Transformers I

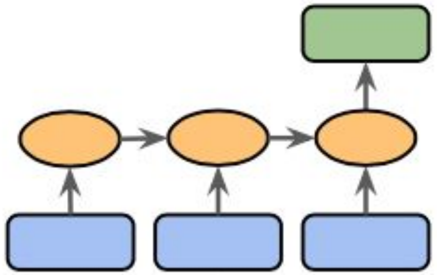
Dr. Parlett-Pelleriti

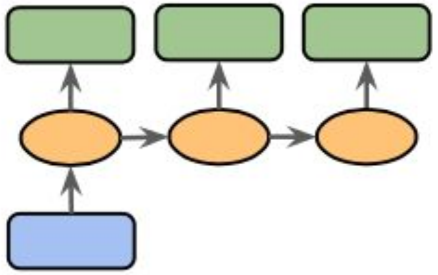
Transformers

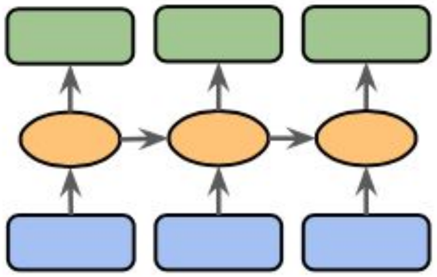
Outline

- Encoder/Decoder Structure
- Transformer Intro
- Word Embeddings
- Positional Encoding

Sequence Models


many-to-one


one-to-many


many-to-many

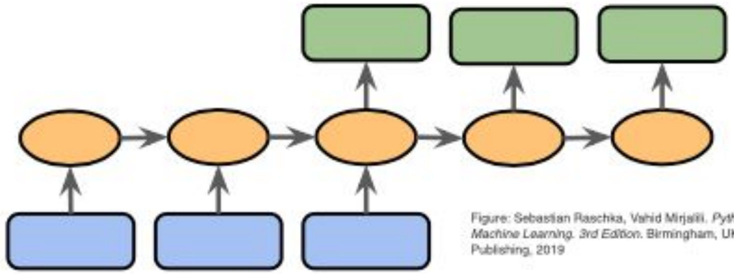

many-to-many

Figure: Sebastian Raschka, Vahid Mirjalili. *Python Machine Learning. 3rd Edition*. Birmingham, UK: Packt Publishing, 2019

Seq-to-Seq Models

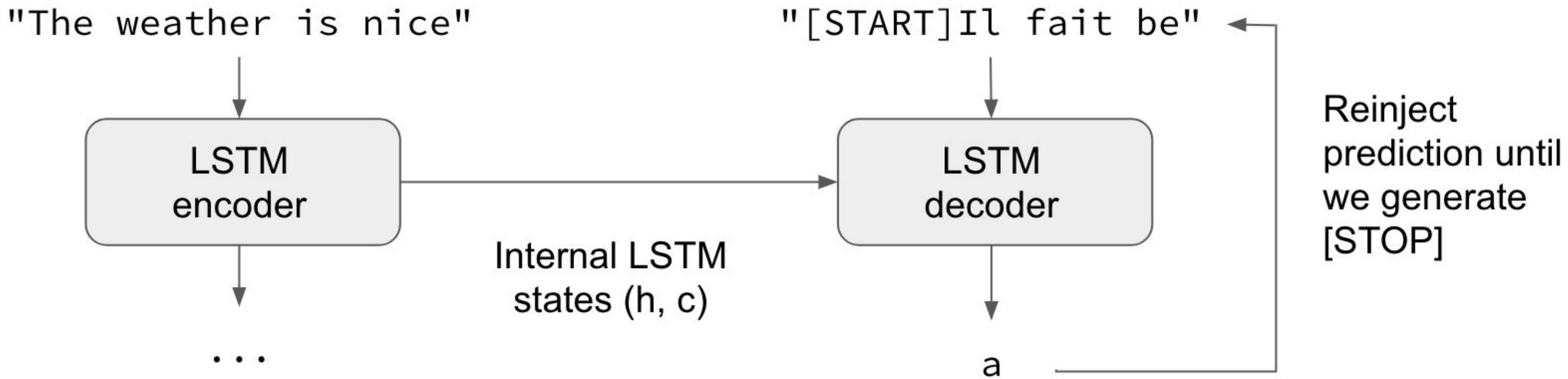
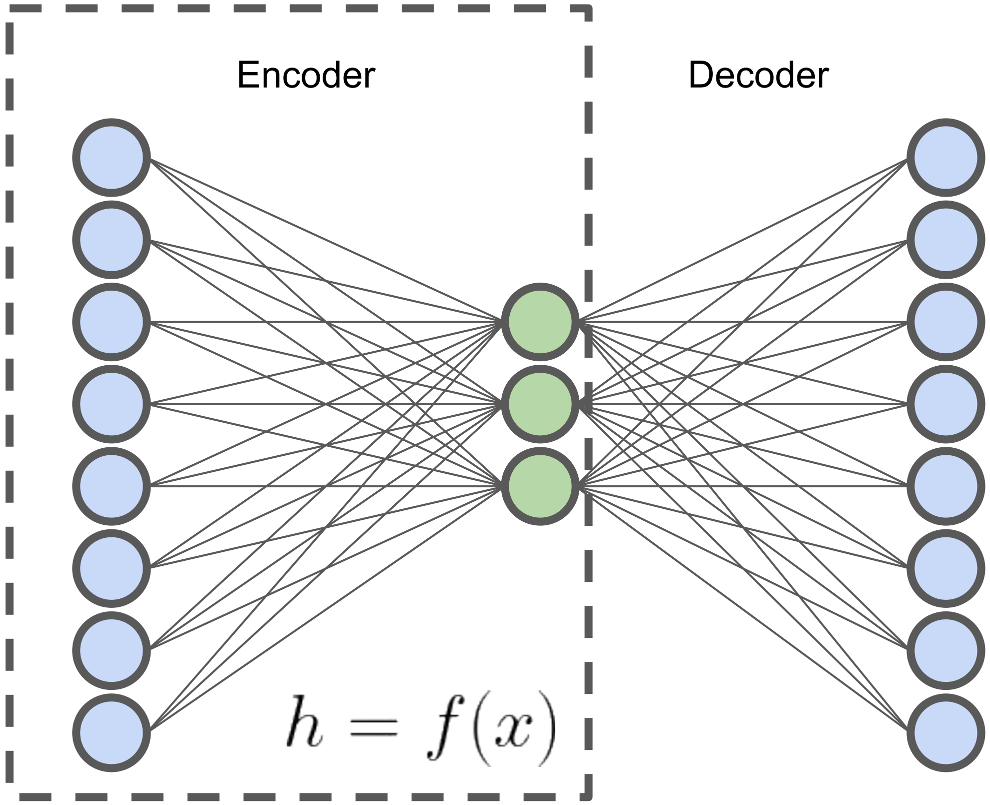
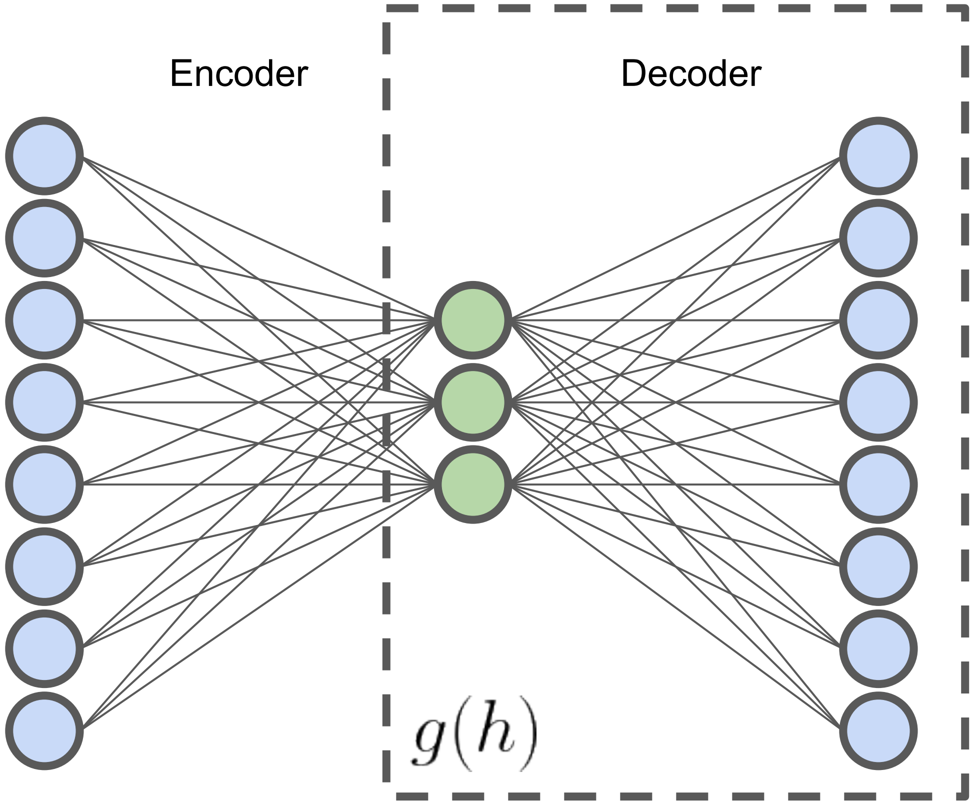

"The weather is nice"
LSTM encoder
...
Internal LSTM states (h, c)
"[START]Il fait be"
LSTM decoder
a
Reinject prediction until we generate [STOP]

Image from: https://blog.keras.io/a-ten-minute-introduction-to-sequence-to-sequence-learning-in-keras.html

