T \in \mathbb{R}^{512 \times T^{\text{in}}}$$

where each column of Z is a 512-dimensional encoding of the corresponding word.

> Note that *each word has now been encoded by attending to all words in the sentence*.

> The scaled dot-product scores in S are the attention weights.

Self-attention
Additional details



Multi-headed attention

- Replicate 8 self-attention modules, each with its own learnable matrices W_{Qi}, W_{Ki}, W_{Vi}.
- Compute encodings $Z_1, \ldots, Z_8$.
- Compute final encoding Z by concatenating the Z_i and projecting onto 512-dimensional space using another learnable matrix $W_O \in \mathbb{R}^{512\times(512*8)}$.

$$Z = W_O \begin{bmatrix} Z_1 \\ Z_2 \\ \vdots \\ Z_8 \end{bmatrix}$$

- This way, the model can learn 8 different ways of encoding the input sentence.

Feed-forward NN

- Pass each encoding in Z through the same 2-layer network.

$$E = W_2 * ReLU(W_1 Z + \mathbf{b}_1 \mathbf{1}^T) + \mathbf{b}_2 \mathbf{1}^T$$

  where W_1 has 2048 rows and W_2 has 512 rows.
- Add residual connection.

$$E = W_2 * ReLU(W_1 Z + \mathbf{b}_1 \mathbf{1}^T) + \mathbf{b}_2 \mathbf{1}^T + Z$$

- Perform LayerNorm on each column of E.
- *Parallelizable in time*.

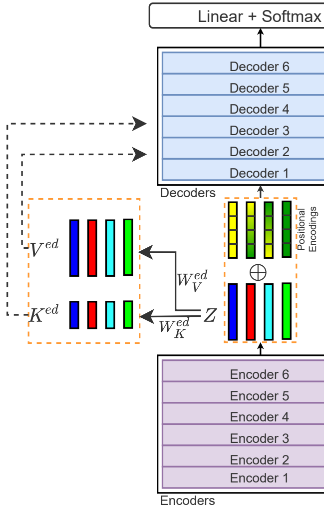
Stacked Encoders

- An encoder involves the transformation

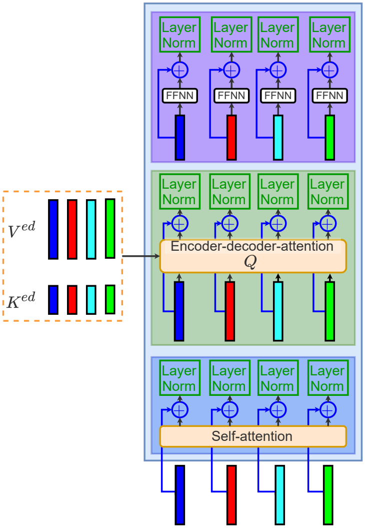
$$\text{Embeddings} \longrightarrow \text{Self-attention} \longrightarrow \text{FFNN} \longrightarrow \text{Encodings}$$

- Encoders can be stacked on top of each other.
- Encoding produced by one encoder becomes the input embedding for the next encoder.
- Final encoded output is the result of the last encoder.

From Encoder to Decoder



Inside a Decoder



Decoder and the future

Self-attention in decoder attends only to the words generated so far in the output sequence. Achieved by setting future times to $-\infty$ in the softmax.