18. Transformers: Part 2

Generative Music AI

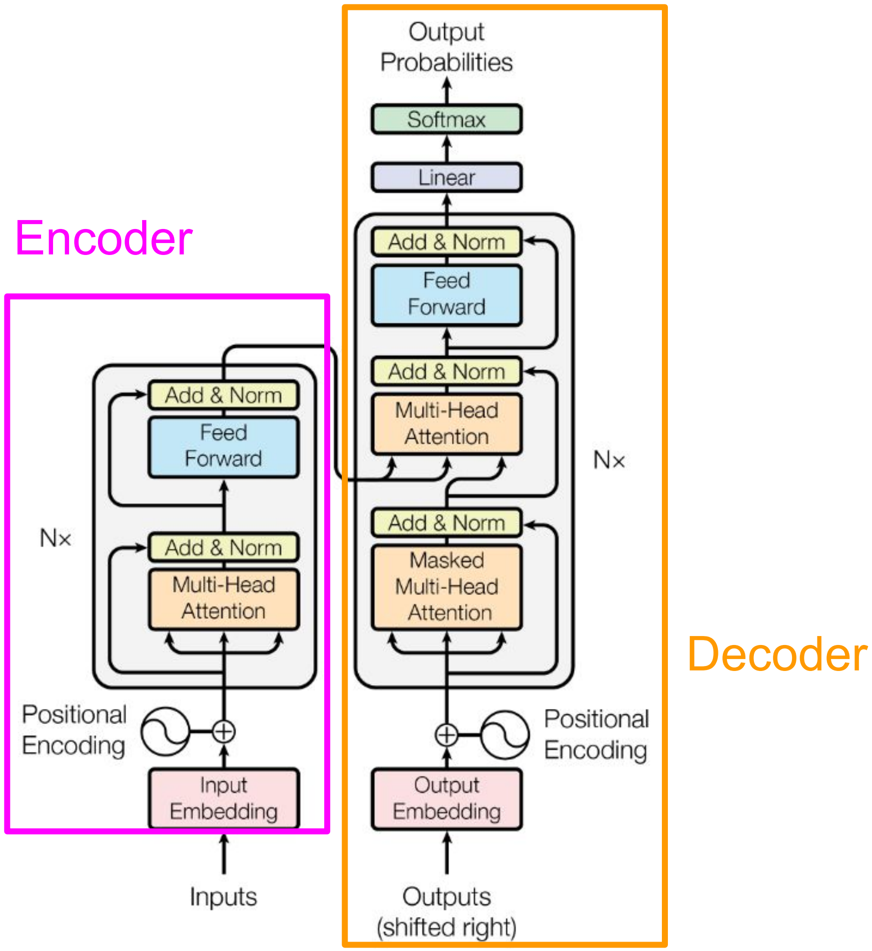
THE SOUND OF AI

Universitat Pompeu Fabra Barcelona

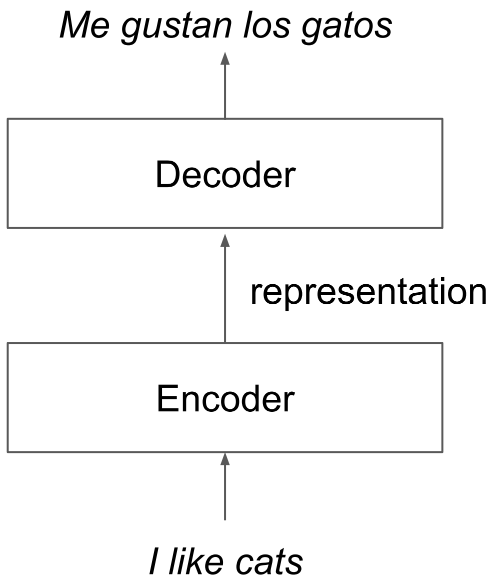
MTG Music Technology Group

Overview

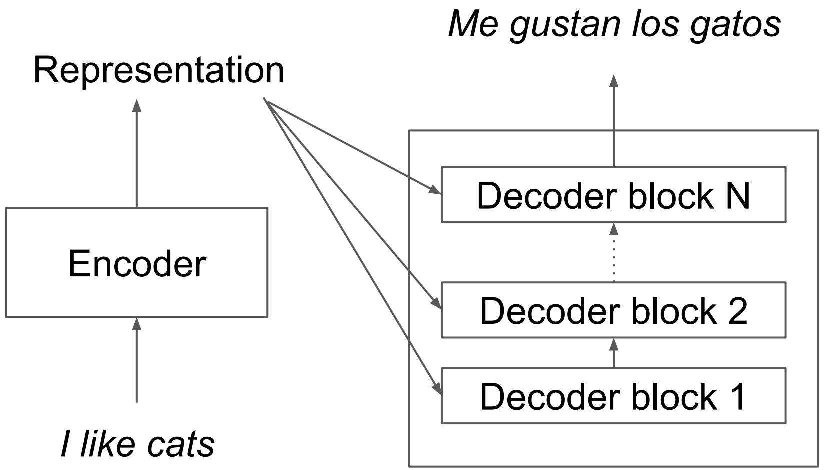
1. Decoder intuition
2. Decoder block
3. Masked multi-head attention
4. Multi-head attention
5. Linear & softmax layers
6. Decoder step-by-step
7. Training a transformer
8. Music generation with transformers
9. Pros and cons
10. Promising lines of research


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Add & Norm
Masked
Multi-Head
Attention
Positional
Encoding
Output
Embedding
Outputs
(shifted right)
Encoder
Add & Norm
Feed
Forward
Nx
Add & Norm
Multi-Head
Attention
Positional
Encoding
Input
Embedding
Inputs
Decoder

