Neural Machine Translation

Anoop Sarkar
anoopsarkar.github.io/neuralmt-class

Simon Fraser University

November 27, 2017

Neural Machine Translation

Anoop Sarkar
anoopsarkar.github.io/neuralmt-class

Simon Fraser University

Part 1: Positional Encoding

Attention is all you need

Attention is all you need

Vaswani+ arXiv:1706.03762v4 Jun 2017

- No recurrent networks

Attention is all you need

Vaswani+ arXiv:1706.03762v4 Jun 2017

- No recurrent networks
- No convolutional networks

Attention is all you need

Vaswani+ arXiv:1706.03762v4 Jun 2017

- No recurrent networks
- No convolutional networks
- Use feedforward networks

Attention is all you need

Vaswani+ arXiv:1706.03762v4 Jun 2017

- No recurrent networks
- No convolutional networks
- Use feedforward networks
- Use positional encoding

Attention is all you need

Vaswani+ arXiv:1706.03762v4 Jun 2017

- No recurrent networks
- No convolutional networks
- Use feedforward networks
- Use positional encoding
- Layer wise rescaling (Ba+, arXiv 2016)

Attention is all you need

Vaswani+ arXiv:1706.03762v4 Jun 2017

- No recurrent networks
- No convolutional networks
- Use feedforward networks
- Use positional encoding
- Layer wise rescaling (Ba+, arXiv 2016)
- Use softmax for output

Attention is all you need

Vaswani+ arXiv:1706.03762v4 Jun 2017

- No recurrent networks
- No convolutional networks
- Use feedforward networks
- Use positional encoding
- Layer wise rescaling (Ba+, arXiv 2016)
- Use softmax for output
- (multi-head) Attention is all you need

Attention is all you need

Vaswani+ arXiv:1706.03762v4 Jun 2017

- No recurrent networks
- No convolutional networks
- Use feedforward networks
- Use positional encoding
- Layer wise rescaling (Ba+, arXiv 2016)
- Use softmax for output
- (multi-head) Attention is all you need
- Any questions?

Notation

- Input is a sequence of symbols: $(w_1, w_2, \ldots, w_n)$

Notation

- Input is a sequence of symbols: $(w_1, w_2, \ldots, w_n)$
- Model creates sequence of continuous representations: $(z_1, z_2, \ldots, z_n)$

Notation

- Input is a sequence of symbols: $(w_1, w_2, \ldots, w_n)$
- Model creates sequence of continuous representations: $(z_1, z_2, \ldots, z_n)$
- Output sequence of symbols: $(o_1, o_2, \ldots, o_m)$

Layer Normalization

Ba+ arXiv 2016

- In a multi-layer model:

$$\mathbf{h}_t^{\ell} = W_{hh}\mathbf{h}_t^{\ell-1} + W_{xh}\mathbf{x}_t$$

Layer Normalization

Ba+ arXiv 2016

- In a multi-layer model:

$$\mathbf{h}_t^{\ell} = W_{hh}\mathbf{h}_t^{\ell-1} + W_{xh}\mathbf{x}_t$$

- Instead of this, let us use an intermediate value **a**

$$\mathbf{a}_t^{\ell} = W_{hh}\mathbf{h}_t^{\ell-1} + W_{xh}\mathbf{x}_t$$

Layer Normalization

Ba+ arXiv 2016

- In a multi-layer model:

$$\mathbf{h}_t^{\ell} = W_{hh}\mathbf{h}_t^{\ell-1} + W_{xh}\mathbf{x}_t$$

- Instead of this, let us use an intermediate value **a**

$$\mathbf{a}_t^{\ell} = W_{hh}\mathbf{h}_t^{\ell-1} + W_{xh}\mathbf{x}_t$$

- Take the mean of **a**

$$\mu^{\ell} = \frac{1}{H}\sum_{i=1}^{H} a_i^{\ell}$$

Layer Normalization

Ba+ arXiv 2016

- In a multi-layer model:

$$\mathbf{h}_t^\ell = W_{hh}\mathbf{h}_t^{\ell-1} + W_{xh}\mathbf{x}_t$$

- Instead of this, let us use an intermediate value **a**

$$\mathbf{a}_t^\ell = W_{hh}\mathbf{h}_t^{\ell-1} + W_{xh}\mathbf{x}_t$$

- Take the mean of **a**

$$\mu^\ell = \frac{1}{H}\sum_{i=1}^{H} a_i^\ell$$

- Take the variance of **a**

$$\sigma^\ell = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(a_i^\ell - \mu^\ell)^2}$$

Layer Normalization

Ba+ arXiv 2016

- Introduce two new hyperparameter vectors $\mathbf{b}$ (bias) and $\mathbf{g}$ (gain). Same dimension as $\mathbf{h}_t$.