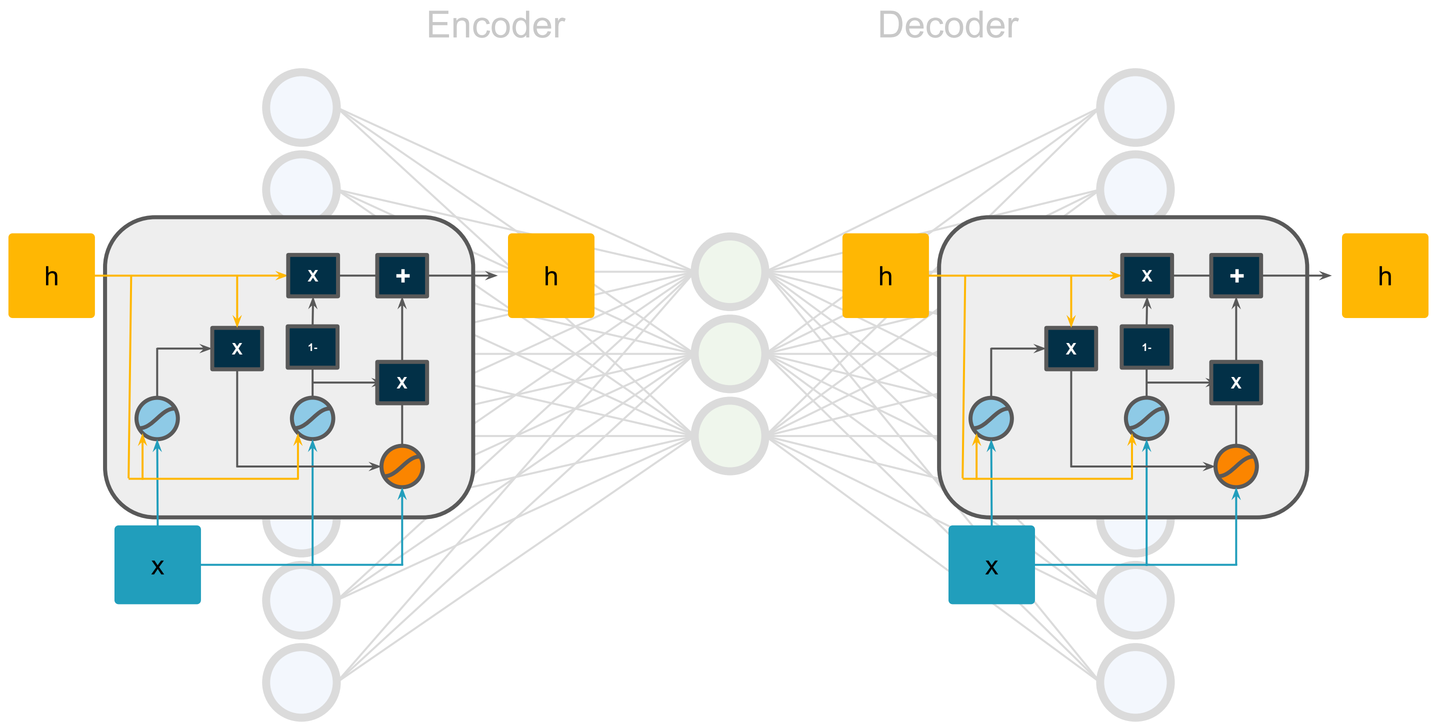
Autoencoder Architecture



Autoencoder Architecture

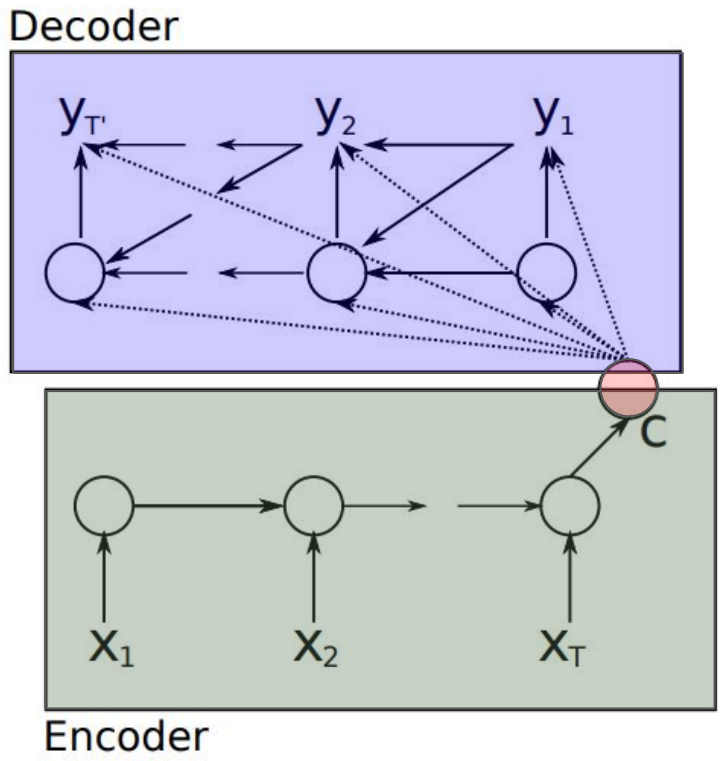


Autoencoder Architecture



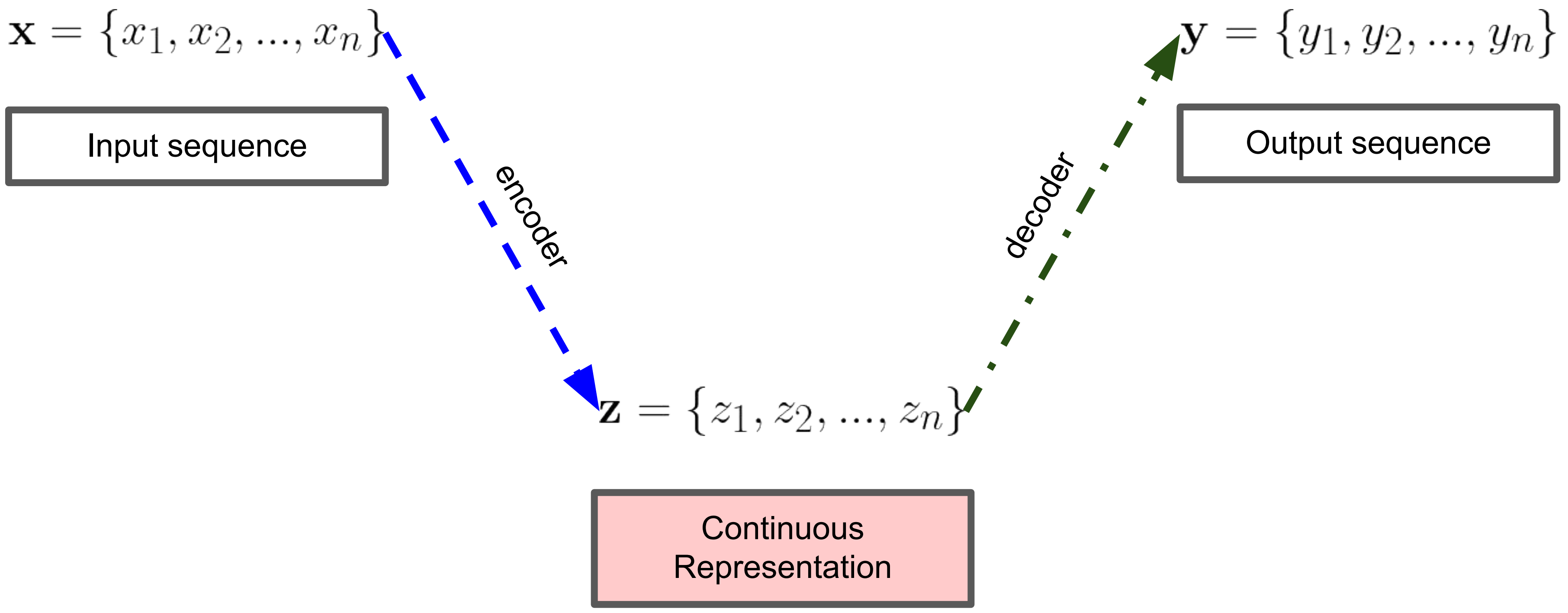
Cho et. al
2014

GRU Machine Translation



Cho et. al
2014

General Encoder/Decoder Model



$\mathbf{x} = \{x_1, x_2, ..., x_n\}$
Input sequence
encoder
$\mathbf{z} = \{z_1, z_2, ..., z_n\}$
Continuous Representation
decoder
$\mathbf{y} = \{y_1, y_2, ..., y_n\}$
Output sequence

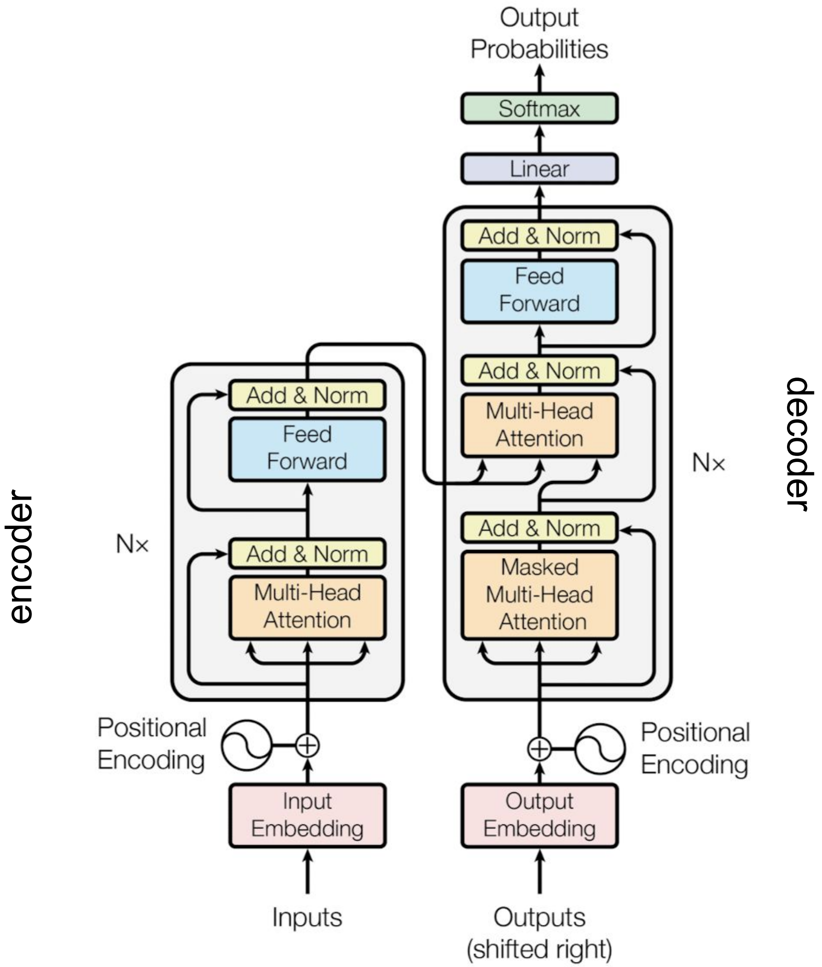


Figure 1: The Transformer - model architecture.

Image from: Attention is all you Need (Vaswani et. al 2017)

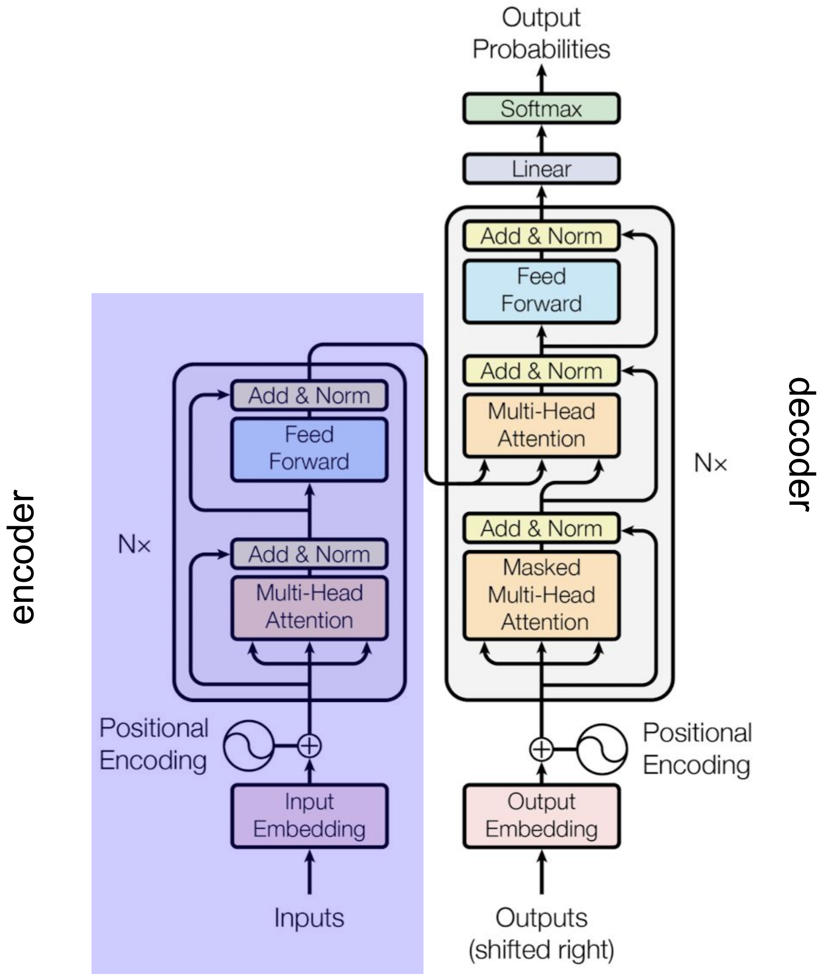


Figure 1: The Transformer - model architecture.

Image from: Attention is all you Need (Vaswani et. al 2017)

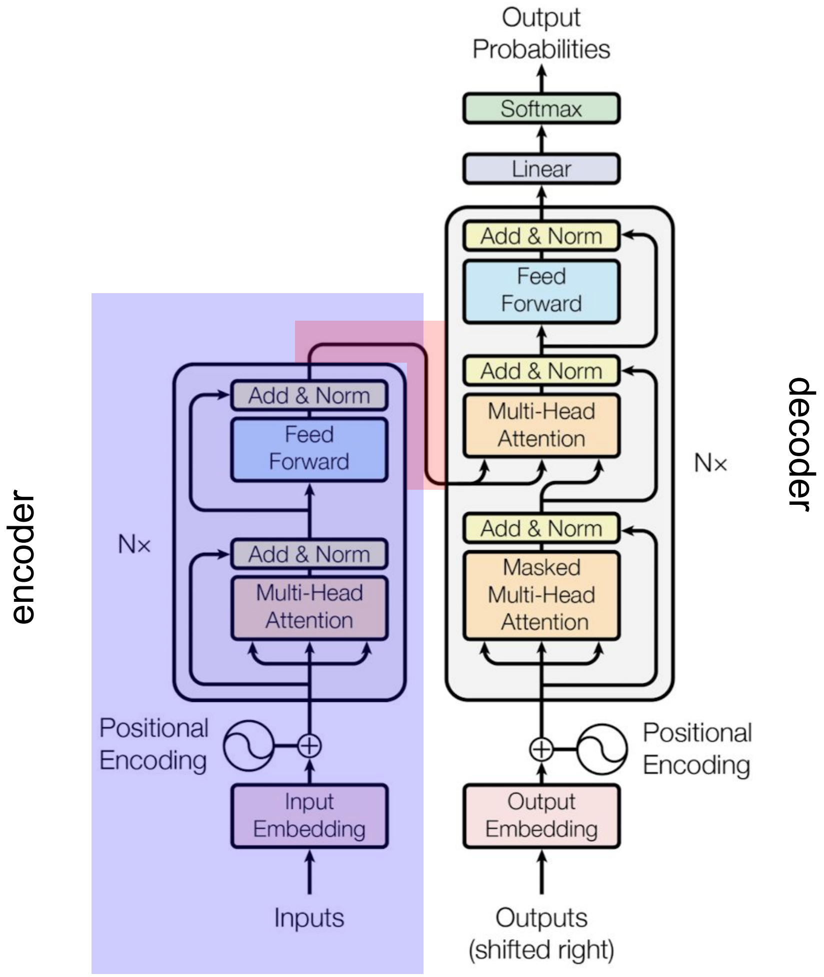


Figure 1: The Transformer - model architecture.

Image from: Attention is all you Need (Vaswani et. al 2017)

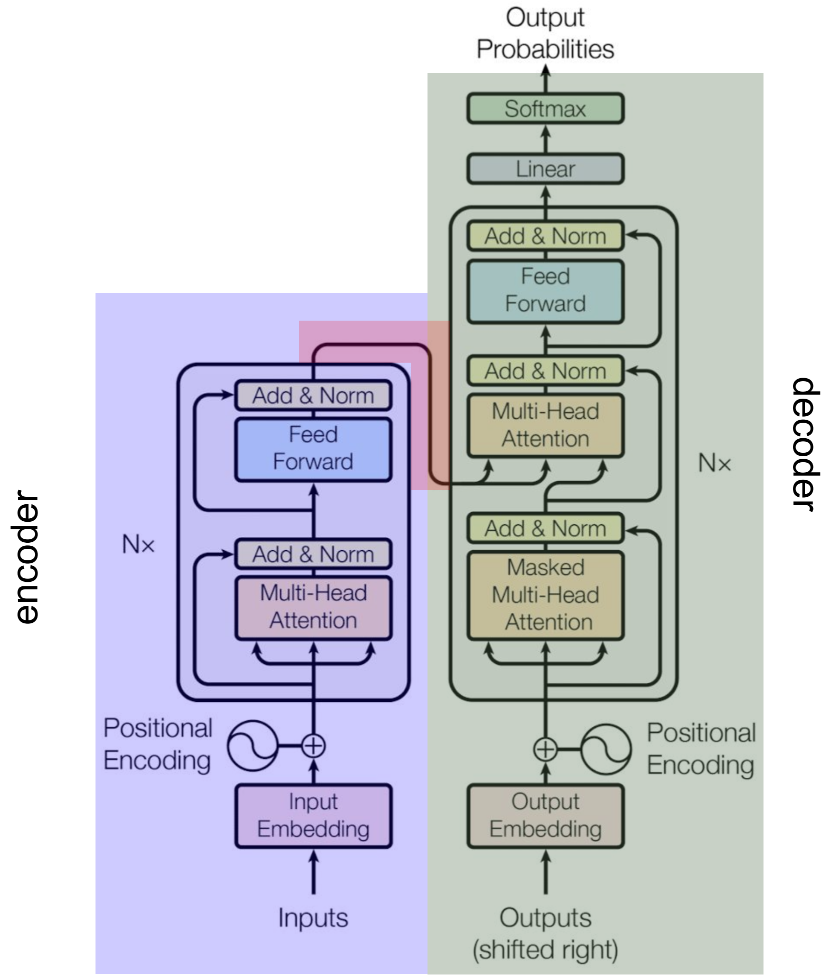


Figure 1: The Transformer - model architecture.

