CS-568 Deep Learning

Nazar Khan

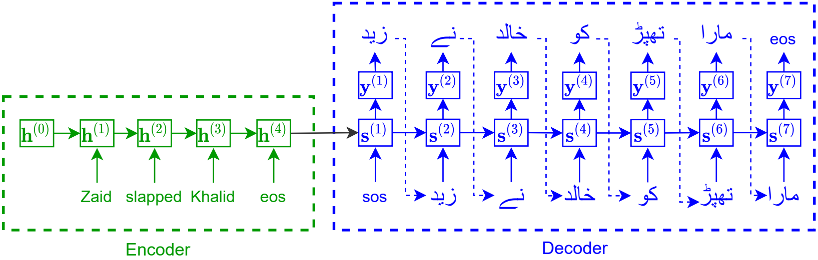
PUCIT

Attention Models

Decoder

Where does it look?

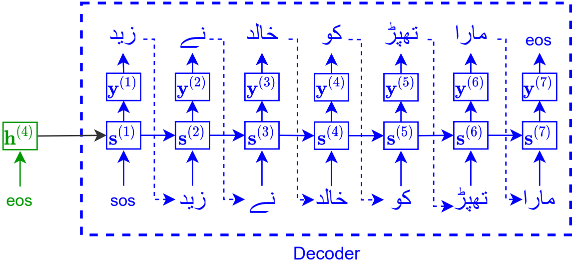
- A standard decoder uses the last hidden state produced by an encoder as its recurrent input.



Decoder

Where does it look?

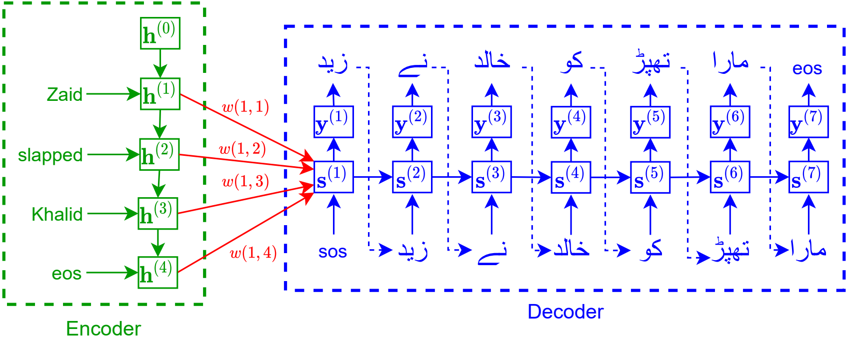
- Interpretation: decoder *looks at* the last input that produced the last hidden state.



Decoder

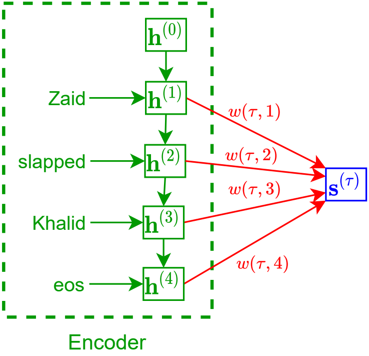
Where does it look?

- The decoder can be made to look at *all hidden states* in the encoder.
- Interpretation: decoder will then *look at* every input.
- Decoder can look at each input in a weighted fashion.



Decoder
Where does it look?

- Weights can be specific to each decoding step τ.



Decoder with attention

- For clarity,
  - T_n^{in}: number of words (time steps) in n-th input sample.
  - $\mathbf{h}^{(t)}$: hidden state in encoder
  - $\mathbf{s}^{(\tau)}$: hidden state in decoder
- Decoder can be made to look at all hidden states of the encoder.
  1. Replace $\mathbf{h}^{(T_n^{\text{in}})}$ by a weighted sum of all encodings $\mathbf{h}^{(1)}, \mathbf{h}^{(2)}, \ldots, \mathbf{h}^{(T_n^{\text{in}})}$.
  2. Feed weighted sum of encodings to *each* state $\mathbf{s}^{(\tau)}$.
  3. Weights change for each time step.

$$\begin{tikzcd}
\mathbf{s}^{(\tau-1)} \arrow[dr] & \\
& \mathbf{s}^{(\tau)} \\
\sum_{t=1}^{T_n^{\text{in}}} w(\tau, t)\mathbf{h}^{(t)} \arrow[ur] &
\end{tikzcd}$$

How to compute attention?

- Make $w(\tau, t)$ depend on $\mathbf{s}^{(\tau-1)}$ and $\mathbf{h}^{(t)}$.
- To ensure *weighted average*, compute $w(\tau, t)$ via softmax to produce probability values.

$$w(\tau, t) = \frac{\exp\left(u(\tau, t)\right)}{\sum_{j=1}^{T_n^{\text{in}}} \exp\left(u(\tau, j)\right)}$$

Options for computing unnormalized weights $u(\tau, t)$

1. Favour input encoding similar to decoder state.

$$u(\tau, t) = \mathbf{h}^{(t)} \cdot \mathbf{s}^{(\tau-1)}$$

2. If encoder and decoder states have different sizes, use a *learnable* projection matrix.

$$u(\tau, t) = \mathbf{h}^{(t)} \cdot \left(W_a \mathbf{s}^{(\tau-1)} \right)$$

3. Use a single hidden-layer network with a single linear output neuron.

$$u(\tau, t) = \mathbf{v}_a^T \tanh\left(W_a \begin{bmatrix} \mathbf{h}^{(t)} \\ \mathbf{s}^{(\tau-1)} \end{bmatrix} \right)$$

