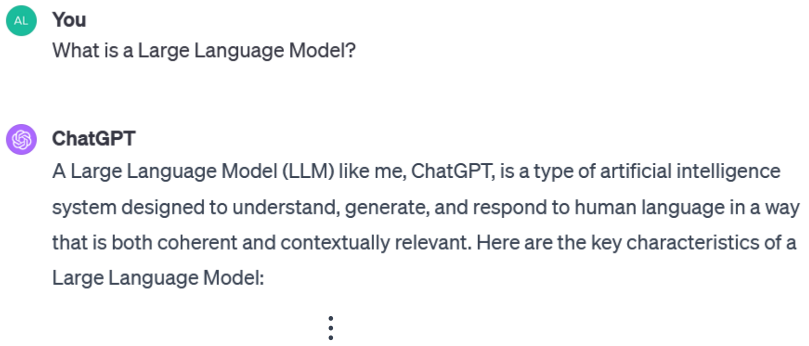
Heinrich Heine Universität Düsseldorf

Transformer Network Encoders

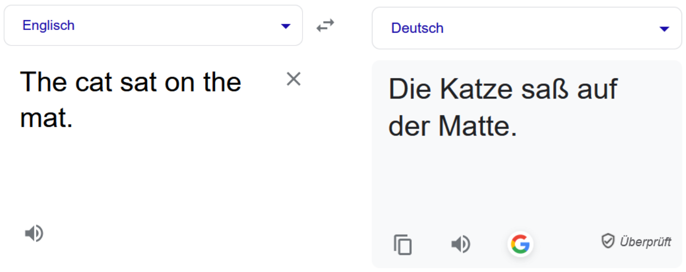
An overview

What are Transformer Networks?

- Transformer Networks are a type of neural network architecture designed to handle sequential data
- Examples:
  - Chatbots