The intuition

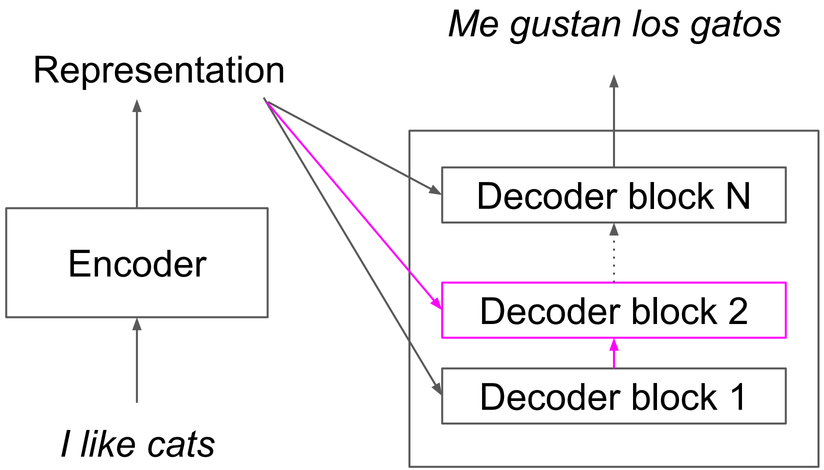


Decoder



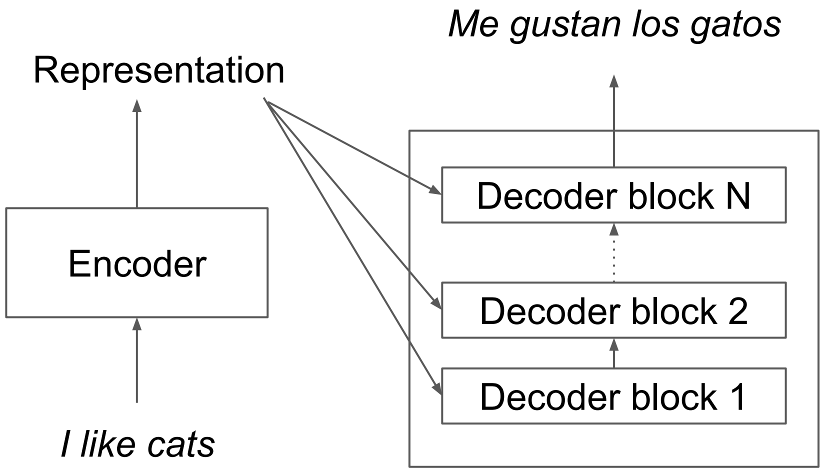
- Stack of *N* decoder blocks

Decoder



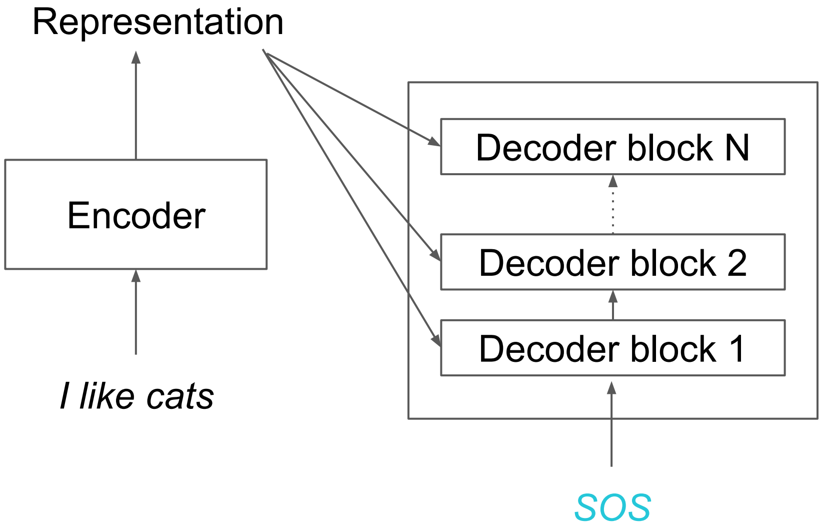
- Stack of *N* decoder blocks