Layer Normalization

Ba+ arXiv 2016

- Introduce two new hyperparameter vectors $\mathbf{b}$ (bias) and $\mathbf{g}$ (gain). Same dimension as $\mathbf{h}_t$.
- Compute $\mathbf{h}_t^\ell$ using μ^ℓ, σ^ℓ, $\mathbf{b}$ and $\mathbf{g}$:

$$\mathbf{h}_t^\ell = f\left(\frac{\mathbf{g}}{\sigma^\ell} \odot (\mathbf{a}^\ell - \mu) + \mathbf{b}\right)$$

Layer Normalization

Ba+ arXiv 2016

- Introduce two new hyperparameter vectors $\mathbf{b}$ (bias) and $\mathbf{g}$ (gain). Same dimension as $\mathbf{h}_t$.
- Compute $\mathbf{h}_t^\ell$ using μ^ℓ, σ^ℓ, $\mathbf{b}$ and $\mathbf{g}$:

$$\mathbf{h}_t^\ell = f\left(\frac{\mathbf{g}}{\sigma^\ell} \odot (\mathbf{a}^\ell - \mu) + \mathbf{b}\right)$$

- There is a tendency for the average magnitude of the summed inputs to the recurrent units to either grow or shrink at every time-step, leading to exploding or vanishing gradients

Layer Normalization

Ba+ arXiv 2016

- Introduce two new hyperparameter vectors $\mathbf{b}$ (bias) and $\mathbf{g}$ (gain). Same dimension as $\mathbf{h}_t$.
- Compute $\mathbf{h}_t^\ell$ using μ^ℓ, σ^ℓ, $\mathbf{b}$ and $\mathbf{g}$:

$$\mathbf{h}_t^\ell = f\left(\frac{\mathbf{g}}{\sigma^\ell} \odot (\mathbf{a}^\ell - \mu) + \mathbf{b}\right)$$

- There is a tendency for the average magnitude of the summed inputs to the recurrent units to either grow or shrink at every time-step, leading to exploding or vanishing gradients
- The normalization terms make it invariant to re-scaling all of the summed inputs to a layer

Layer Normalization

Ba+ arXiv 2016

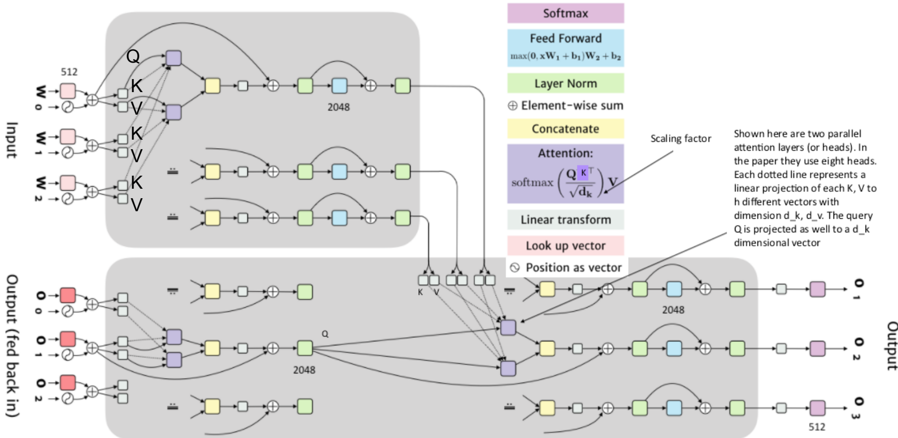
- Introduce two new hyperparameter vectors $\mathbf{b}$ (bias) and $\mathbf{g}$ (gain). Same dimension as $\mathbf{h}_t$.
- Compute $\mathbf{h}_t^\ell$ using μ^ℓ, σ^ℓ, $\mathbf{b}$ and $\mathbf{g}$:

$$\mathbf{h}_t^\ell = f\left(\frac{\mathbf{g}}{\sigma^\ell} \odot (\mathbf{a}^\ell - \mu) + \mathbf{b}\right)$$

- There is a tendency for the average magnitude of the summed inputs to the recurrent units to either grow or shrink at every time-step, leading to exploding or vanishing gradients
- The normalization terms make it invariant to re-scaling all of the summed inputs to a layer
- Results in much more stable hidden-to-hidden dynamics

Attention is all you need

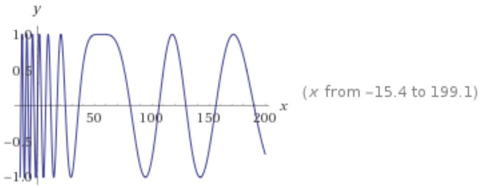
Figure modified from one made by J. Kummerfeld



Attention is all you need

- **Positional encoding** which is a vector of the same length as the word representation. Depends only on position in the input.

$$f(pos, dim) = sin(\frac{pos}{10000^{\frac{2dim}{d_w}}})$$



Attention is all you need: Other details

- When outputs are words, vectors used to represent input words are also used for outputs and in the final linear transformation (with some rescaling).

Attention is all you need: Other details

- When outputs are words, vectors used to represent input words are also used for outputs and in the final linear transformation (with some rescaling).
- They use a new formula for adjusting the learning rate.

Attention is all you need: Other details

- When outputs are words, vectors used to represent input words are also used for outputs and in the final linear transformation (with some rescaling).
- They use a new formula for adjusting the learning rate.
- They use dropout in several places and label smoothing for regularization