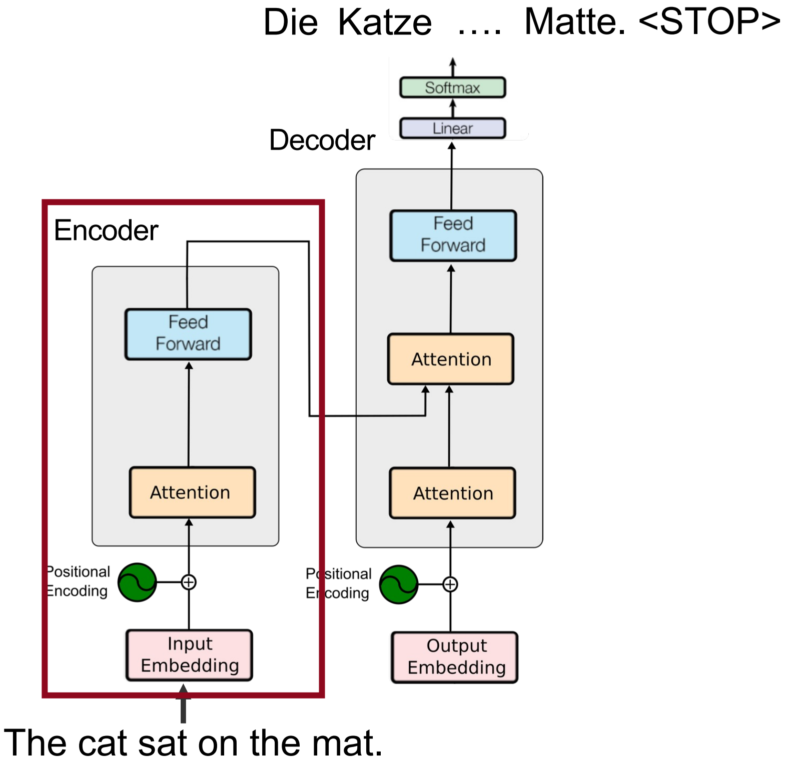


  - Translator



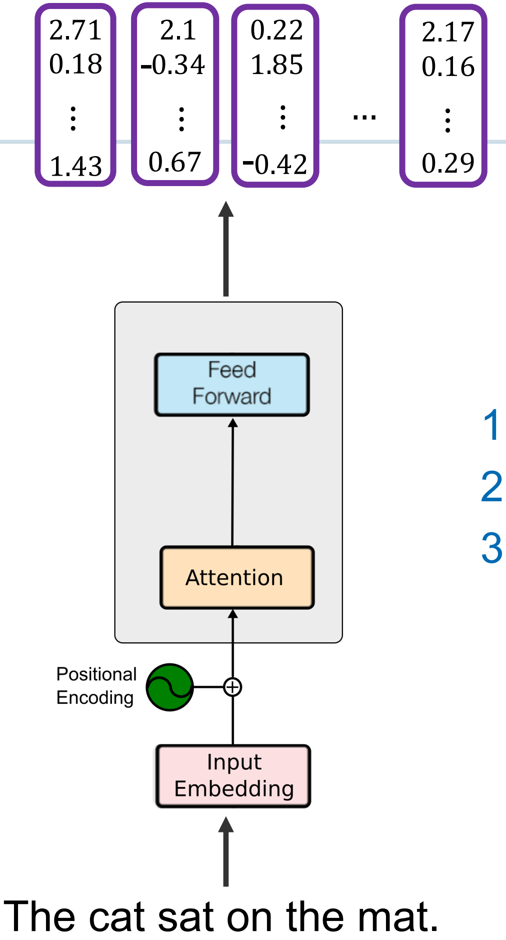
Transformer Network Architecture

- Transformer Network
  - Encoder: Encodes the input sequence into numerical vectors
  - Decoder: Uses the numerically encoded input sequence as its input and produces an output sequence.



Modified figure from: Vaswani, A., et al. (2017). Attention is all you need. *Adv. Neural Inf. Process. Syst.*, *30*.

Encoder



1. Input Embedding & Positional Encoding
2. Attention
3. Feed Forward Neural Network

Modified figure from: Vaswani, A., et al. (2017). Attention is all you need. *Adv. Neural Inf. Process. Syst.*, *30*.

Input Embedding

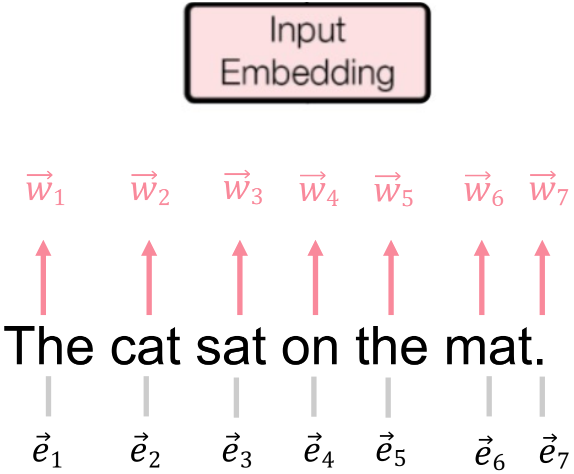
- For every word, we have a specific vector representing this word

„The“ →  → $\begin{pmatrix} 0.17 \\ 0.32 \\ \vdots \\ 0.25 \end{pmatrix} = \vec{w}_1 \in \mathbb{R}^d$

„cat“ →  → $\begin{pmatrix} 0.40 \\ -0.21 \\ \vdots \\ 0.53 \end{pmatrix} = \vec{w}_2$

⋮ ⋮ ⋮

„.“ →  → $\begin{pmatrix} -1.22 \\ 0.76 \\ \vdots \\ -2.19 \end{pmatrix} = \vec{w}_7$



Positional Encoding

Positional Encoding: $\vec{p}_1 = \begin{bmatrix} 0.00 \\ 0.84 \\ \vdots \\ 0.14 \end{bmatrix}$ $\vec{p}_2 = \begin{bmatrix} 1.00 \\ 0.54 \\ \vdots \\ -0.99 \end{bmatrix}$ … $\vec{p}_7 = \begin{bmatrix} 0.00 \\ 1.00 \\ \vdots \\ 0.96 \end{bmatrix} \in \mathbb{R}^d$

- The same positional encoding is used for every input sequence!