- 2 inputs: encoder representation + output previous decoder

Decoder



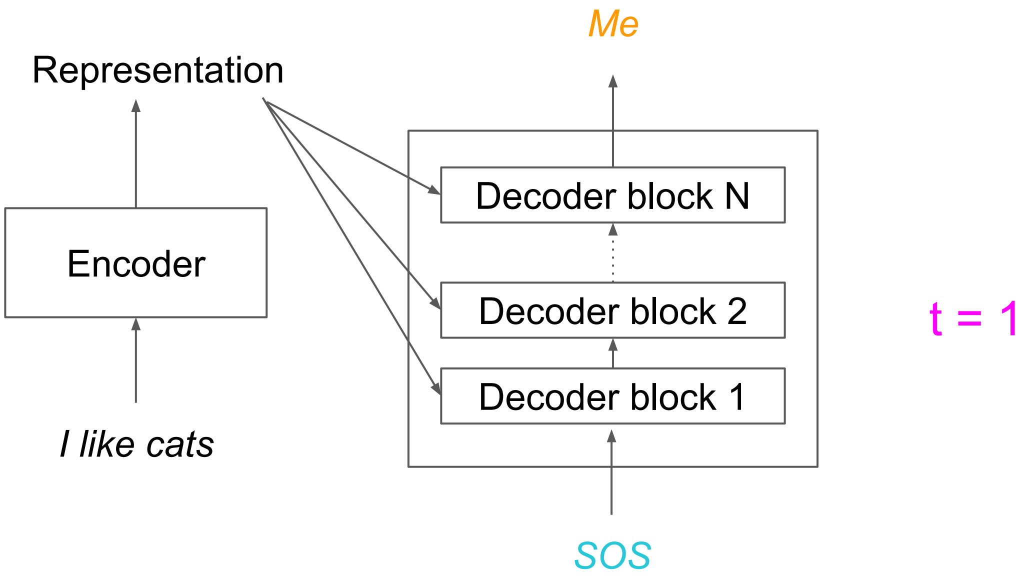
- Stack of *N* decoder blocks
- 2 inputs: encoder representation + output previous decoder
- Generate output in steps (autoregressive)

