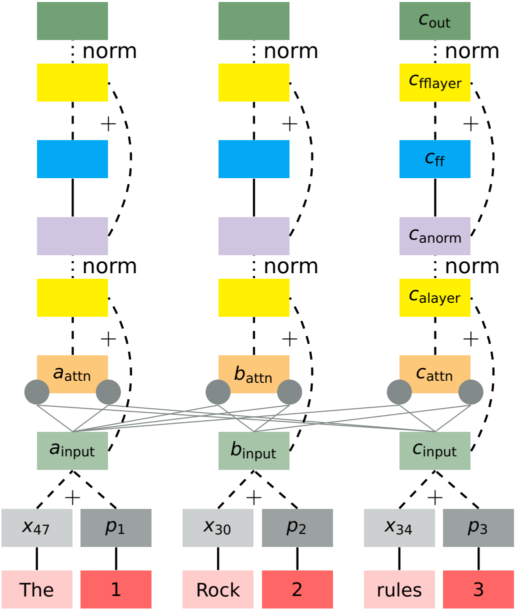
Contextual word representations: Transformers

Christopher Potts

Stanford Linguistics

CS224u: Natural language understanding

Core model structure



$$c_{\text{out}} = \frac{c_{\text{fflayer}} - \textbf{mean}(c_{\text{fflayer}})}{\textbf{std}(c_{\text{fflayer}}) + \varepsilon}$$

$$c_{\text{fflayer}} = c_{\text{anorm}} + \textbf{Dropout}(c_{\text{ff}})$$

$$c_{\text{ff}} = \textbf{ReLU}(c_{\text{anorm}} W_1 + b_1) W_2 + b_2$$

$$c_{\text{anorm}} = \frac{c_{\text{alayer}} - \textbf{mean}(c_{\text{alayer}})}{\textbf{std}(c_{\text{alayer}}) + \varepsilon}$$

$$c_{\text{alayer}} = \textbf{Dropout}\left(c_{\text{attn}} + c_{\text{input}}\right)$$

$$c_{\text{attn}} = \textbf{sum}\left(\left[\alpha_1 a_{\text{input}}, \alpha_2 b_{\text{input}}\right]\right)$$

$$\alpha = \textbf{softmax}(\tilde{\alpha})$$

$$\tilde{\alpha} = \left[\frac{{c_{\text{input}}}^{\top} a_{\text{input}}}{\sqrt{d_k}}, \frac{{c_{\text{input}}}^{\top} b_{\text{input}}}{\sqrt{d_k}}\right]$$

$$c_{\text{input}} = x_{34} + p_3$$

Computing the attention representations

Calculation as previously given

$$c_{\text{attn}} = \textbf{sum}\left(\left[\alpha_1 a_{\text{input}}, \alpha_2 b_{\text{input}}\right]\right)$$
$$\alpha = \textbf{softmax}(\tilde{\alpha})$$
$$\tilde{\alpha} = \left[\frac{{c_{\text{input}}}^{\top} a_{\text{input}}}{\sqrt{d_k}}, \frac{{c_{\text{input}}}^{\top} b_{\text{input}}}{\sqrt{d_k}}\right]$$

Matrix format

$$\textbf{softmax}\left(\frac{c_{\text{input}} \begin{bmatrix} a_{\text{input}} \\ b_{\text{input}} \end{bmatrix}^{\top}}{\sqrt{d_k}}\right) \begin{bmatrix} a_{\text{input}} \\ b_{\text{input}} \end{bmatrix}$$

Computing the attention representations

[1]:
```
import numpy as np
```

[2]:
```
seq_length = 3
d_k = 4
```

[3]:
```
inputs = np.random.uniform(size=(seq_length, d_k))
inputs
```

[3]:
```
array([[0.31436922, 0.66969307, 0.270804  , 0.72023504],
       [0.87180132, 0.27637445, 0.43091867, 0.34138704],
       [0.20292054, 0.6345131 , 0.01058343, 0.22846636]])
```

[4]:
```
a_input = inputs[0]
b_input = inputs[1]
c_input = inputs[2]
```

Computing the attention representations

[5]:
```
def softmax(X):
    z = np.exp(X)
    return (z / z.sum(axis=0)).T
```

[6]:
```
c_alpha = softmax([
    (c_input.dot(a_input) / np.sqrt(d_k)),
    (c_input.dot(b_input) / np.sqrt(d_k))])
```