Positional Encoding 

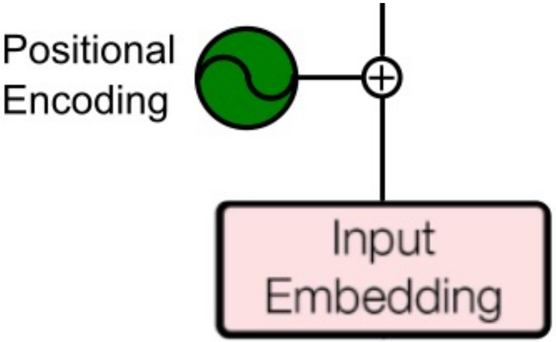
$\vec{p}_1$ $\vec{p}_2$ $\vec{p}_3$ $\vec{p}_4$ $\vec{p}_5$ $\vec{p}_6$ $\vec{p}_7$

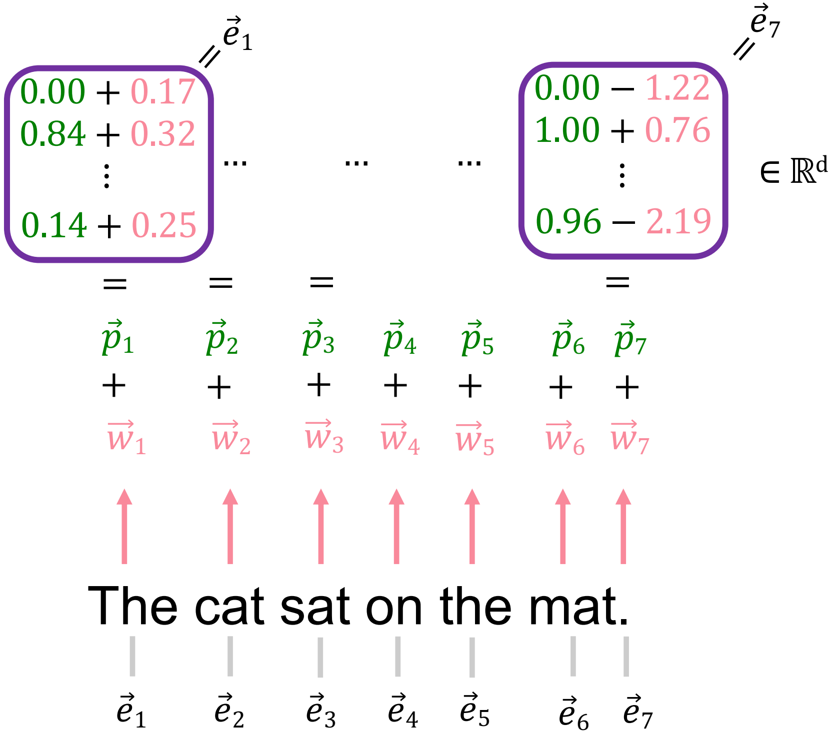
$\uparrow$ $\uparrow$ $\uparrow$ $\uparrow$ $\uparrow$ $\uparrow$ $\uparrow$

The cat sat on the mat.

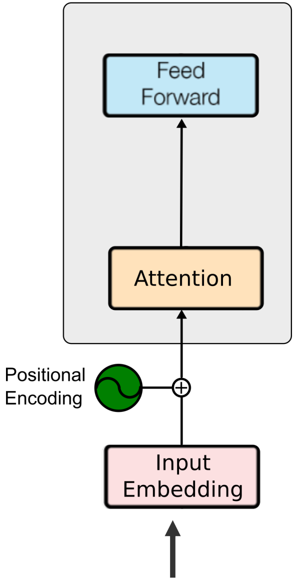
$\vec{e}_1$ $\vec{e}_2$ $\vec{e}_3$ $\vec{e}_4$ $\vec{e}_5$ $\vec{e}_6$ $\vec{e}_7$

Input Embedding & Positional Encoding





Encoder



The cat sat on the mat.

1. Input Embedding & Positional Encoding
2. Attention
3. Feed Forward Neural Network

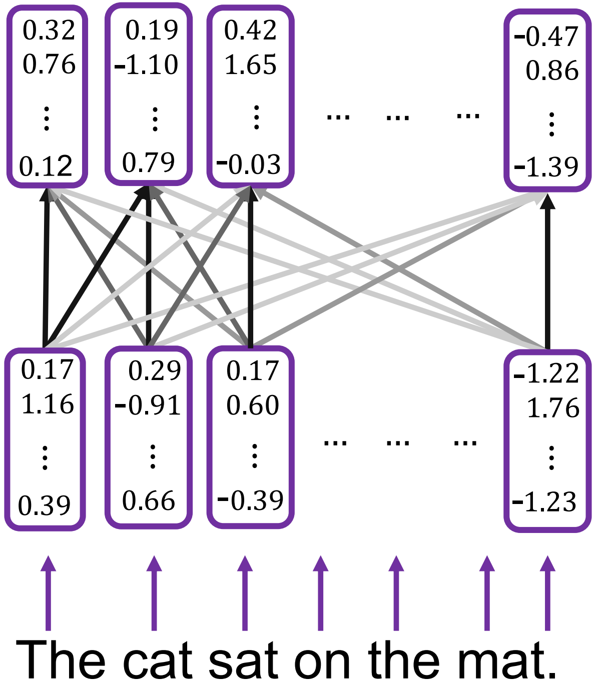
Attention

$$Attention(\vec{e}_i, K, V) = softmax\left(\frac{\vec{e}_i^{\,T} \cdot K}{\sqrt{d}}\right) \cdot V^T$$