Decoder: Output generation

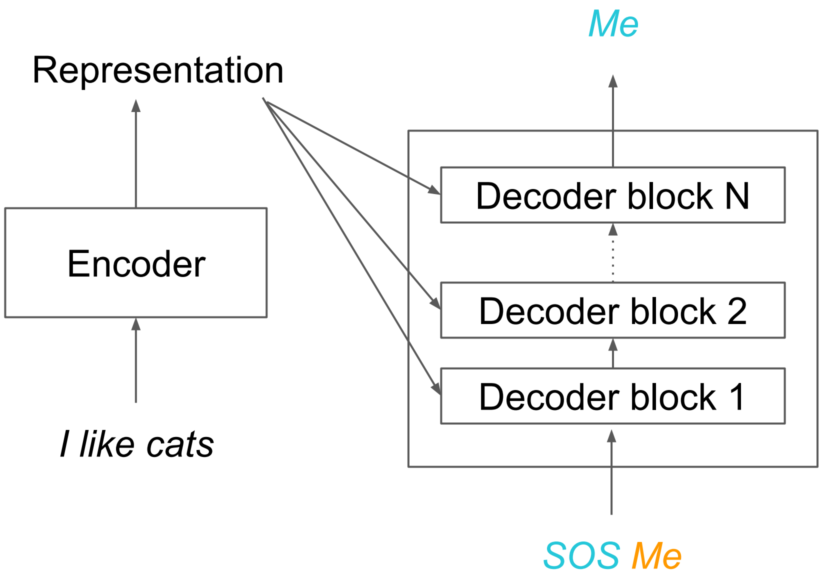


t = 1

Decoder: Output generation

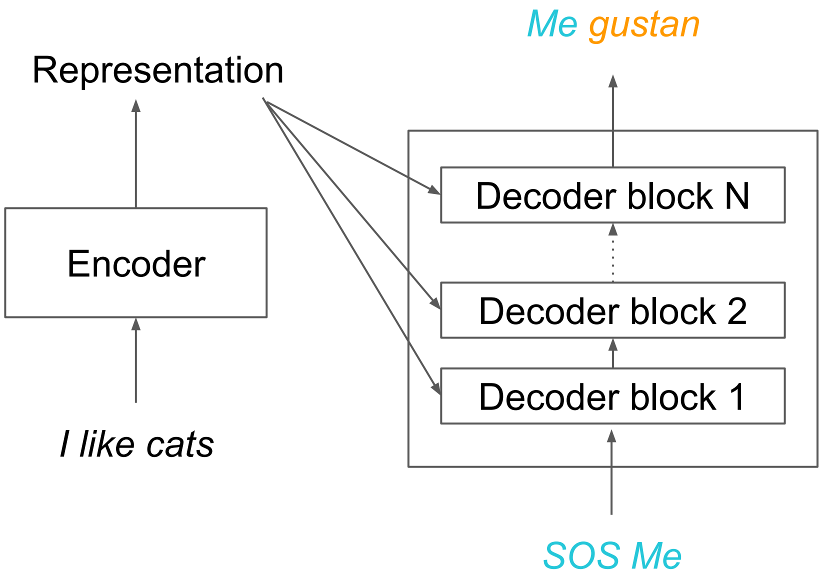


Decoder: Output generation



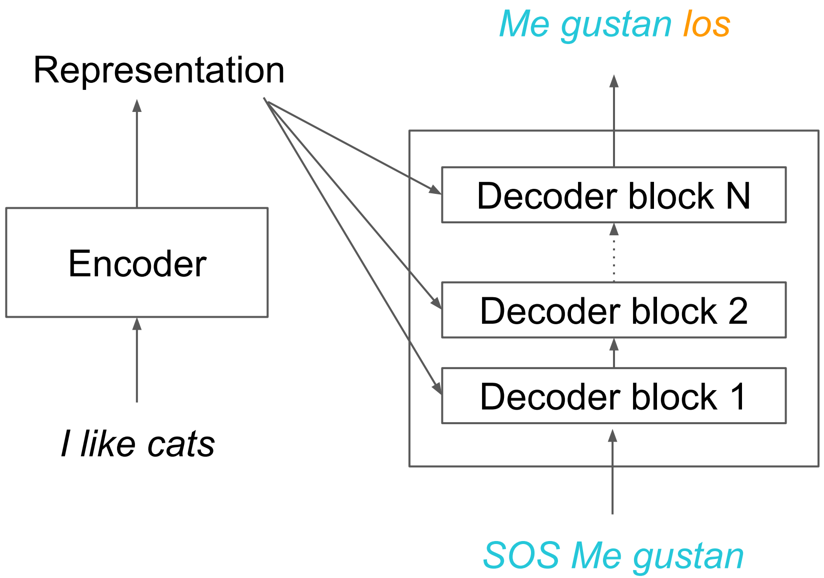
t = 2

Decoder: Output generation



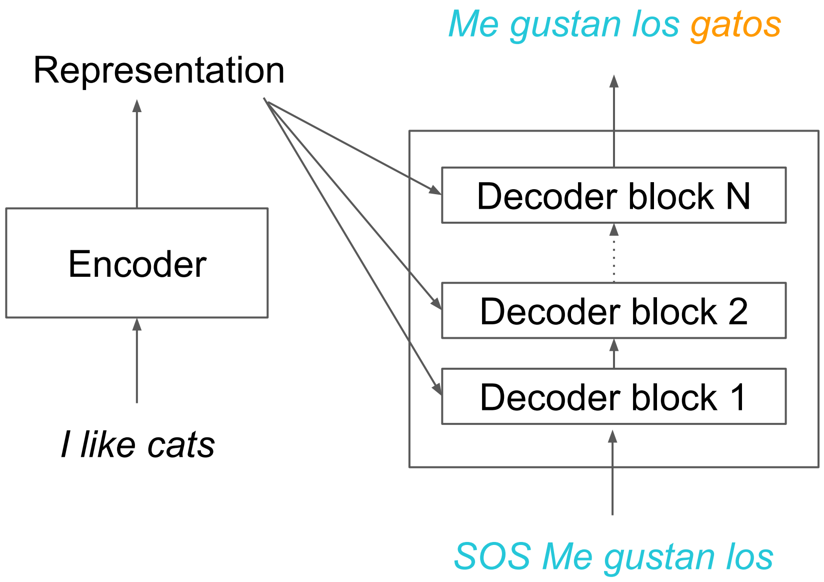
t = 2

Decoder: Output generation



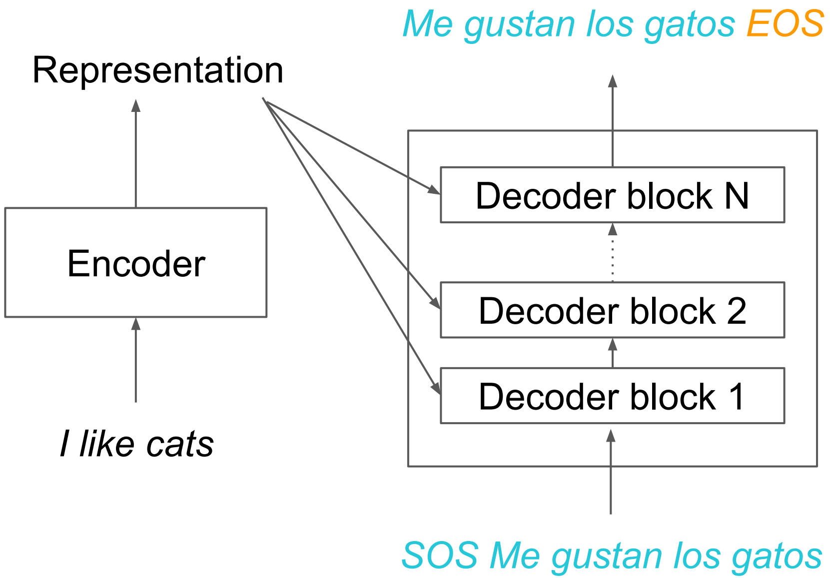
t = 3

Decoder: Output generation



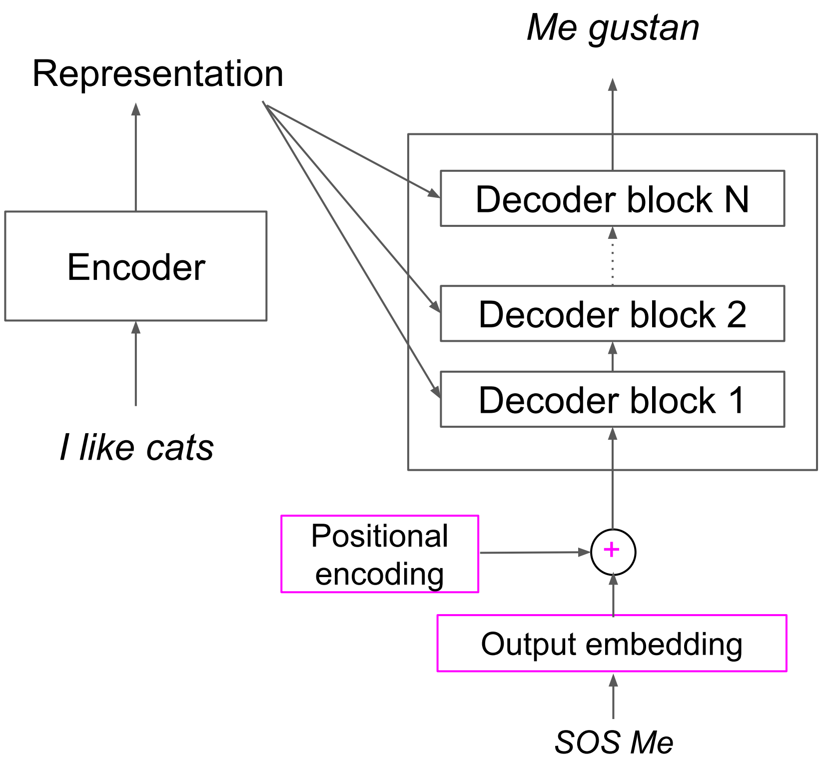
t = 4

Decoder: Output generation

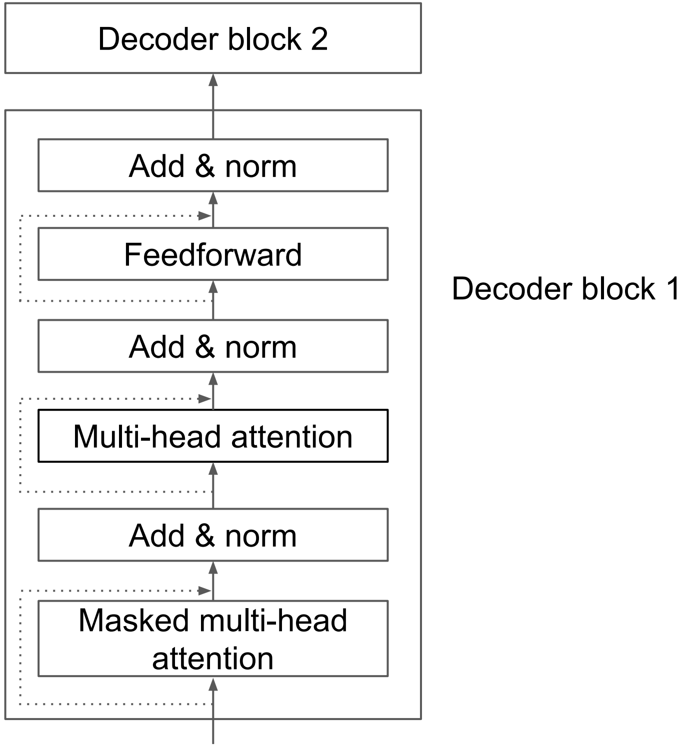


t = 5

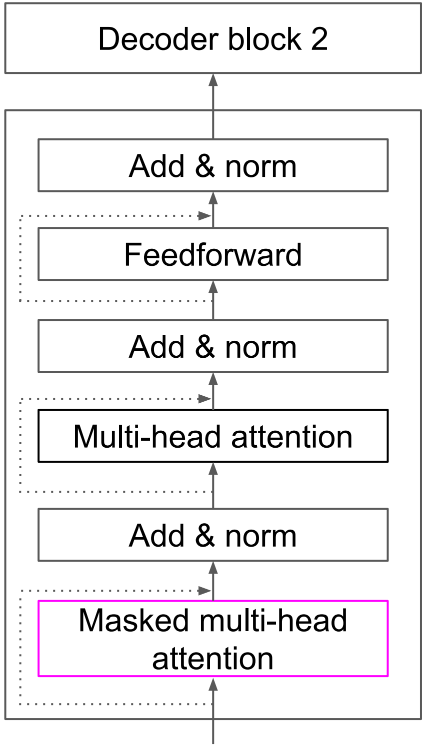
Decoder input



Decoder block



Decoder block



Decoder block 1

Training dataset

