Transformers

Natural Language Processing

University of Maryland

Induction Head Example

Adapted from Michael Hahn

Vocabulary and Embeddings

Vocabulary

`!, a, b, c, d, e`

Vocabulary and Embeddings

Vocabulary

`!, a, b, c, d, e`

One-hot Embeddings

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Vocabulary and Embeddings

Vocabulary

`!, a, b, c, d, e`

Sequence + Positional Embeddings

! `[1 0 0 0 0 0 1 0 0 0 0 0]`

a `[0 1 0 0 0 0 0 1 0 0 0 0]`

b `[0 0 1 0 0 0 0 0 1 0 0 0]`

a `[0 1 0 0 0 0 0 0 0 1 0 0]`

c `[0 0 0 1 0 0 0 0 0 0 1 0]`

b `[0 0 1 0 0 0 0 0 0 0 0 1]`

Simplified Model

- One hot encoding of tokens
- Using previous layer input as values for key–value lookup
- One hot positional encoding (not sinusoidal)
- No feed forward after key–value

Attention and Layer Equations

Encoding of token i at layer l

$$Y_i^{(l)} = \sum_{j=1}^{i} \hat{a}_{i,j} \underbrace{\mathbf{V}_l}_{d\times d} Y_j^{(l-1)} + \underbrace{\mathbf{W}_l}_{d\times d} Y_i^{(l-1)} \tag{1}$$

Attention Logits (scalar)

$$a_{i,j}^{(l)} = \underbrace{Y_i^{(l-1)\top}}_{1\times d} \underbrace{A_l}_{d\times d} \underbrace{Y_j^{(l-1)}}_{d\times 1} \tag{2}$$

Masked Attention (if $i \geq j$, 0 otherwise)

$$\hat{a}_{i,j}^{(l)} = \frac{\exp\left(a_{i,j}^{(l)}\right)}{\sum_s \exp\left(a_{i,s}^{(l)}\right)} \tag{3}$$

Layer 1: Attend to Previous Position

Attention Matrix A_1:

$$A_1 = \begin{pmatrix} \mathbf{0} & & \mathbf{0} & & \\ & -100 & -100 & -100 & \cdots & -100 \\ & 100 & -100 & -100 & \cdots & -100 \\ \mathbf{0} & -100 & 100 & -100 & \cdots & -100 \\ & \vdots & \vdots & \vdots & \ddots & \vdots \\ & -100 & -100 & \cdots & 100 & -100 \end{pmatrix}$$

Value Matrix V_1:

$$V_1 = \begin{pmatrix} 0_{d/2 \times d/2} & 0_{d/2 \times d/2} \\ I_{d/2} & 0_{d/2 \times d/2} \end{pmatrix}$$

Passthrough Matrix W_1:

$$W_1 = \begin{pmatrix} I_{d/2} & 0_{d/2 \times d/2} \\ 0_{d/2 \times d/2} & 0_{d/2 \times d/2} \end{pmatrix}$$

Layer 2: Induction Head

Attention Matrix A_2:

$$A_2 = \begin{pmatrix} 0_{d/2 \times d/2} & 100 \cdot I_{d/2} \\ 0_{d/2 \times d/2} & 0_{d/2 \times d/2} \end{pmatrix}$$

Value Matrix V_2:

$$V_2 = \begin{pmatrix} 100 \cdot I_{d/2} & 0_{d/2 \times d/2} \\ 0_{d/2 \times d/2} & 0_{d/2 \times d/2} \end{pmatrix}$$

Passthrough Matrix W_2:

$$W_2 = 0_{d \times d}$$

Let's do a walkthrough...

- Matrices are getting big enough that I won't force you to do the math yourself
- I'll show you the outputs, you need to tell the story of conceptually of what they're doing

Layer 1: Attention Logits

$$a_{i,j}^{(1)} = \left(\underbrace{Y_i^{(1-1)}}_{1\times d} \right)^{\top} \underbrace{A_1}_{d\times d} \underbrace{Y_j^{(1-1)}}_{d\times 1} \tag{4}$$

Layer 1: Attention Logits

$$a_{i,j}^{(1)} = \left(\underbrace{Y_i^{(1-1)}}_{1\times d} \right)^{\top} \underbrace{A_1}_{d\times d} \underbrace{Y_j^{(1-1)}}_{d\times 1} \tag{4}$$

	!	a	b	a	c	b
!	-100.0	-100.0	-100.0	-100.0	-100.0	-100.0
a	100.0	-100.0	-100.0	-100.0	-100.0	-100.0
b	-100.0	100.0	-100.0	-100.0	-100.0	-100.0
a	-100.0	-100.0	100.0	-100.0	-100.0	-100.0
c	-100.0	-100.0	-100.0	100.0	-100.0	-100.0
b	-100.0	-100.0	-100.0	-100.0	100.0	-100.0

Layer 1: Masked Attention

Masked Attention (if $i \geq j$, 0 otherwise)

$$\hat{a}_{i,j}^{(1)} = \frac{\exp\left(a_{i,j}^{(1)}\right)}{\sum_s \exp\left(a_{i,s}^{(1)}\right)} \tag{5}$$

Masked + Normalized Attention Weights:

	!	a	b	a	c	b
!	1.0	0.0	0.0	0.0	0.0	0.0
a	1.0	0.0	0.0	0.0	0.0	0.0
b	0.0	1.0	0.0	0.0	0.0	0.0
a	0.0	0.0	1.0	0.0	0.0	0.0
c	0.0	0.0	0.0	1.0	0.0	0.0
b	0.0	0.0	0.0	0.0	1.0	0.0

Layer 1: Layer Output

$$Y_i^{(1)} = \sum_{j=1}^{i} \hat{a}_{i,j} \underbrace{\mathbf{V}_1}_{d \times d} Y_j^{(1-1)} + \underbrace{\mathbf{W}_1}_{d \times d} Y_i^{(1-1)} \tag{6}$$

Updated Representation $Y^{(1)}$:

	E_0	E_1	E_2	E_3	E_4	E_5	P_0	P_1	P_2	P_3	P_4	P_5
!	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
a	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
b	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
a	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
c	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
b	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0

What does Layer 1 Actually do?

- Attention looks at

What does Layer 1 Actually do?

- Attention looks at previous token

$$\hat{a}_{i,j}^{(1)} = \begin{cases} \approx 1 & \text{if } i = j+1 \text{or} i = j = 0 \\ \approx 0 & \text{otherwise} \end{cases} \tag{7}$$

- Output of layer captures embedding

$$\left(Y_i^{(1)}\right)^\top = \left[\underbrace{E[x_i][0], \ldots E[x_i][d/2]}_{\text{current token}}, \underbrace{E[x_i][0] \ldots E[x_i][d/2]}_{\text{previous token}}\right] \tag{8}$$

Layer 2: Attention Logits

$$a_{i,j}^{(2)} = \left(\underbrace{Y_i^{(2-1)}}_{1\times d} \right)^{\top} \underbrace{A_2}_{d\times d} \underbrace{Y_j^{(2-1)}}_{d\times 1} \tag{9}$$

Layer 2: Attention Logits

$$a_{i,j}^{(2)} = \left(\underbrace{Y_i^{(2-1)}}_{1\times d} \right)^{\top} \underbrace{A_2}_{d\times d} \underbrace{Y_j^{(2-1)}}_{d\times 1} \tag{9}$$

	!	a	b	a	c	b
!	0.0	100.0	0.0	0.0	0.0	0.0
a	0.0	0.0	100.0	0.0	100.0	0.0
b	0.0	0.0	0.0	100.0	0.0	0.0
a	0.0	0.0	100.0	0.0	100.0	0.0
c	0.0	0.0	0.0	0.0	0.0	100.0
b	0.0	0.0	0.0	100.0	0.0	0.0

Layer 2: Masked Attention

Masked Attention (if $i \geq j$, 0 otherwise)

$$\hat{a}_{i,j}^{(2)} = \frac{\exp\left(a_{i,j}^{(2)}\right)}{\sum_s \exp\left(a_{i,s}^{(2)}\right)} \tag{10}$$

Masked + Normalized Attention Weights:

	!	a	b	a	c	b
!	1.0	0.0	0.0	0.0	0.0	0.0
a	0.0	0.0	0.0	0.0	0.0	0.0
b	0.0	0.0	0.0	0.0	0.0	0.0
a	0.0	0.0	1.0	0.0	0.0	0.0
c	0.0	0.0	0.0	0.0	0.0	0.0
b	0.0	0.0	0.0	1.0	0.0	0.0

Layer 2: Layer Output

$$Y_i^{(2)} = \sum_{j=1}^{i} \hat{a}_{i,j} \underbrace{\mathbf{V}_2}_{d \times d} Y_j^{(2-1)} + \underbrace{\mathbf{W}_2}_{d \times d} Y_i^{(2-1)} \tag{11}$$

Updated Representation $Y^{(2)}$:

	E_0	E_1	E_2	E_3	E_4	E_5	P_0	P_1	P_2	P_3	P_4	P_5
!	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
a	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
b	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
a	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
c	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
b	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

What does Layer 2 Actually do?

- Unmasked attention $a_{i,j}^{(2)}$ checks

What does Layer 2 Actually do?

- Unmasked attention $a_{i,j}^{(2)}$ checks if token i is the same as $j-1$ token
- Before it gets zeroed out, by V_2, $\mathbf{Y}_i^{(2)}$

What does Layer 2 Actually do?

- Unmasked attention $a_{i,j}^{(2)}$ checks if token i is the same as $j-1$ token
- Before it gets zeroed out, by V_2, $\mathbf{Y}_i^{(2)}$

$$= \begin{cases} \mathbf{V}_2 \left[\underbrace{E[x_{i+1}][0], \ldots E[x_{i+1}][d/2]}_{\text{current token}}, \underbrace{E[x_i][0] \ldots E[x_i][d/2]}_{\text{previous token}} \right]^\top & x_i \text{ seen} \\ \vec{0} & \text{otherwise} \end{cases} \tag{12}$$

Big picture

- Shows mechanics of how attention can directly copy previous tokens in history
- More interesting: copying for similar words, copying for similar contexts
- Even more interesting: copying with patterns (e.g., skipping n tokens to make ends of poetry rhyme)