$$K = (\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_7), \qquad V^T = \begin{pmatrix} \vec{e}_1 \\ \vec{e}_2 \\ \vdots \\ \vec{e}_7 \end{pmatrix}$$

<u>Example for $i = 1$:</u>

$$\vec{e}_1 = 0.45 \cdot \vec{e}_1 + 0.25 \cdot \vec{e}_2 + 0.14 \cdot \vec{e}_3 + 0.05 \cdot \vec{e}_4 + 0.03 \cdot \vec{e}_5 + 0.06 \cdot \vec{e}_6 + 0.02 \cdot \vec{e}_7$$



Attention

Heinrich Heine
Universität Düsseldorf

$$\boxed{Attention(\vec{e}_i, K, V) = softmax\left(\frac{\vec{e}_i^{\,T}\cdot K}{\sqrt{d}}\right)\cdot V^T}$$

$$K = (\vec{e}_1, \vec{e}_2, \dots, \vec{e}_7), \qquad V^T = \begin{pmatrix} \vec{e}_1^{\,T} \\ \vec{e}_2^{\,T} \\ \vdots \\ \vec{e}_7^{\,T} \end{pmatrix}$$

<u>Example for $i = 2$:</u>

$Attention(\vec{e}_2, K, V)$

1. $\vec{e}_i^{\,T}\cdot K = \vec{e}_2^{\,T}\cdot K =$

$$= (\vec{e}_2^{\,T}\cdot\vec{e}_1, \vec{e}_2^{\,T}\cdot\vec{e}_2, \dots, \vec{e}_2^{\,T}\cdot\vec{e}_7)$$

$$= (23.2, 70.8, 33.7, 5.7, -12.4, 27.8, -22.4)$$

2. $\frac{\vec{e}_i^{\,T}\cdot K}{\sqrt{d}} = (0.8, 2.6, 1.2, 0.2, -0.4, 1.0, -0.8)$

3. $softmax\left(\frac{\vec{e}_i^{\,T}\cdot K}{\sqrt{d}}\right) = (0.1, 0.55, 0.14, 0.05, 0.03, 0.12, 0.02)$

4. $softmax\left(\frac{\vec{e}_i^{\,T}\cdot K}{\sqrt{d}}\right)\cdot V^T = (0.1\cdot\vec{e}_1 + 0.55\cdot\vec{e}_2 + 0.14\cdot\vec{e}_3 + 0.05\cdot\vec{e}_4 + 0.03\cdot\vec{e}_5 + 0.12\cdot\vec{e}_6 + 0.02\cdot\vec{e}_7)^T$

$$\begin{array}{ccccccc} \vec{e}_1 & \vec{e}_2 & \vec{e}_3 & & & & \vec{e}_7 \\ \boxed{\begin{matrix} 0.7 \\ 0.4 \\ \vdots \\ 2.1 \end{matrix}} & \boxed{\begin{matrix} 0.2 \\ -0.9 \\ \vdots \\ 0.6 \end{matrix}} & \boxed{\begin{matrix} 0.1 \\ 0.6 \\ \vdots \\ -0.3 \end{matrix}} & \dots & \dots & \dots & \boxed{\begin{matrix} -0.1 \\ 0.8 \\ \vdots \\ -1.6 \end{matrix}} \\ \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \end{array}$$

The cat sat on the mat.