I love cats	Me gustan los gatos
I take a shower	Me ducho
How are you?	Que tal?

Training

- Pass entire target sentence as an embedding to the decoder

Training

- Pass entire target sentence as an embedding to the decoder
- Add SOS at the beginning

Training

- Pass entire target sentence as an embedding to the decoder
- Add SOS at the beginning

$$I = \begin{array}{c} \text{SOS} \\ \text{Me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} \begin{bmatrix} 0.2 & 1.2 \\ 0.5 & 4.1 \\ 1.2 & 1.2 \\ 3.5 & 2.1 \\ 2.2 & 3.4 \end{bmatrix}$$

Training problem

- Self attention relates each word to all other words

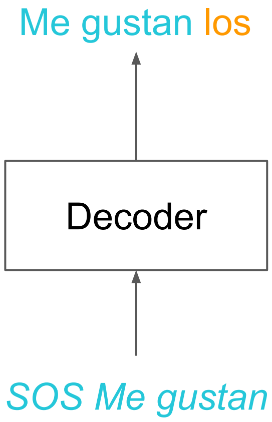
Training problem

- Self attention relates each word to all other words
- Decoder generates output one word at a time

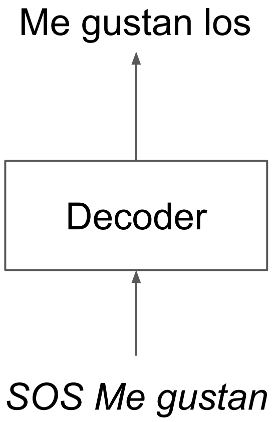
Training problem

- Self attention relates each word to all other words
- Decoder generates output one word at a time
- Decoder knows about future generated words (information leakage)

Training / inference discrepancy



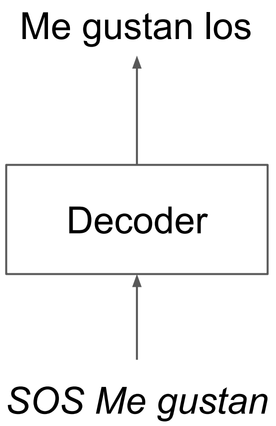
Training / inference discrepancy



What decoder knows during inference

SOS me gustan

Training / inference discrepancy



What decoder knows during inference

SOS me gustan

What decoder knows during training

SOS me gustan los gatos


MASKED MULTI-HEAD ATTENTION
imgflip.com

Masked multi-head attention

$$Z_i(Q_i, K_i, V_i) = \mathrm{softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_k}}\right) V_i$$

Masked multi-head attention

$$Z_i(Q_i, K_i, V_i) = \text{softmax}\left(\frac{\boxed{Q_i K_i^T}}{\sqrt{d_k}}\right) V_i$$

Masked multi-head attention

$$Z_i(Q_i, K_i, V_i) = \mathrm{softmax}\left(\boxed{\frac{Q_i K_i^T}{\sqrt{d_k}}}\right) V_i$$

Masked multi-head attention

$$Z_i(Q_i, K_i, V_i) = \boxed{\text{softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_k}}\right)} V_i$$

Masked multi-head attention

$$Z_i(Q_i, K_i, V_i) = \boxed{\mathrm{softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_k}}\right) V_i}$$