Image from: Attention is all you Need (Vaswani et. al 2017)

Attention Is All You Need

Ashish Vaswani[*]
Google Brain
avaswani@google.com

Noam Shazeer[*]
Google Brain
noam@google.com

Niki Parmar[*]
Google Research
nikip@google.com

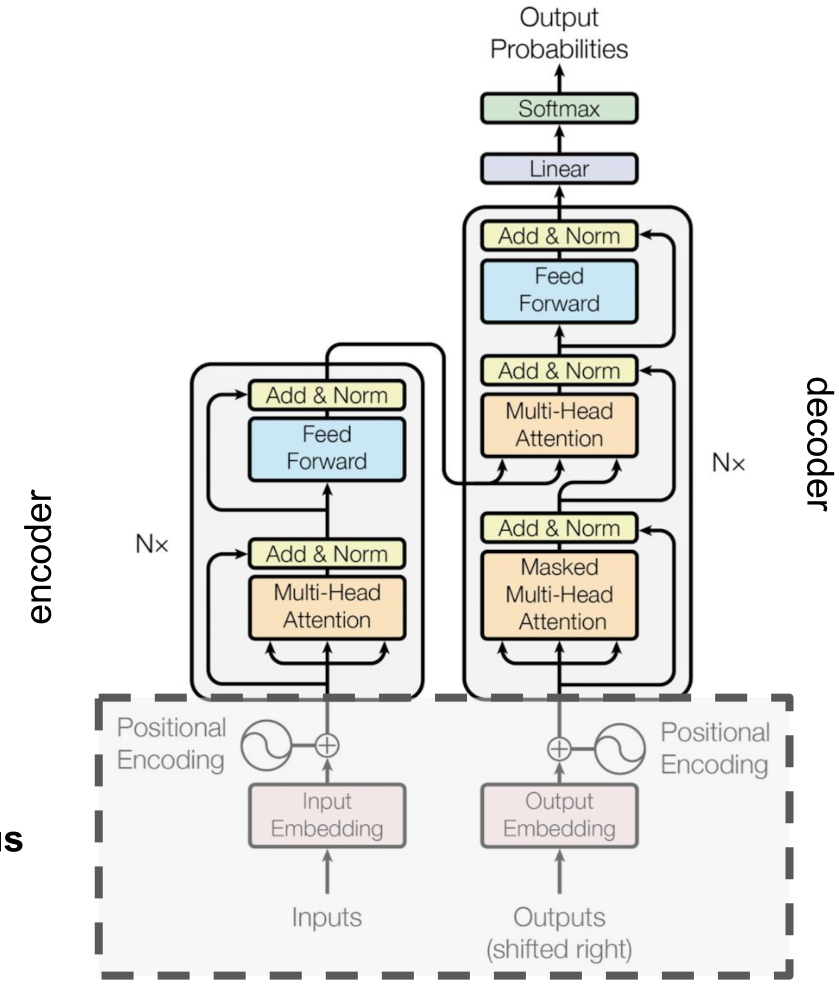
Jakob Uszkoreit[*]
Google Research
usz@google.com

Llion Jones[*]
Google Research
llion@google.com

Aidan N. Gomez[*] [†]
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser[*]
Google Brain
lukaszkaiser@google.com

Illia Polosukhin[*] [‡]
illia.polosukhin@gmail.com



Today's Focus

Figure 1: The Transformer - model architecture.

Image from: Attention is all you Need (Vaswani et. al 2017)

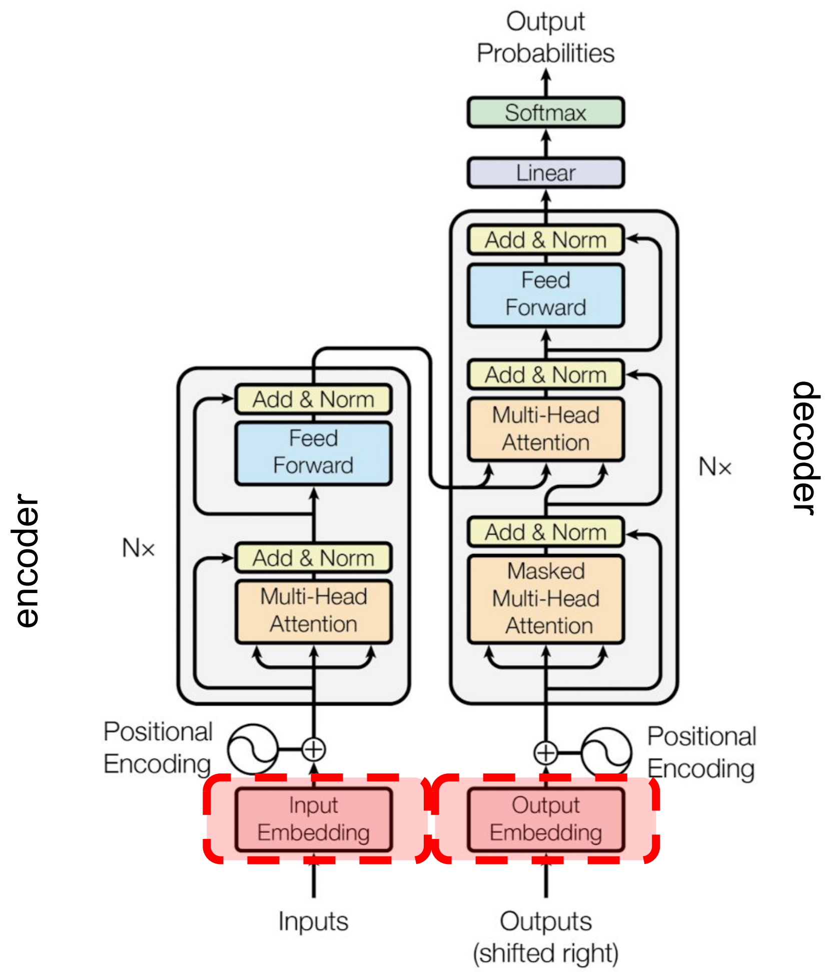


Figure 1: The Transformer - model architecture.

Image from: Attention is all you Need (Vaswani et. al 2017)

Word Embeddings

$$\begin{bmatrix} 0.1 \\ 0.2 \\ 0.4 \\ 0.2 \\ 0.6 \\ 0.1 \end{bmatrix}$$

gorgeous

$$\begin{bmatrix} 0.2 \\ 0.4 \\ 0.5 \\ 0.9 \\ 0.3 \\ 0.1 \end{bmatrix}$$

python

Word Embeddings

$$d_{model} \left\{ \underset{\text{gorgeous}}{\begin{bmatrix} 0.1 \\ 0.2 \\ 0.4 \\ 0.2 \\ 0.6 \\ 0.1 \end{bmatrix}} \right. \qquad d_{model} \left\{ \underset{\text{python}}{\begin{bmatrix} 0.2 \\ 0.4 \\ 0.5 \\ 0.9 \\ 0.3 \\ 0.1 \end{bmatrix}} \right.$$

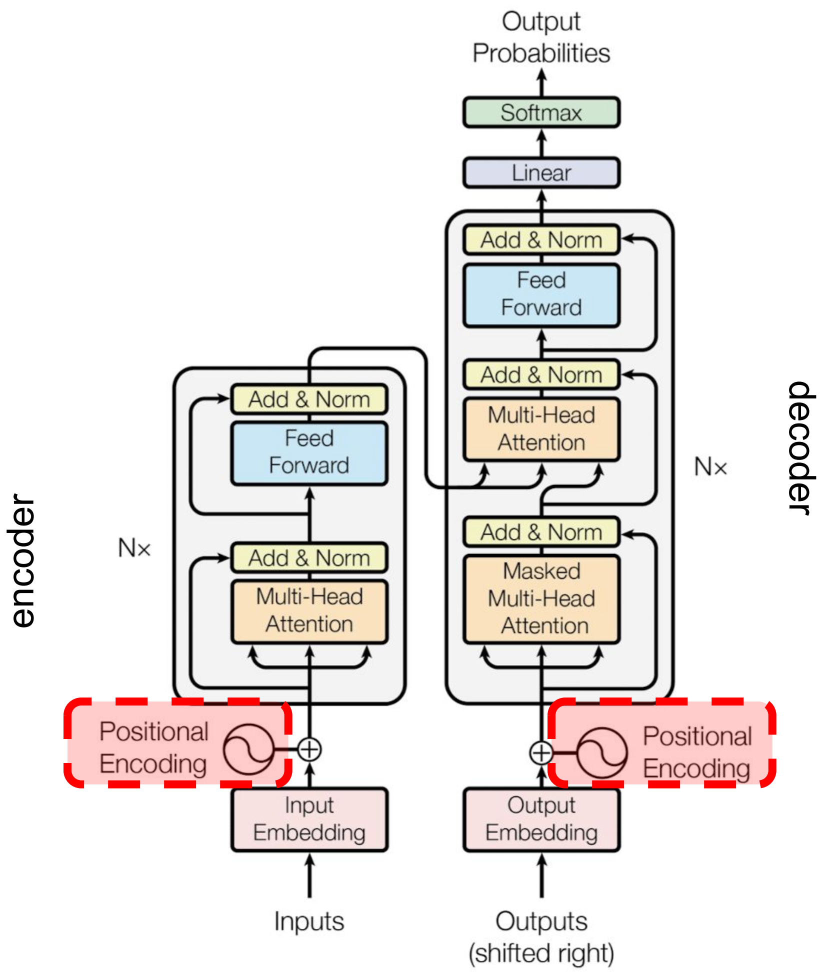
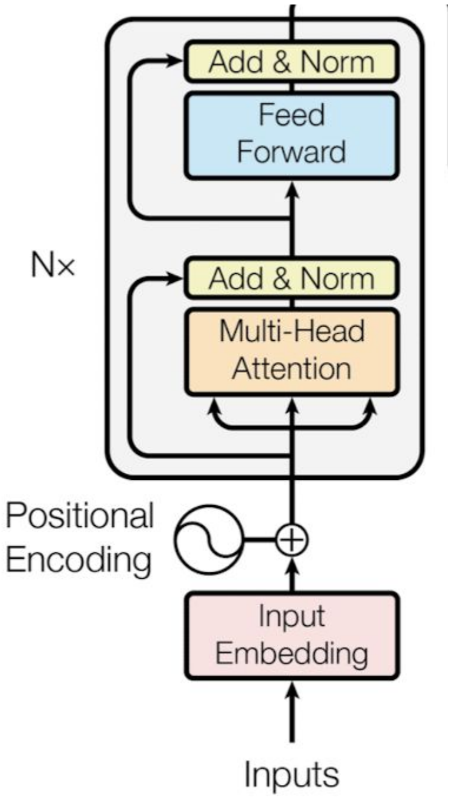


Figure 1: The Transformer - model architecture.

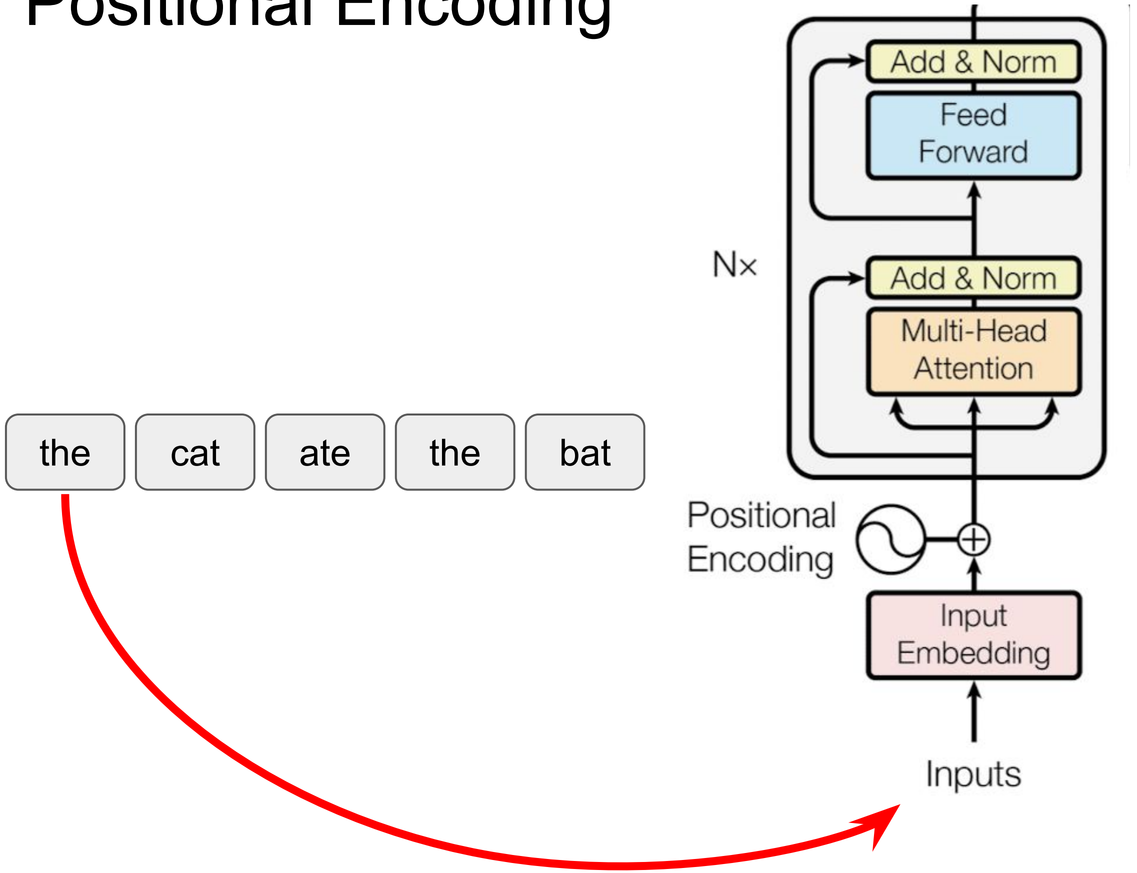
Image from: Attention is all you Need (Vaswani et. al 2017)