[7]:
```
c_attn = sum([c_alpha[0]*a_input, c_alpha[1]*b_input])
c_attn
```

[7]:
```
array([0.57768027, 0.48390338, 0.34643646, 0.54128076])
```

[8]:
```
ab = inputs[:-1]
```

[9]:
```
softmax(c_input.dot(ab.T) / np.sqrt(d_k)).dot(ab)
```

[9]:
```
array([0.57768027, 0.48390338, 0.34643646, 0.54128076])
```

[10]:
```
# If we allow every input to attend to itself:
softmax(inputs.dot(inputs.T) / np.sqrt(d_k)).dot(inputs)
```

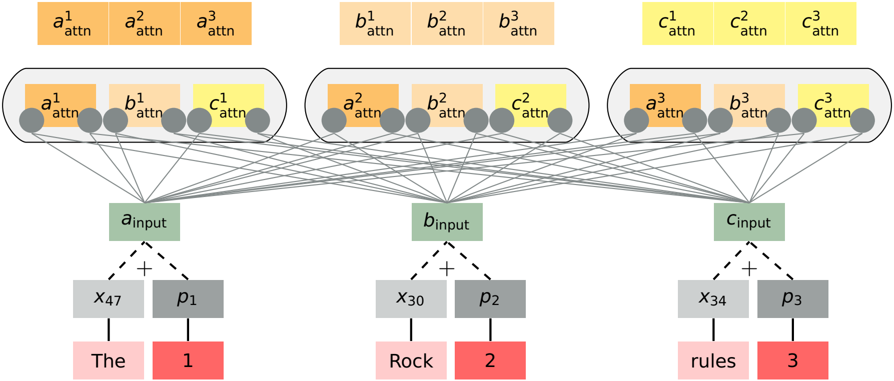
[10]:
```
array([[0.4614388 , 0.53204444, 0.2451212 , 0.45136127],
       [0.50173123, 0.50618272, 0.26184404, 0.43678288],
       [0.45493467, 0.5332328 , 0.23643403, 0.4388242 ]])
```

Multi-headed attention

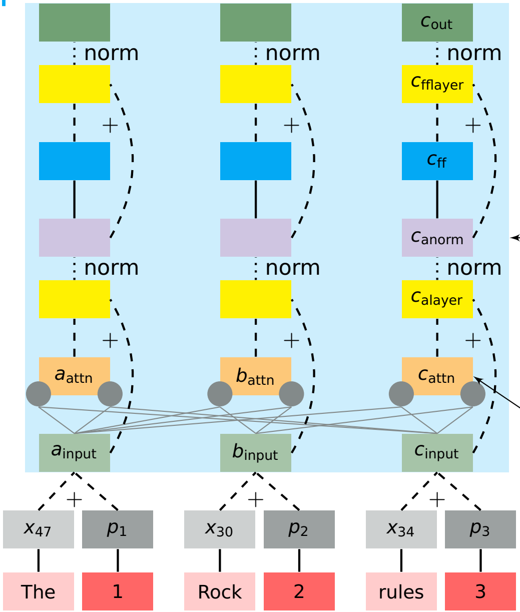
$$c^3_{\text{attn}} = \textbf{sum}\left(\left[\alpha_1(a_{\text{input}} W^V_3), \alpha_2(b_{\text{input}} W^V_3)\right]\right)$$

$$\alpha = \textbf{softmax}(\tilde{\alpha})$$

$$\tilde{\alpha} = \left[\frac{(c_{\text{input}} W^Q_3)^\top (a_{\text{input}} W^K_3)}{\sqrt{d_k}}, \frac{(c_{\text{input}} W^Q_3)^\top (b_{\text{input}} W^K_3)}{\sqrt{d_k}}\right]$$



Repeated transformer blocks



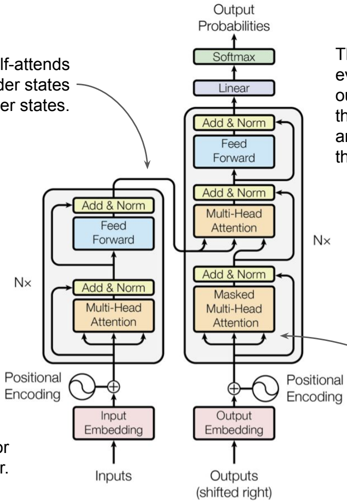
Repeated N times with c_{out} serving as c_{input} at each successive layer.

Includes multi-headed attention in each block

The architecture diagram

Each decoder state self-attends with all of its fellow decoder states and with all the encoder states.

The right side is repeated for every decoder state, with outputs for each state that has them (all of them for dialogue and machine translation, only the final one for NLI).



The left side is repeated for every state in the encoder.

In the decoder, self-attention is limited to preceding words.

Figure 1: The Transformer - model architecture.

References I

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.