4. Use an MLP with a single linear output neuron.

$$u(\tau, t) = MLP\left(\begin{bmatrix} \mathbf{h}^{(t)} \\ \mathbf{s}^{(\tau-1)} \end{bmatrix} \right)$$

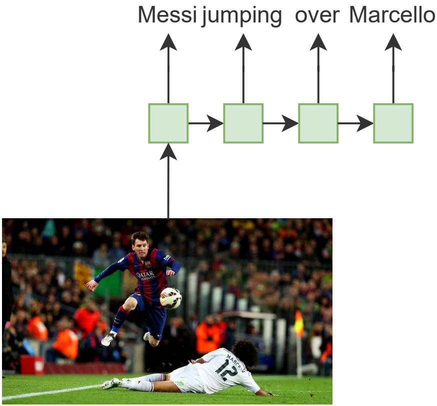
Options 2, 3 and 4 correspond to learning a model for computing attention.

The Encoder-Attention-Decoder Model

Training of all 3 modules (encoder-attention-decoder) takes place jointly.

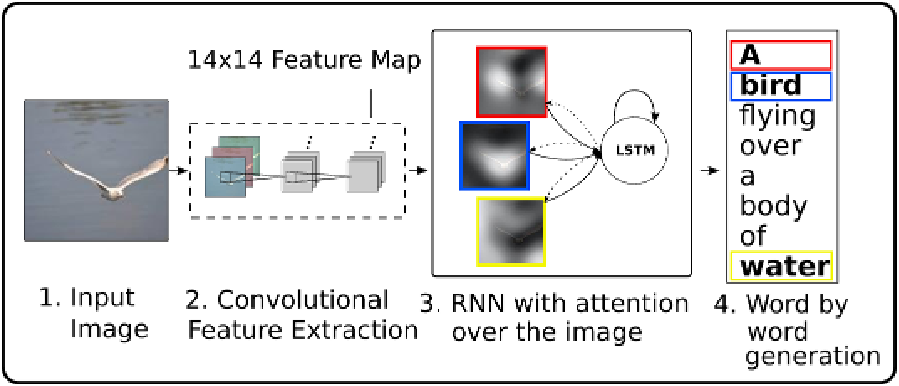
$$E(\theta_E) \longrightarrow A(\theta_A) \longrightarrow D(\theta_D) \longrightarrow \mathcal{L}$$
$$\nabla_{\theta_E}\mathcal{L} \longleftarrow \nabla_{\theta_A}\mathcal{L} \longleftarrow \nabla_{\theta_D}\mathcal{L} \longleftarrow \mathcal{L}$$

Image Captioning



Attention-based Decoder for Image Captioning[1]

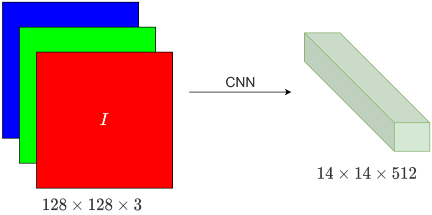
- Attention based model that automatically learns to describe the content of images.



[1] Kelvin Xu et al. 'Show, attend and tell: Neural image caption generation with visual attention'. In: *International conference on machine learning*. PMLR. 2015, pp. 2048–2057.

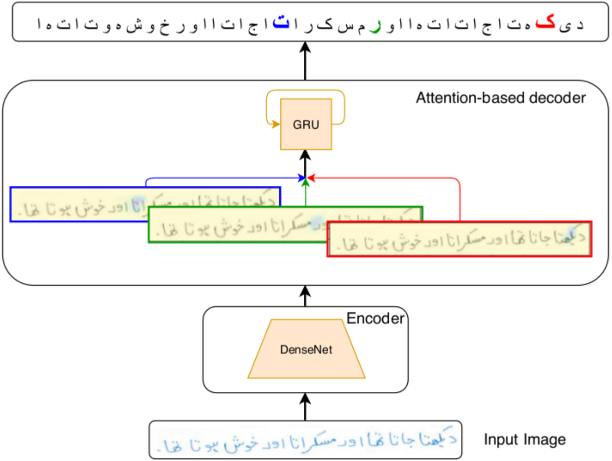
Attention-based Decoder for Image Captioning

- Feature volume computed through a CNN can be used as initial hidden state $\mathbf{s}^{(0)}$ of the decoder.



- The CNN is the encoder.
- Each pixel in $\mathbf{s}^{(0)}$ represents some portion of the input image.
- Attention weight $w(\tau, i, j)$ represents the importance of image region i, j in producing the decoded output at time τ.

Attention-based Decoder for Handwritten Urdu Recognition[2]



[2] Tayaba Anjum and Nazar Khan. 'An attention based method for offline handwritten Urdu text recognition'. In: *International Conference on Frontiers in Handwriting Recognition (ICFHR)*. 2020.

Summary

- Traditional decoders use the final encoded state as their initial hidden state.
- Attention-based decoders use weighted-average of all encoded hidden states.
- By allowing weights to change at each decoding step, the decoder can focus on different parts of the input as it decodes.