Positional Encoding

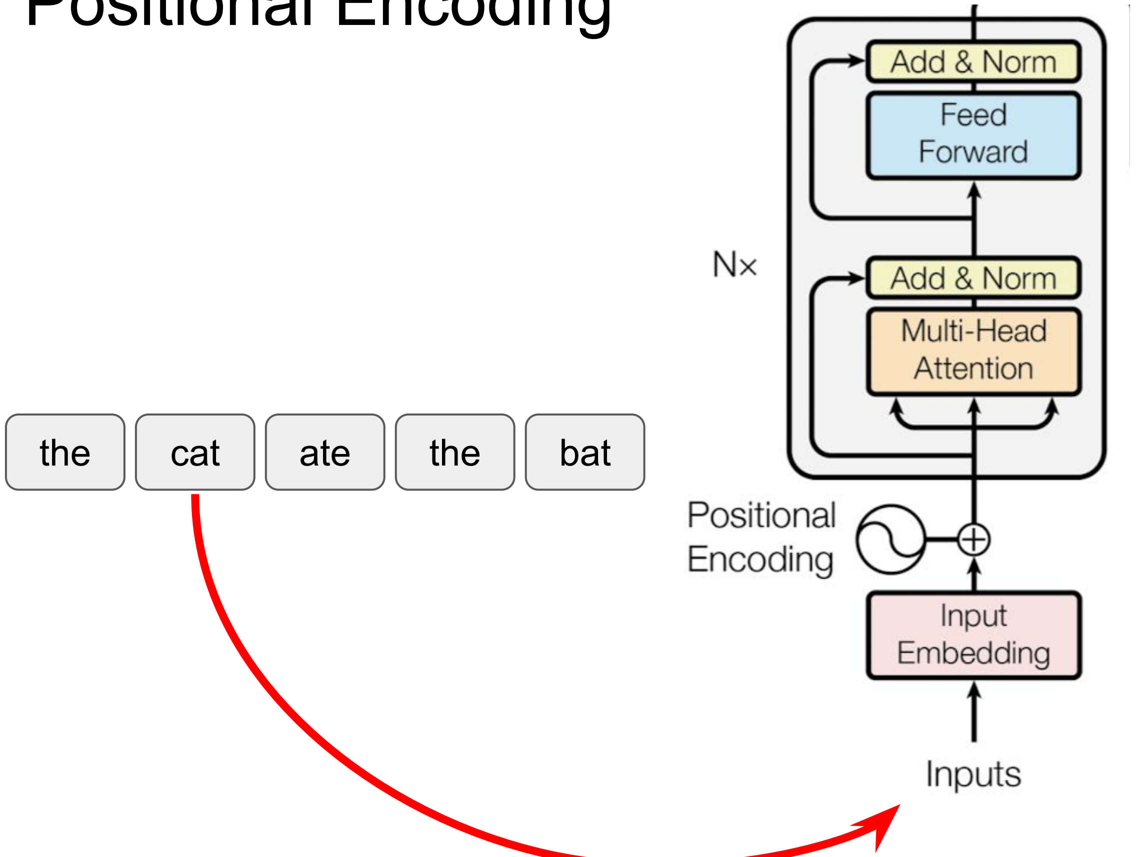
the	cat	ate	the	bat



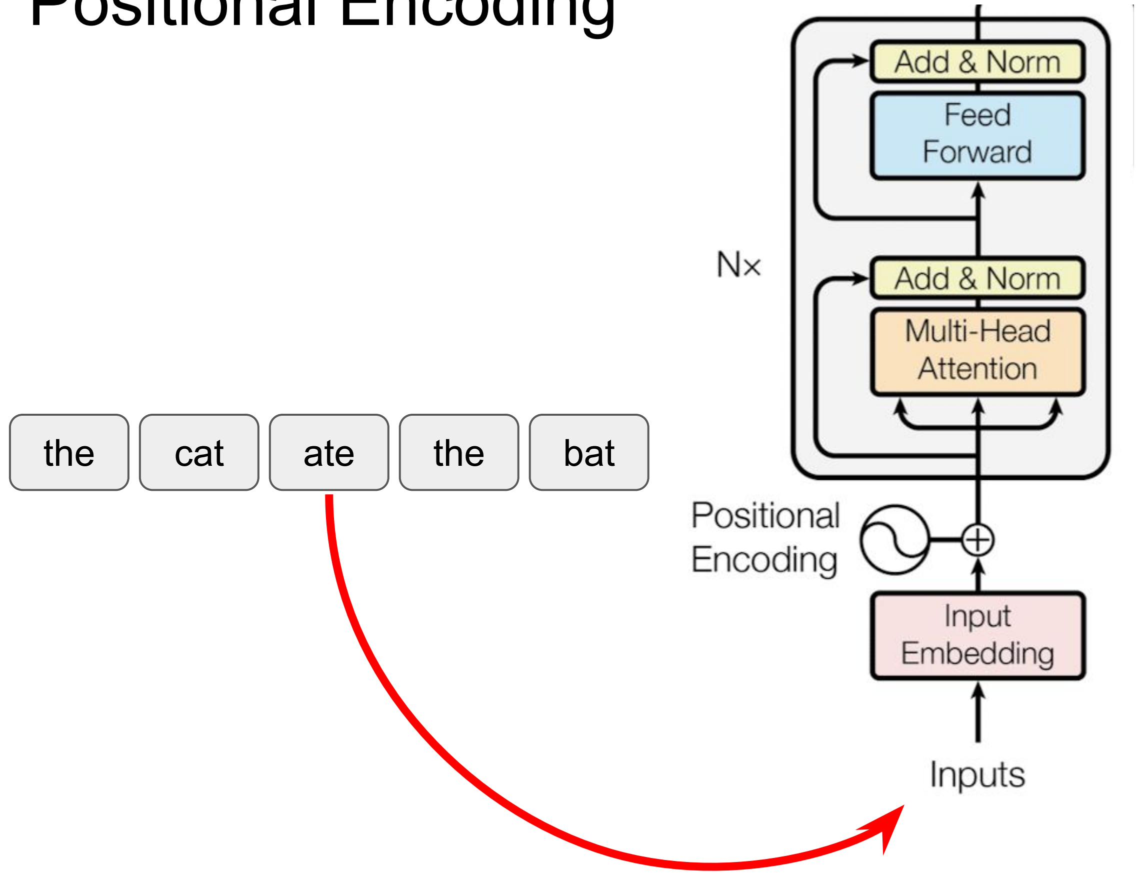
Positional Encoding



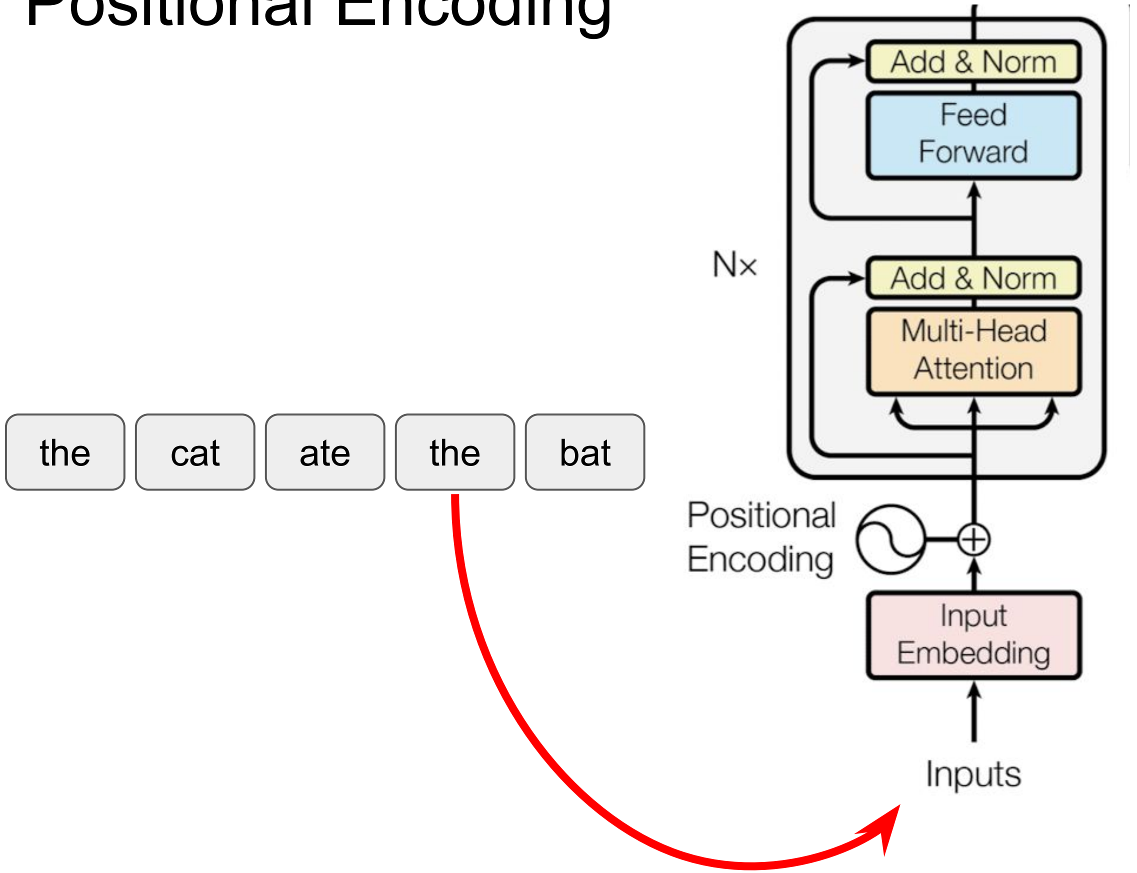
Positional Encoding



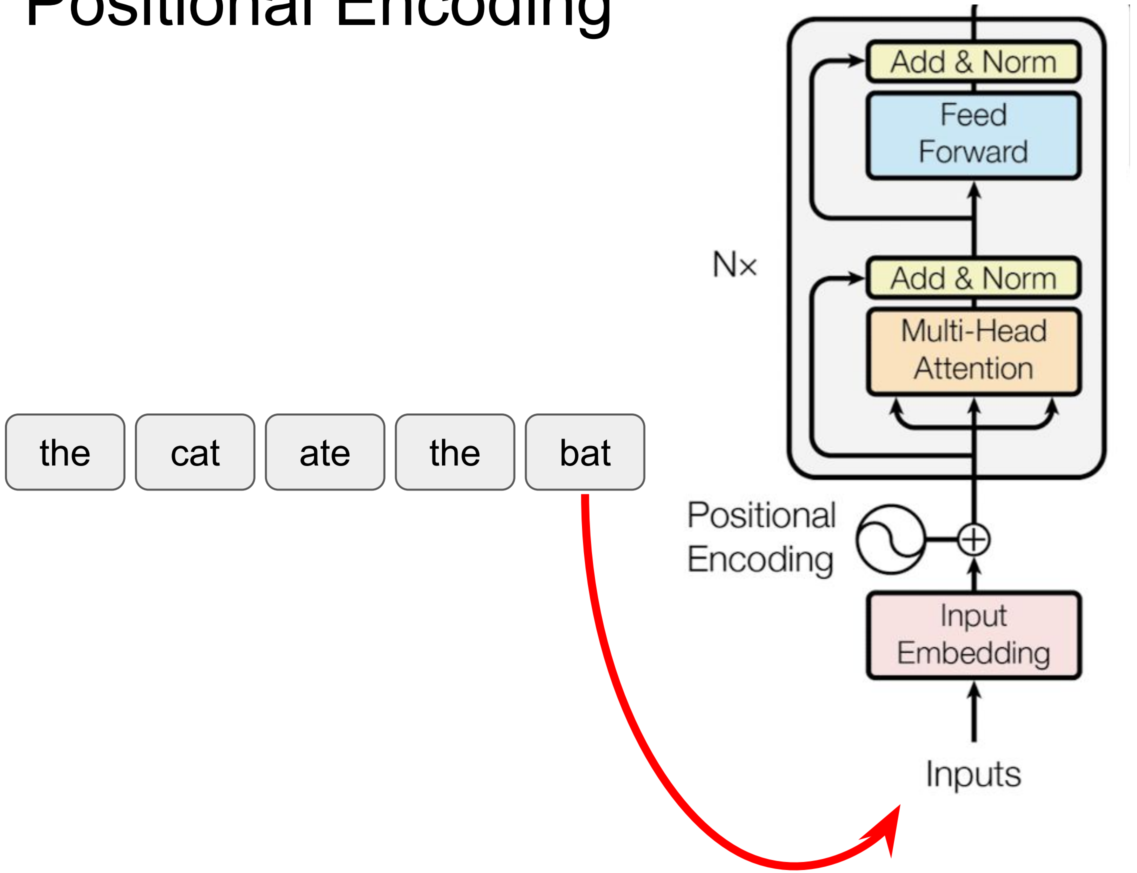
Positional Encoding



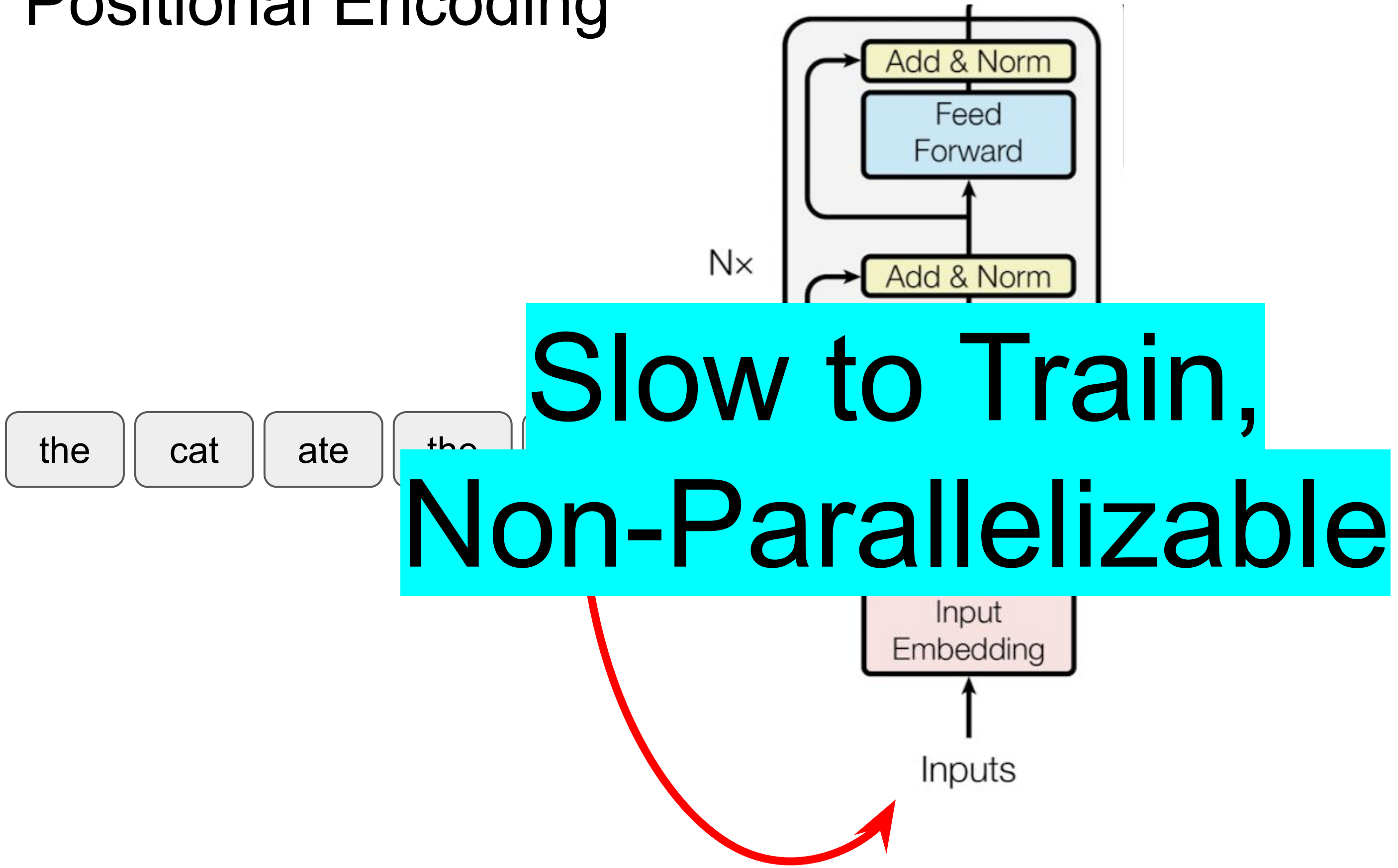
Positional Encoding



Positional Encoding



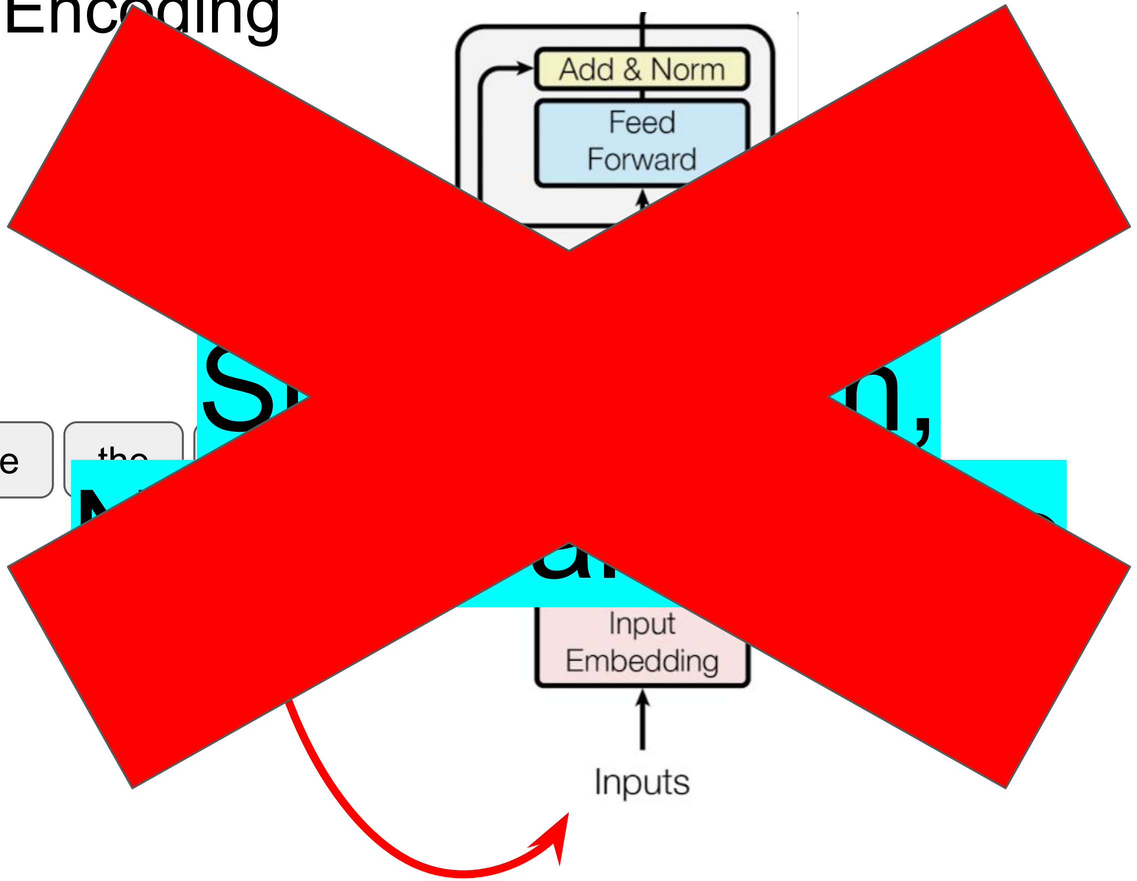
Positional Encoding



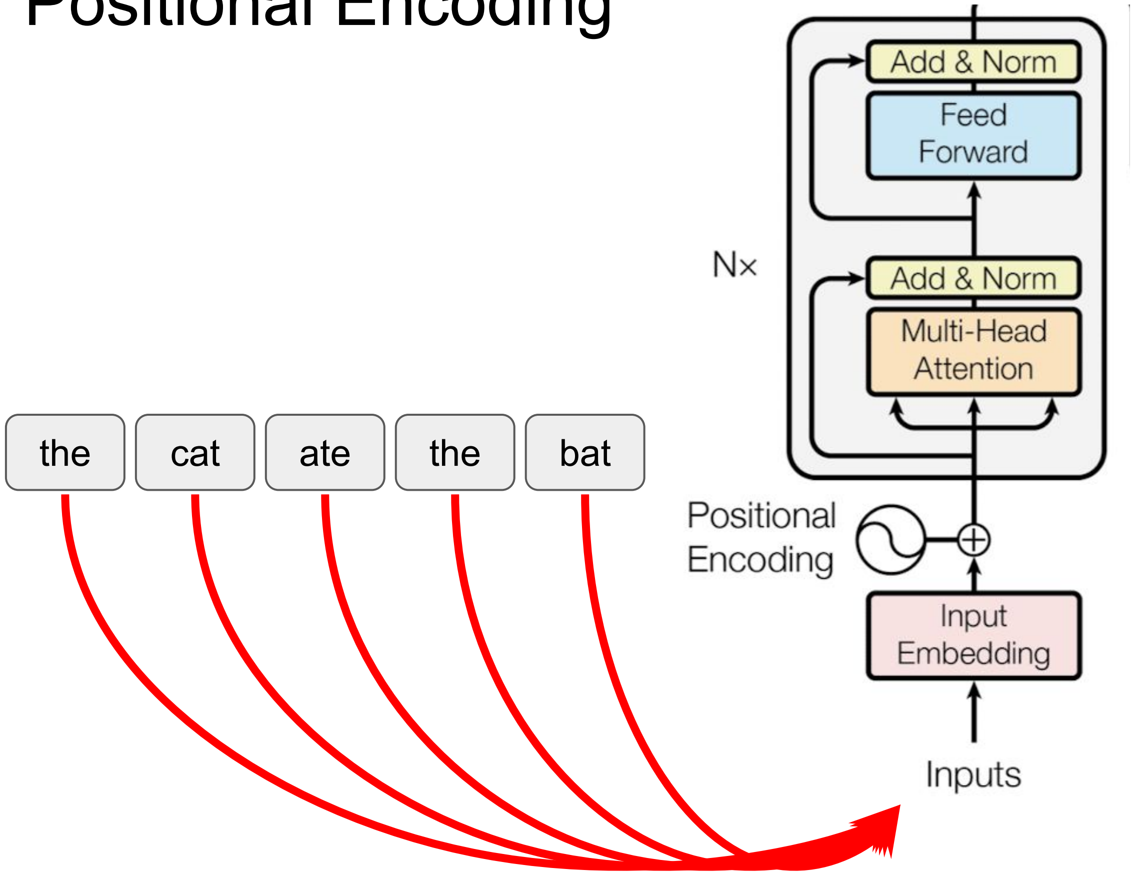
Slow to Train,
Non-Parallelizable

Positional Encoding

the cat ate the



Positional Encoding



Word Embeddings + Positional Encoding

$$d_{model}\left\{\underset{\text{gorgeous}}{\begin{bmatrix}0.1\\0.2\\0.4\\0.2\\0.6\\0.1\end{bmatrix}}\right. + \begin{bmatrix}?\\?\\?\\?\\?\\?\end{bmatrix} \qquad d_{model}\left\{\underset{\text{python}}{\begin{bmatrix}0.2\\0.4\\0.5\\0.9\\0.3\\0.1\end{bmatrix}}\right. + \begin{bmatrix}?\\?\\?\\?\\?\\?\end{bmatrix}$$

Word Embeddings + Positional Encoding

$$d_{model}\left\{\underset{\text{gorgeous}}{\begin{bmatrix}0.1\\0.2\\0.4\\0.2\\0.6\\0.1\end{bmatrix}}\right. + \begin{bmatrix}?\\?\\?\\?\\?\\?\end{bmatrix} \qquad d_{model}\left\{\underset{\text{python}}{\begin{bmatrix}0.2\\0.4\\0.5\\0.9\\0.3\\0.1\end{bmatrix}}\right. + \begin{bmatrix}?\\?\\?\\?\\?\\?\end{bmatrix}$$

Generating Positional Encodings

Positional Encodings

your python code is gorgeous

$$\begin{array}{ccccc}
\begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix} & \begin{bmatrix}0.2\\0.4\\0.5\\0.9\\0.3\\0.1\end{bmatrix} & \begin{bmatrix}0.2\\0.4\\0.6\\1\\0.2\\0.1\end{bmatrix} & \begin{bmatrix}0.1\\0.3\\0.1\\0\\0\\0.7\end{bmatrix} & \begin{bmatrix}0.1\\0.2\\0.4\\0.2\\0.6\\0.1\end{bmatrix} \\
\text{your} & \text{python} & \text{code} & \text{is} & \text{gorgeous}
\end{array}$$

$$\begin{array}{ccccc}
\begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix} & + & \begin{bmatrix}0\\0\\0\\0\\0\\0\end{bmatrix} & = & \begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix} \\
\text{your} & & \textit{position} & &
\end{array}$$

Positional Encodings

your python code is gorgeous

$$\begin{array}{ccccc}
\begin{bmatrix} 0.1 \\ 0 \\ 0.2 \\ 0.1 \\ 0.6 \\ 0.2 \end{bmatrix} &
\begin{bmatrix} 0.2 \\ 0.4 \\ 0.5 \\ 0.9 \\ 0.3 \\ 0.1 \end{bmatrix} &
\begin{bmatrix} 0.2 \\ 0.4 \\ 0.6 \\ 1 \\ 0.2 \\ 0.1 \end{bmatrix} &
\begin{bmatrix} 0.1 \\ 0.3 \\ 0.1 \\ 0 \\ 0 \\ 0.7 \end{bmatrix} &