Attention (2)

$$Attention(\vec{e}_i, K, V) = softmax(\frac{\vec{e}_i^{\,T} \cdot K}{\sqrt{d}}) \cdot V^T$$

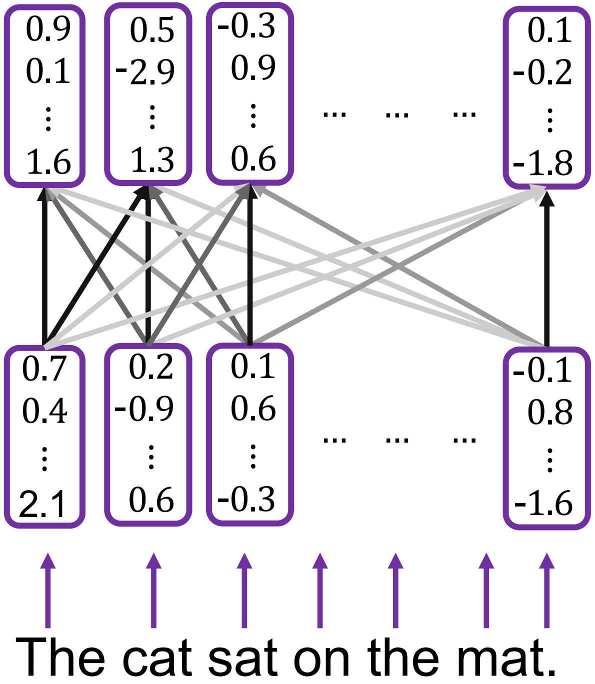
$$Attention(W_E \cdot \vec{e}_i, W_K \cdot K, W_V \cdot V)$$

$$= Attention(\vec{\tilde{e}}_i, \tilde{K}, \tilde{V})$$

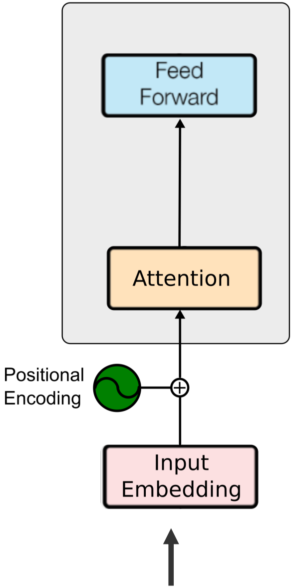
$$\tilde{K} = W_K \cdot K = (W_K \cdot \vec{e}_1, W_K \cdot \vec{e}_2, \ldots, W_K \cdot \vec{e}_7),$$

$$\tilde{V} = WV \cdot V = (W_V \cdot \vec{e}_1,\ WV \cdot \vec{e}_2, \ldots, W_V \cdot \vec{e}_7)$$

$$softmax(\vec{x})_j = \frac{e^{x_j}}{\sum_{k=1}^{n} e^{x_k}}, \quad \vec{x} \in \mathbb{R}^N$$



Encoder

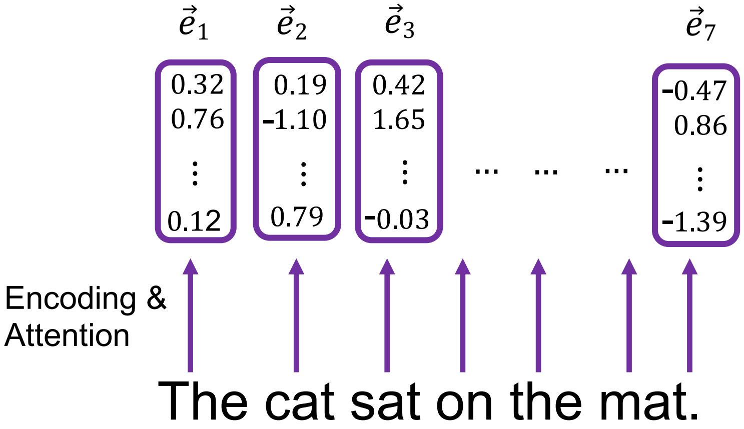


The cat sat on the mat.

1. Input Embedding & Positional Encoding
2. Attention
3. Feed Forward Neural Network

Feed Forward Neural Network

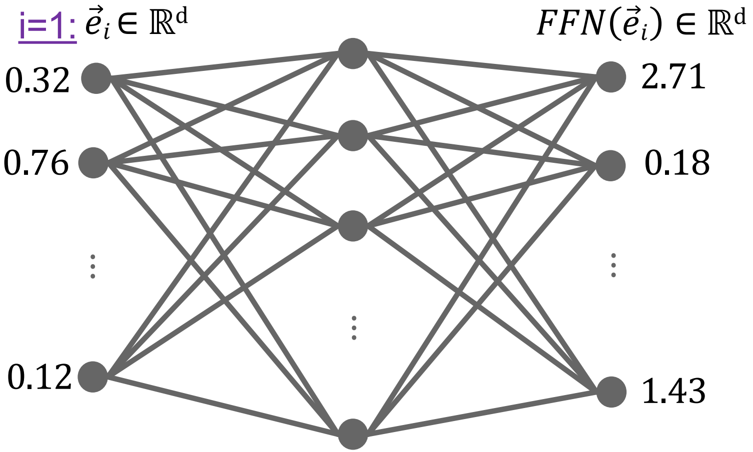
- Mapping through Encoding and Attention:



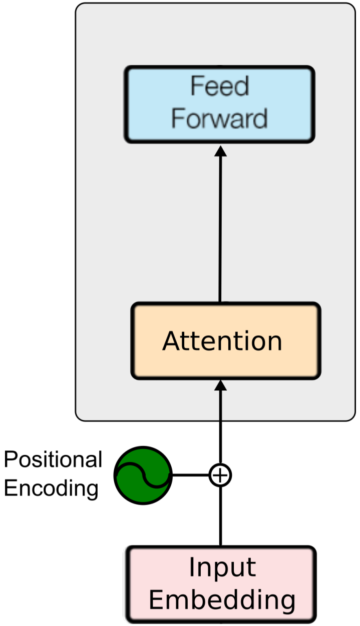
- Feed Forward Neural Network:

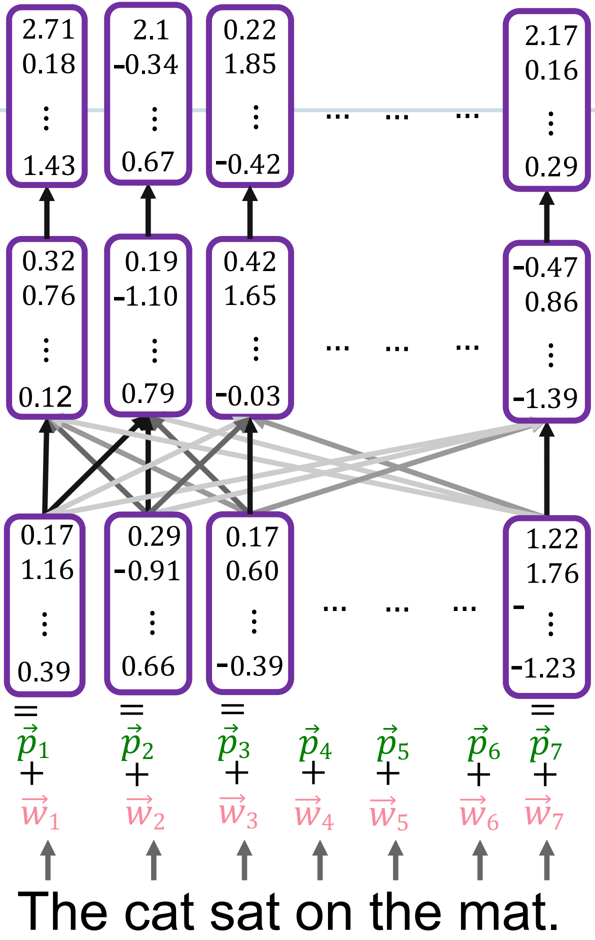
Feed Forward

$$FFN(\vec{e}_i) = W_2 \cdot \mathrm{ReLU}(W_1 \cdot \vec{e}_i + \vec{b_1}) + \vec{b_2}$$



Encoder


Feed
Forward
Attention
Positional
Encoding
Input
Embedding


2.71
0.18
1.43
2.1
-0.34
0.67
0.22
1.85
-0.42
2.17
0.16
0.29
0.32
0.76
0.12
0.19
-1.10
0.79
0.42
1.65
-0.03
-0.47
0.86
-1.39
0.17
1.16
0.39
0.29
-0.91
0.66
0.17
0.60
-0.39
1.22
1.76
-1.23
$\vec{p}_1 + \vec{w}_1$
$\vec{p}_2 + \vec{w}_2$
$\vec{p}_3 + \vec{w}_3$
$\vec{p}_4 + \vec{w}_4$
$\vec{p}_5 + \vec{w}_5$
$\vec{p}_6 + \vec{w}_6$
$\vec{p}_7 + \vec{w}_7$
The cat sat on the mat.

Applying Encoders to Protein Sequences