\begin{bmatrix} 0.1 \\ 0.2 \\ 0.4 \\ 0.2 \\ 0.6 \\ 0.1 \end{bmatrix} \\
\text{your} & \text{python} & \text{code} & \text{is} & \text{gorgeous}
\end{array}$$

$$\begin{array}{ccccc}
\begin{bmatrix} 0.2 \\ 0.4 \\ 0.5 \\ 0.9 \\ 0.3 \\ 0.1 \end{bmatrix} & + &
\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} & = &
\begin{bmatrix} 1.2 \\ 1.4 \\ 1.5 \\ 1.9 \\ 1.3 \\ 1.1 \end{bmatrix} \\
\text{python} & & \textit{position} & &
\end{array}$$

Positional Encodings

your python code is gorgeous

$$\begin{array}{ccccc} \begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix} & \begin{bmatrix}0.2\\0.4\\0.5\\0.9\\0.3\\0.1\end{bmatrix} & \begin{bmatrix}0.2\\0.4\\0.6\\1\\0.2\\0.1\end{bmatrix} & \begin{bmatrix}0.1\\0.3\\0.1\\0\\0\\0.7\end{bmatrix} & \begin{bmatrix}0.1\\0.2\\0.4\\0.2\\0.6\\0.1\end{bmatrix} \\ \text{your} & \text{python} & \text{code} & \text{is} & \text{gorgeous} \end{array}$$

$$\begin{array}{ccccc} \begin{bmatrix}0.2\\0.4\\0.6\\1\\0.2\\0.1\end{bmatrix} & + & \begin{bmatrix}2\\2\\2\\2\\2\\2\end{bmatrix} & = & \begin{bmatrix}2.2\\2.4\\2.6\\3\\2.2\\2.1\end{bmatrix} \\ \text{code} & & \textit{position} & & \end{array}$$

Positional Encodings

Instead of:	0	1	2	3	4
Use:	0	0.25	0.5	0.75	1

Positional Encodings

your python code is gorgeous

$$\underset{\text{your}}{\begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix}}\underset{\text{python}}{\begin{bmatrix}0.2\\0.4\\0.5\\0.9\\0.3\\0.1\end{bmatrix}}\underset{\text{code}}{\begin{bmatrix}0.2\\0.4\\0.6\\1\\0.2\\0.1\end{bmatrix}}\underset{\text{is}}{\begin{bmatrix}0.1\\0.3\\0.1\\0\\0\\0.7\end{bmatrix}}\underset{\text{gorgeous}}{\begin{bmatrix}0.1\\0.2\\0.4\\0.2\\0.6\\0.1\end{bmatrix}}$$

$$\underset{\text{your}}{\begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix}}+\underset{\textit{position}}{\begin{bmatrix}0\\0\\0\\0\\0\\0\end{bmatrix}}=\begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix}$$

Positional Encodings

your python code is gorgeous

$$\begin{array}{ccccc}
\begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix} &
\begin{bmatrix}0.2\\0.4\\0.5\\0.9\\0.3\\0.1\end{bmatrix} &
\begin{bmatrix}0.2\\0.4\\0.6\\1\\0.2\\0.1\end{bmatrix} &
\begin{bmatrix}0.1\\0.3\\0.1\\0\\0\\0.7\end{bmatrix} &
\begin{bmatrix}0.1\\0.2\\0.4\\0.2\\0.6\\0.1\end{bmatrix} \\
\text{your} & \text{python} & \text{code} & \text{is} & \text{gorgeous}
\end{array}$$

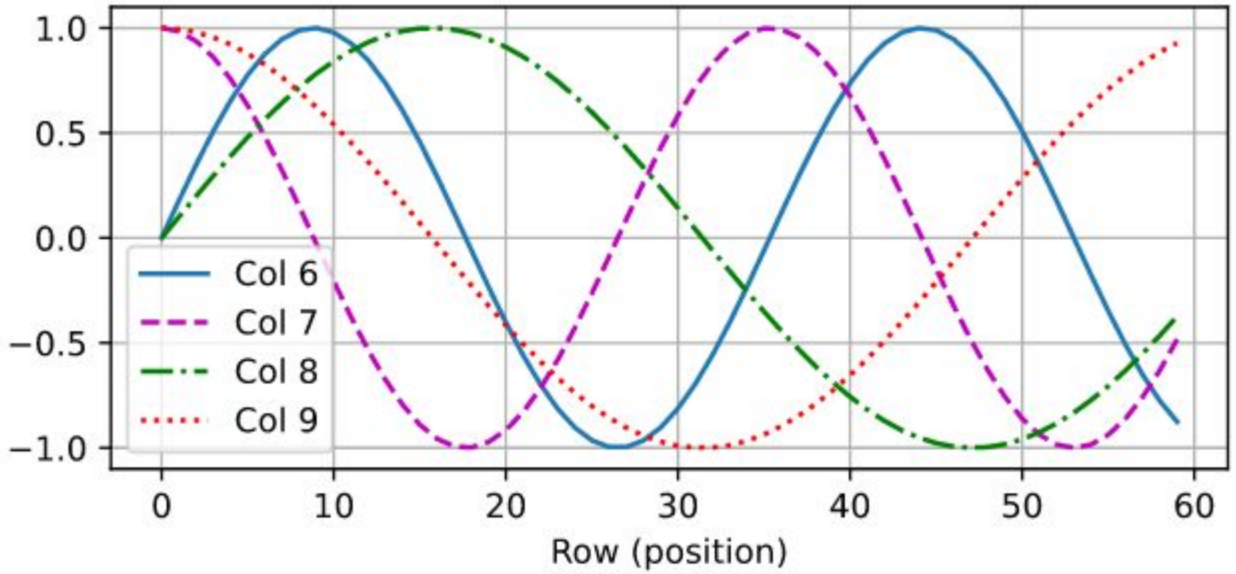
$$\begin{array}{ccccc}
\begin{bmatrix}0.2\\0.4\\0.5\\0.9\\0.3\\0.1\end{bmatrix} & + &
\begin{bmatrix}0.25\\0.25\\0.25\\0.25\\0.25\\0.25\end{bmatrix} & = &
\begin{bmatrix}0.45\\0.65\\0.75\\1.15\\0.55\\0.35\end{bmatrix} \\
\text{python} & & \textit{position} & &
\end{array}$$

Positional Encoding

Things we want:

- Encode position
- No large numbers
- Allow for variable sequence length

Positional Encodings



Positional Encoding

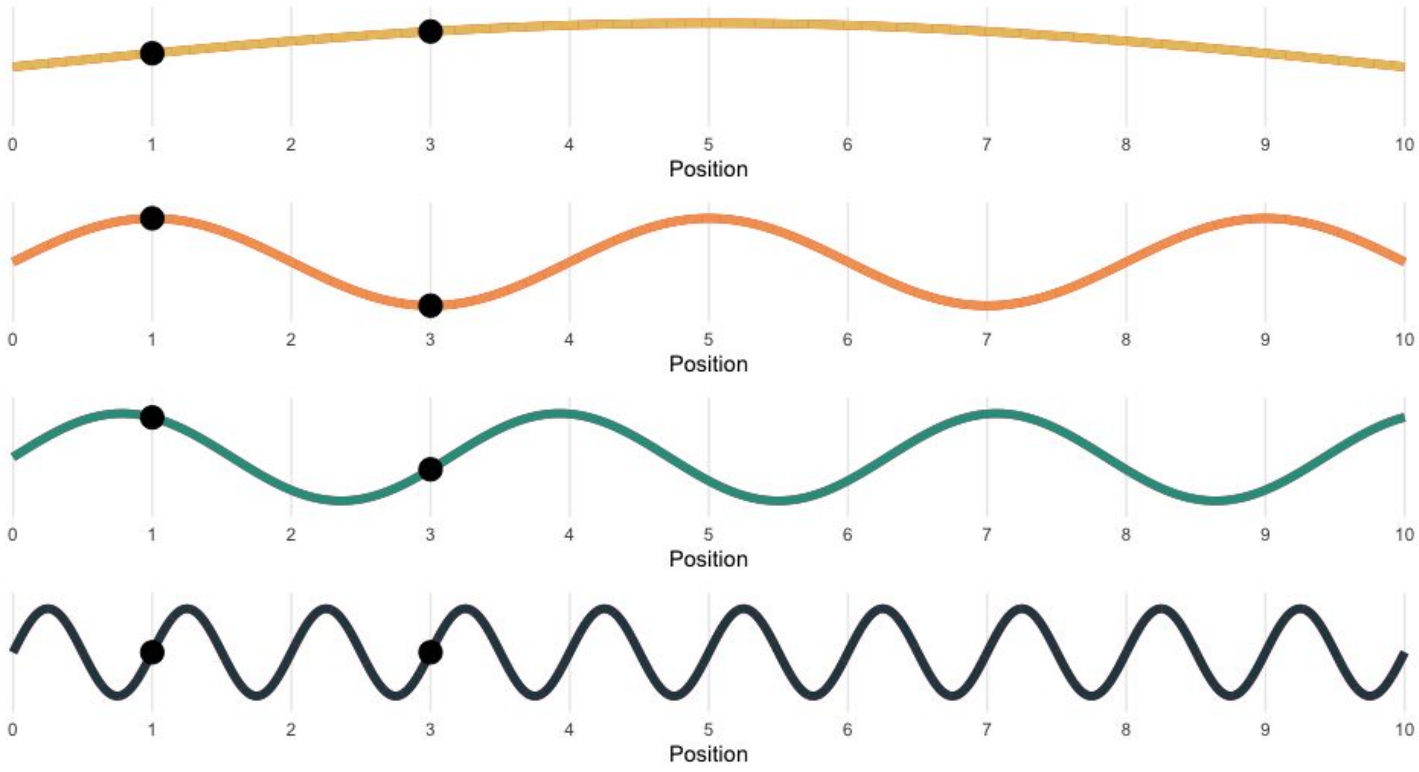
Pros

- sin() and cos() are bounded between -1, 1
- sin() and cos() can go on forever

Cons

- sin() and cos() are periodic

Positional Encoding



Positional Encoding

$$sin(\boxed{pos}/10000^{\frac{2*i}{d}})$$

$$cos(\boxed{pos}/10000^{\frac{2*i}{d}})$$

Positional Encoding

$$sin(pos/10000^{\frac{2*i}{d}})$$

$$cos(pos/10000^{\frac{2*i}{d}})$$

Positional Encoding

$$sin(pos/10000^{\frac{2*i}{d}})$$

$$cos(pos/10000^{\frac{2*i}{d}})$$

Positional Encoding

$$d_{model}\left\{\begin{bmatrix} sin(pos/10000^{\frac{2*0}{d}}) \\ cos(pos/10000^{\frac{2*0}{d}}) \\ \\ sin(pos/10000^{\frac{2*1}{d}}) \\ cos(pos/10000^{\frac{2*1}{d}}) \\ \\ \ldots \\ \\ sin(pos/10000^{\frac{2*j}{d}}) \\ cos(pos/10000^{\frac{2*j}{d}}) \end{bmatrix}\right.$$

$$j = d_{model}//2$$

Positional Encoding

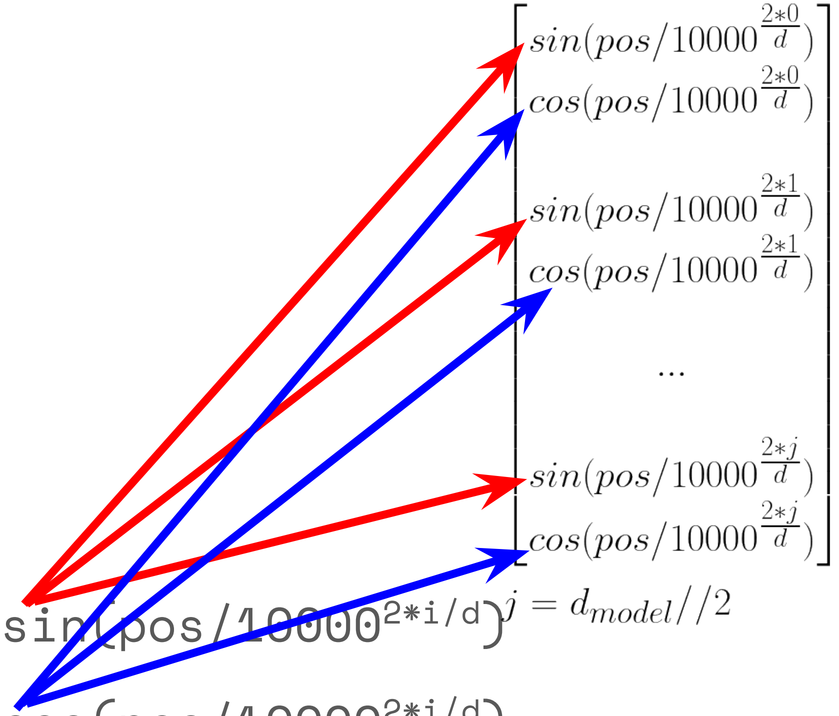
```
pos = 0

d = 6

pos_encoding = np.zeros(d)

for i in range(d//2):

    pos_encoding[2*i]     = sin(pos/10000^{2*i/d})

    pos_encoding[2*i + 1] = cos(pos/10000^{2*i/d})
```

$$\begin{bmatrix} sin(pos/10000^{\frac{2*0}{d}}) \\ cos(pos/10000^{\frac{2*0}{d}}) \\ \\ sin(pos/10000^{\frac{2*1}{d}}) \\ cos(pos/10000^{\frac{2*1}{d}}) \\ \\ \cdots \\ \\ sin(pos/10000^{\frac{2*j}{d}}) \\ cos(pos/10000^{\frac{2*j}{d}}) \end{bmatrix}$$

$$j = d_{model}//2$$

Positional Encoding

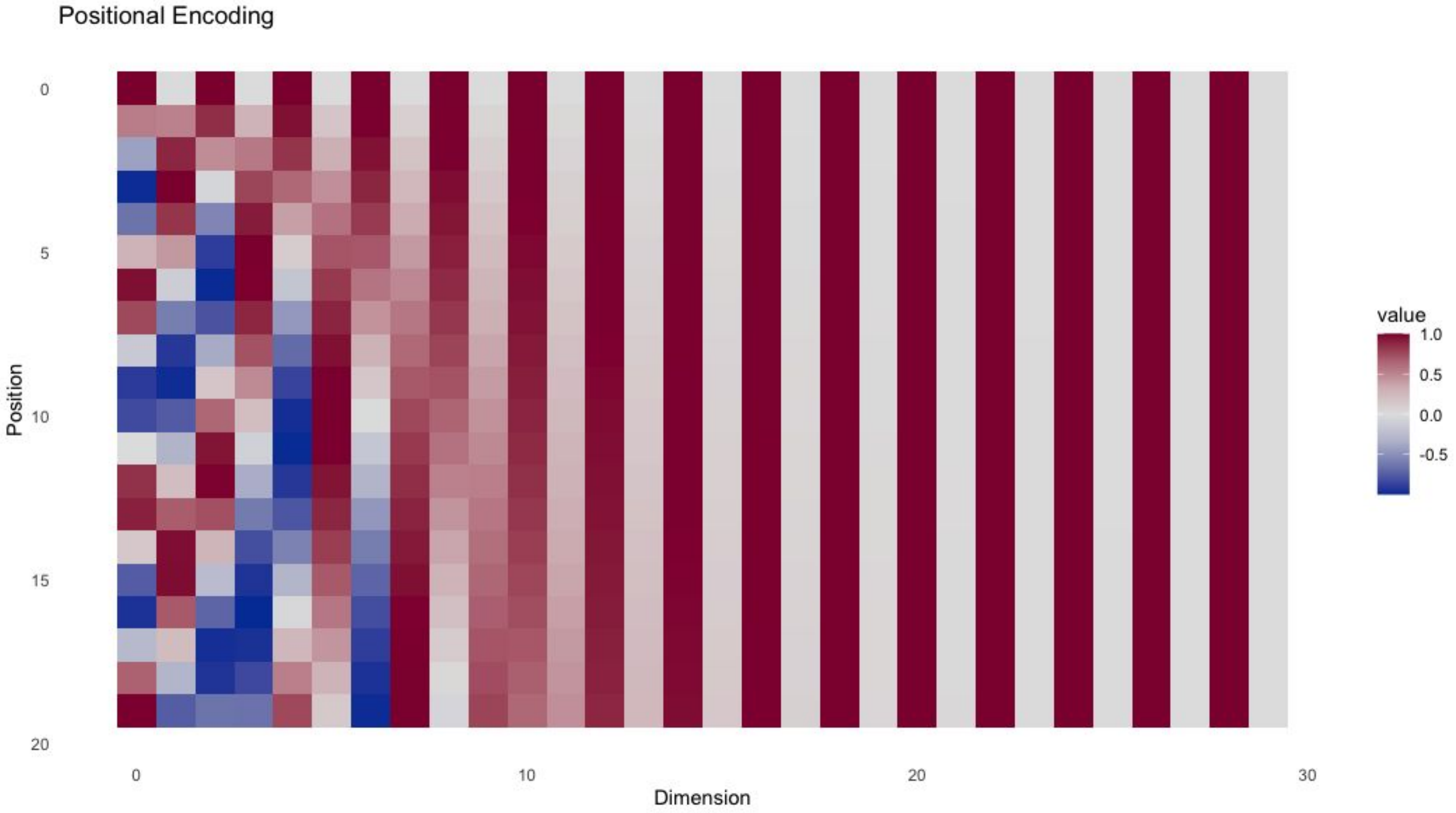
```
pos = 0

d = 6

pos_encoding = np.zeros(d)

for i in range(d//2):

    pos_encoding[2*i]     = sin(pos/10000^{2*i/d})

    pos_encoding[2*i + 1] = cos(pos/10000^{2*i/d})
```



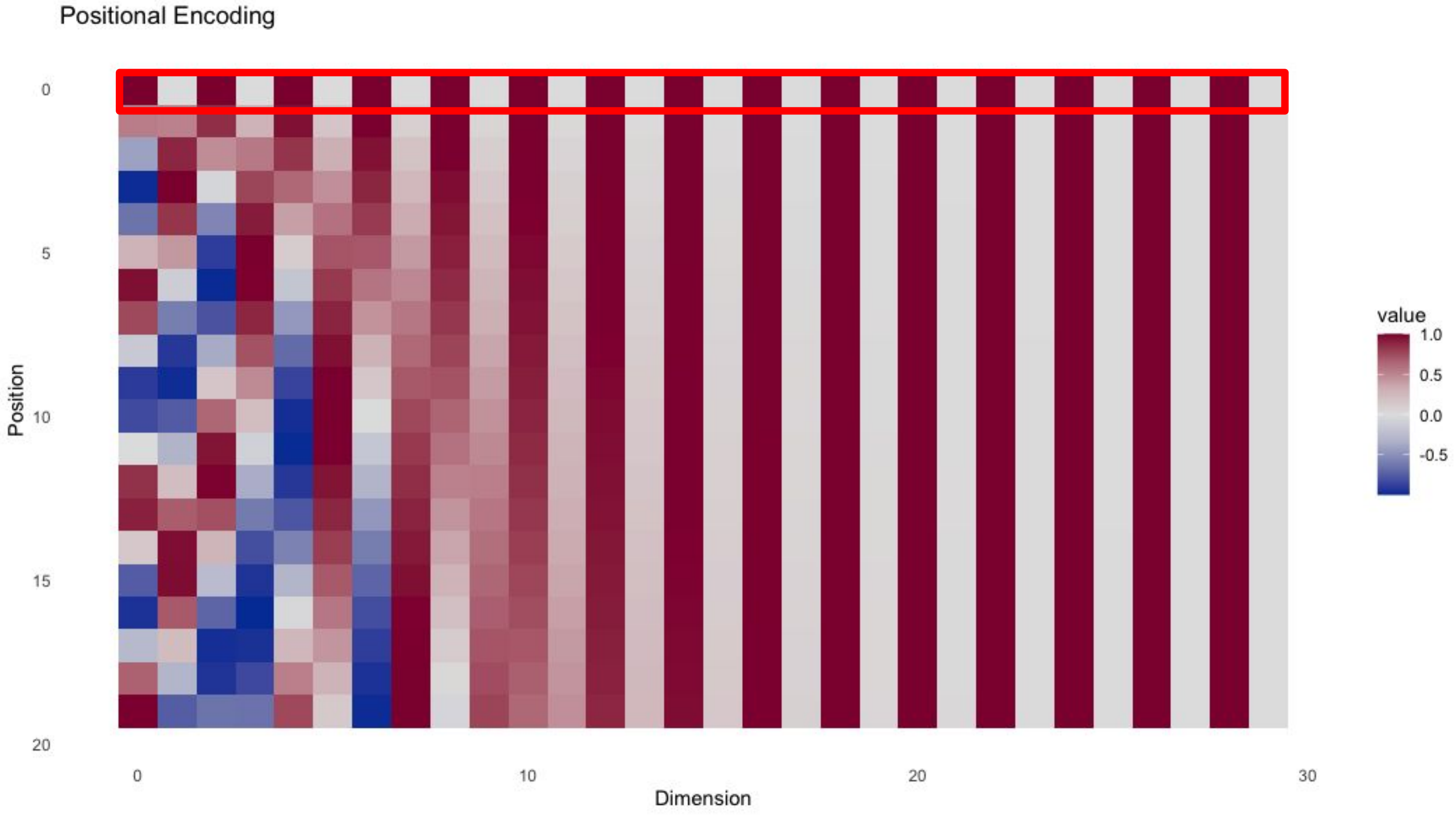
$$\begin{bmatrix} sin(pos/10000^{\frac{2*0}{d}}) \\ cos(pos/10000^{\frac{2*0}{d}}) \\ sin(pos/10000^{\frac{2*1}{d}}) \\ cos(pos/10000^{\frac{2*1}{d}}) \\ \cdots \\ sin(pos/10000^{\frac{2*j}{d}}) \\ cos(pos/10000^{\frac{2*j}{d}}) \end{bmatrix}$$

$j = d_{model}//2$

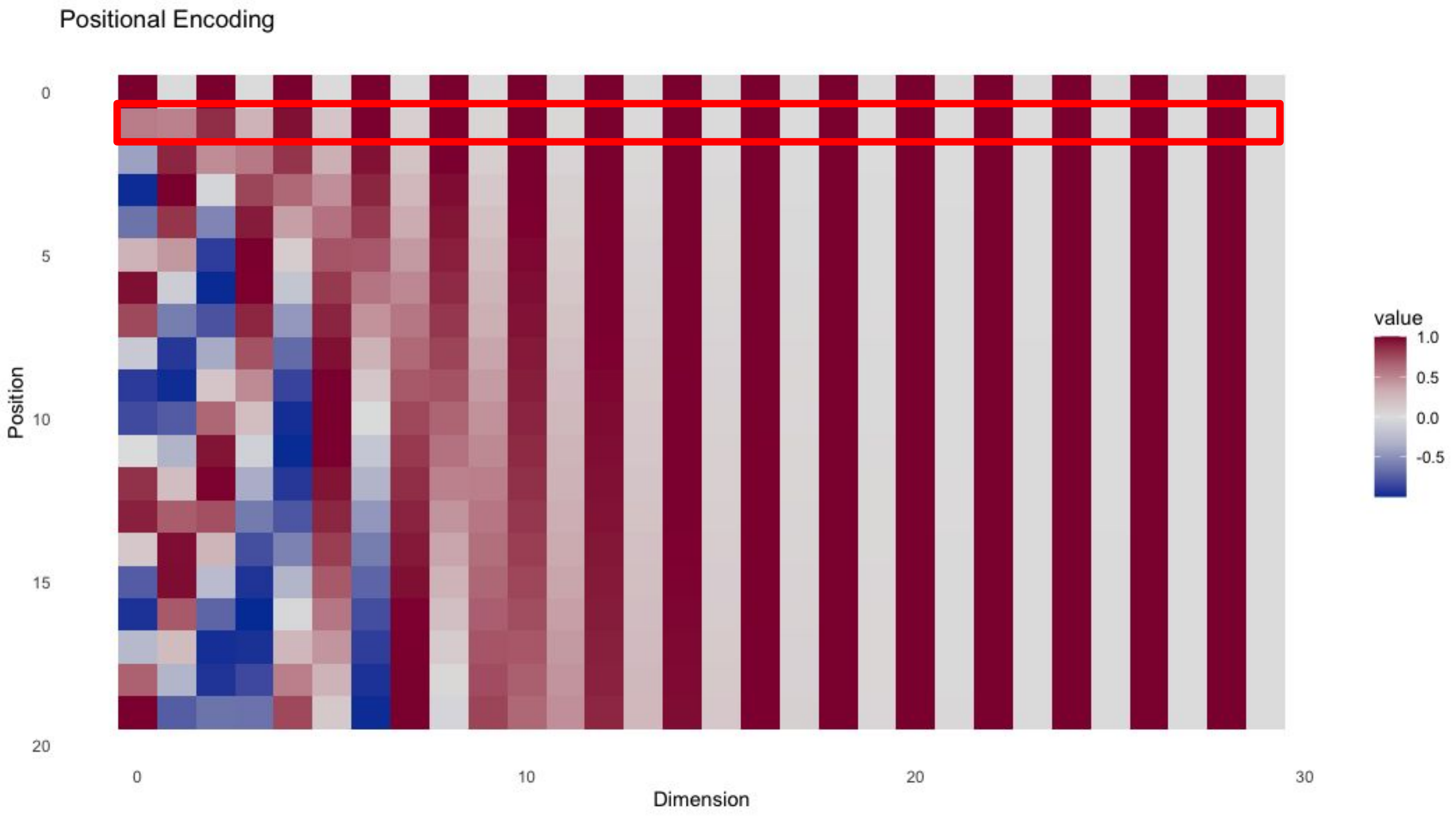
Positional Encoding



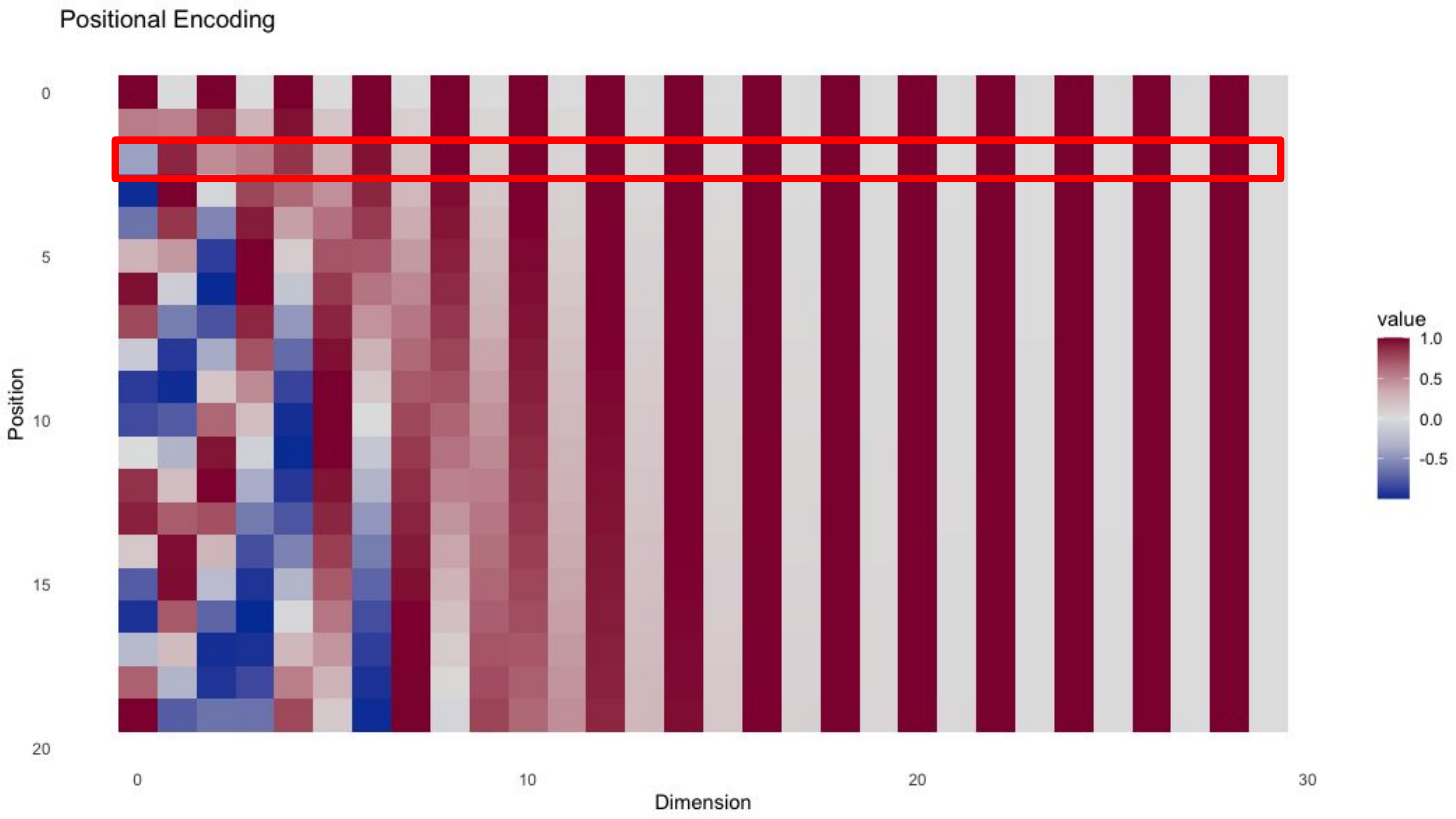
Positional Encoding



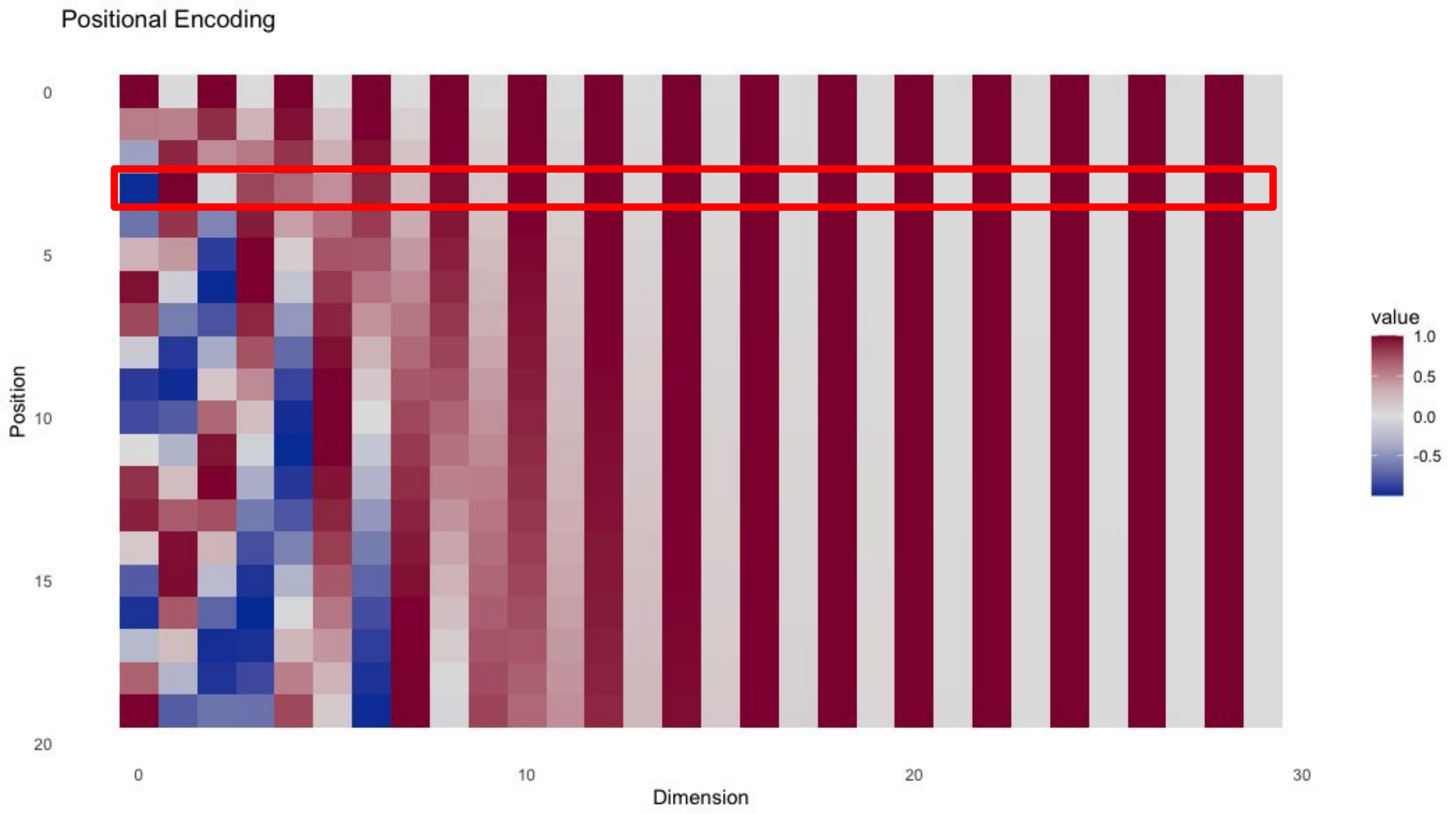
Positional Encoding



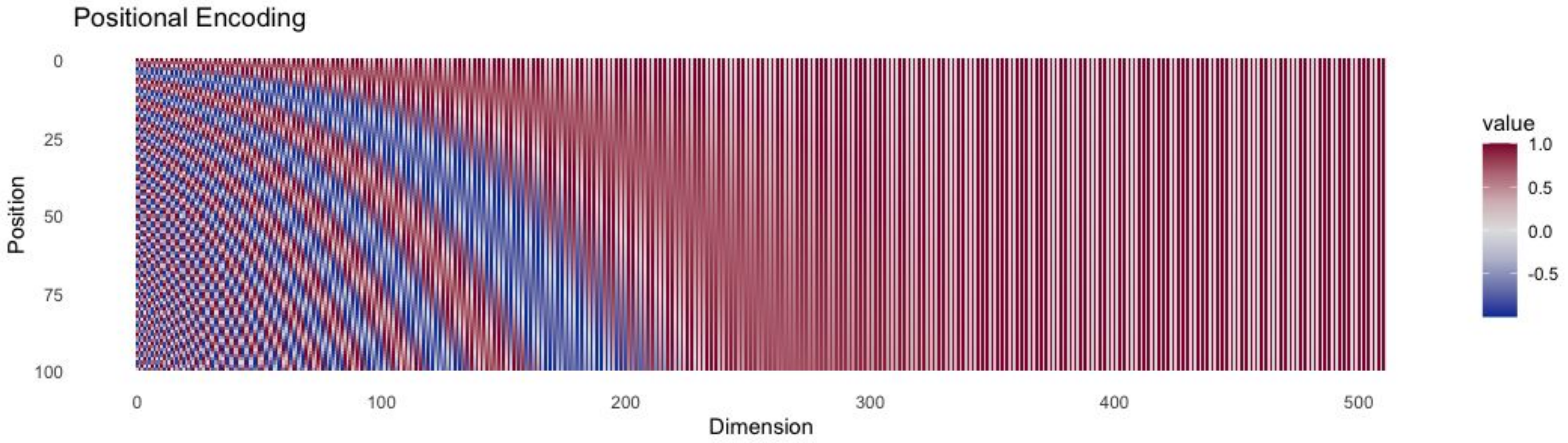
Positional Encoding



Positional Encoding



Positional Encoding



Positional Encodings

your python code is gorgeous

$$\begin{array}{ccccc} \begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix} & \begin{bmatrix}0.2\\0.4\\0.5\\0.9\\0.3\\0.1\end{bmatrix} & \begin{bmatrix}0.2\\0.4\\0.6\\1\\0.2\\0.1\end{bmatrix} & \begin{bmatrix}0.1\\0.3\\0.1\\0\\0\\0.7\end{bmatrix} & \begin{bmatrix}0.1\\0.2\\0.4\\0.2\\0.6\\0.1\end{bmatrix} \\ \text{your} & \text{python} & \text{code} & \text{is} & \text{gorgeous}\end{array} \longrightarrow \begin{array}{ccc} \begin{bmatrix}0.1\\0\\0.2\\0.1\\0.6\\0.2\end{bmatrix} & + & \begin{bmatrix} sin(0/10000^{\frac{2*1}{6}})\\ cos(0/10000^{\frac{2*1}{6}})\\ sin(0/10000^{\frac{2*2}{6}})\\ cos(0/10000^{\frac{2*2}{6}})\\ sin(0/10000^{\frac{2*3}{6}})\\ cos(0/10000^{\frac{2*3}{6}}) \end{bmatrix} \\ \text{your} & & \textit{position}\end{array}$$

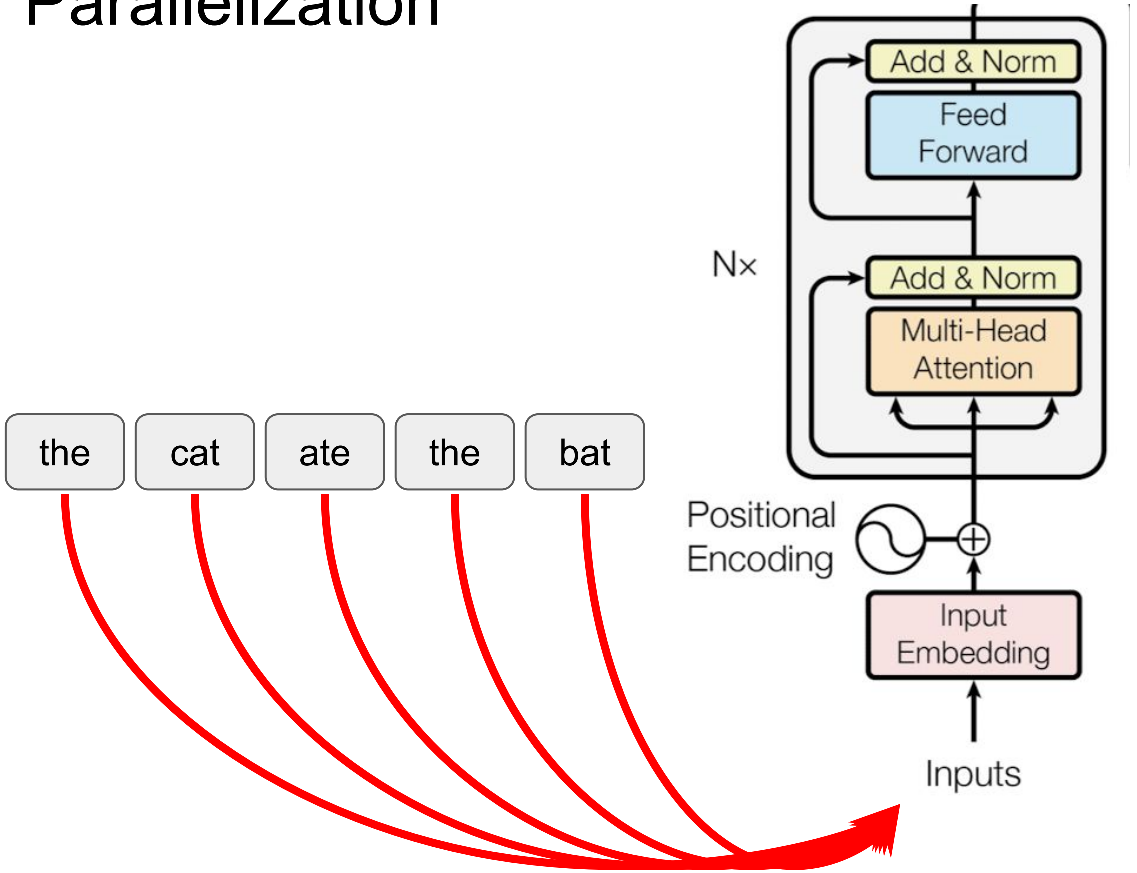
Positional Encodings

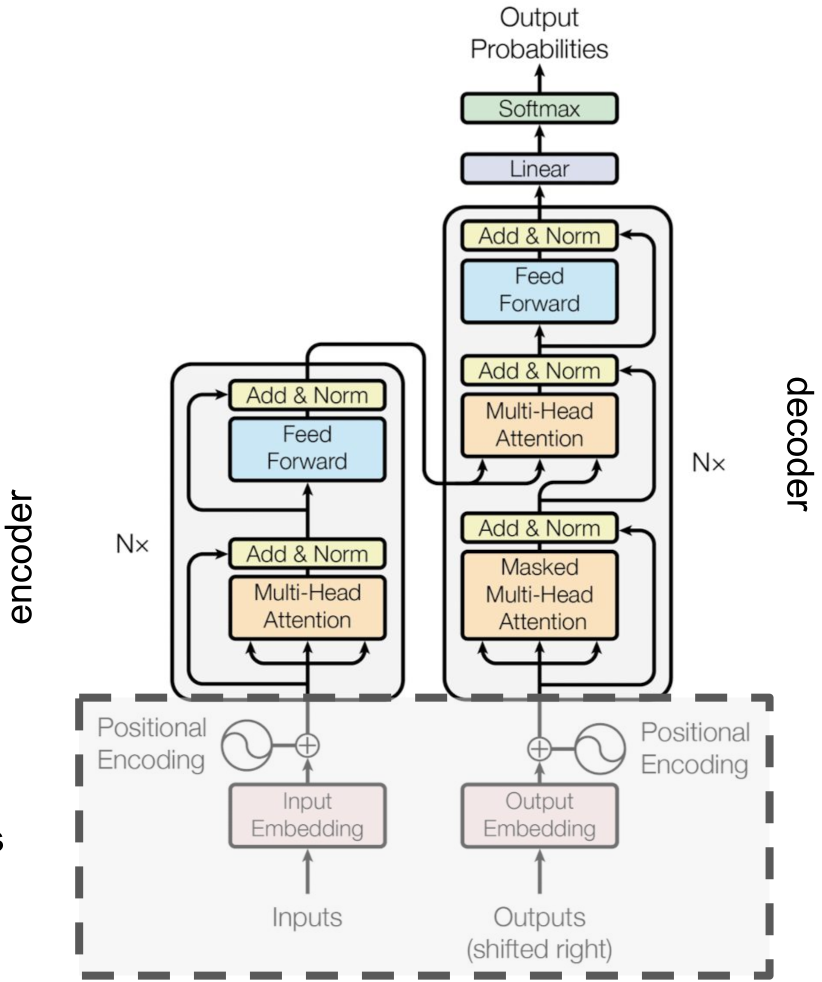
your python code is gorgeous

$$\begin{array}{ccccc} \begin{bmatrix} 0.1 \\ 0 \\ 0.2 \\ 0.1 \\ 0.6 \\ 0.2 \end{bmatrix} & \begin{bmatrix} 0.2 \\ 0.4 \\ 0.5 \\ 0.9 \\ 0.3 \\ 0.1 \end{bmatrix} & \begin{bmatrix} 0.2 \\ 0.4 \\ 0.6 \\ 1 \\ 0.2 \\ 0.1 \end{bmatrix} & \begin{bmatrix} 0.1 \\ 0.3 \\ 0.1 \\ 0 \\ 0 \\ 0.7 \end{bmatrix} & \begin{bmatrix} 0.1 \\ 0.2 \\ 0.4 \\ 0.2 \\ 0.6 \\ 0.1 \end{bmatrix} \\ \text{your} & \text{python} & \text{code} & \text{is} & \text{gorgeous} \end{array}$$

$$\begin{array}{ccc} \begin{bmatrix} 0.2 \\ 0.4 \\ 0.5 \\ 0.9 \\ 0.3 \\ 0.1 \end{bmatrix} & + & \begin{bmatrix} sin(1/10000^{\frac{2*1}{6}}) \\ cos(1/10000^{\frac{2*1}{6}}) \\ sin(1/10000^{\frac{2*2}{6}}) \\ cos(1/10000^{\frac{2*2}{6}}) \\ sin(1/10000^{\frac{2*3}{6}}) \\ cos(1/10000^{\frac{2*3}{6}}) \end{bmatrix} \\ \text{python} & & \textit{position} \end{array}$$

Parallelization





Today's Focus

Figure 1: The Transformer - model architecture.

Image from: Attention is all you Need (Vaswani et. al 2017)

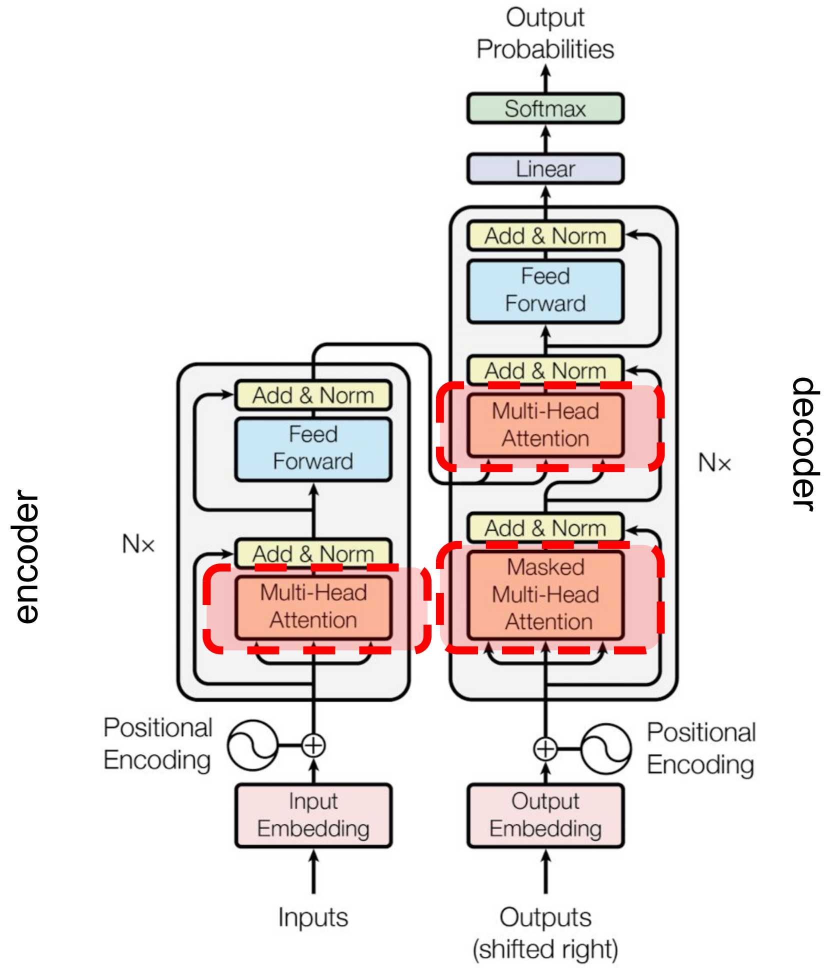


Figure 1: The Transformer - model architecture.

Image from: Attention is all you Need (Vaswani et. al 2017)