Masked multi-head attention

$$Z_i(Q_i, K_i, V_i) = \mathrm{softmax}\left(\boxed{\frac{Q_i K_i^T}{\sqrt{d_k}}}\right) V_i$$

Masked multi-head attention

$$\frac{Q_i K_i^T}{\sqrt{d_k}} = \begin{array}{c|c} & \begin{array}{ccccc} \text{SOS} & \text{me} & \text{gustan} & \text{los} & \text{gatos} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} 1.3 & 0.8 & 1.3 & 2.8 & 2.3 \\ 2.4 & 2.8 & 2.3 & 6.8 & 1.9 \\ 1.6 & 7.4 & 1.6 & 0.3 & 0.5 \\ 2.1 & 1.2 & 9.3 & 5.2 & 0.2 \\ 4.3 & 3.8 & 6.3 & 1.8 & 2.3 \end{bmatrix} \end{array}$$

Masked multi-head attention

$$\frac{Q_i K_i^T}{\sqrt{d_k}} = \begin{array}{c c} & \begin{array}{ccccc} \text{SOS} & \text{me} & \text{gustan} & \text{los} & \text{gatos} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} 1.3 & 0.8 & 1.3 & 2.8 & 2.3 \\ 2.4 & 2.8 & \boxed{2.3} & 6.8 & 1.9 \\ 1.6 & 7.4 & 1.6 & 0.3 & 0.5 \\ 2.1 & 1.2 & 9.3 & 5.2 & 0.2 \\ 4.3 & 3.8 & 6.3 & 1.8 & 2.3 \end{bmatrix} \end{array}$$

Masked multi-head attention

$$\frac{Q_i K_i^T}{\sqrt{d_k}} = \begin{array}{c c} & \begin{array}{ccccc} \text{SOS} & \text{me} & \text{gustan} & \text{los} & \text{gatos} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} \boxed{1.3 \quad 0.8 \quad 1.3 \quad 2.8 \quad 2.3} \\ 2.4 \quad 2.8 \quad 2.3 \quad 6.8 \quad 1.9 \\ 1.6 \quad 7.4 \quad 1.6 \quad 0.3 \quad 0.5 \\ 2.1 \quad 1.2 \quad 9.3 \quad 5.2 \quad 0.2 \\ 4.3 \quad 3.8 \quad 6.3 \quad 1.8 \quad 2.3 \end{bmatrix} \end{array}$$

Masked multi-head attention

$$\frac{Q_i K_i^T}{\sqrt{d_k}} = \begin{array}{c c} & \begin{array}{ccccc} \text{SOS} & \text{me} & \text{gustan} & \text{los} & \text{gatos} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} 1.3 & \cancel{0.8} & \cancel{1.3} & \cancel{2.8} & \cancel{2.3} \\ 2.4 & 2.8 & 2.3 & 6.8 & 1.9 \\ 1.6 & 7.4 & 1.6 & 0.3 & 0.5 \\ 2.1 & 1.2 & 9.3 & 5.2 & 0.2 \\ 4.3 & 3.8 & 6.3 & 1.8 & 2.3 \end{bmatrix} \end{array}$$

Masked multi-head attention

$$\frac{Q_i K_i^T}{\sqrt{d_k}} = \begin{array}{c c} & \begin{array}{ccccc} \text{SOS} & \text{me} & \text{gustan} & \text{los} & \text{gatos} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} 1.3 & \cancel{0.8} & \cancel{1.3} & \cancel{2.8} & \cancel{2.3} \\ 2.4 & 2.8 & \cancel{2.3} & \cancel{6.8} & \cancel{1.9} \\ 1.6 & 7.4 & 1.6 & 0.3 & 0.5 \\ 2.1 & 1.2 & 9.3 & 5.2 & 0.2 \\ 4.3 & 3.8 & 6.3 & 1.8 & 2.3 \end{bmatrix} \end{array}$$

Masked multi-head attention

$$\frac{Q_i K_i^T}{\sqrt{d_k}} = \begin{array}{cc} & \begin{array}{ccccc} \text{SOS} & \text{me} & \text{gustan} & \text{los} & \text{gatos} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} 1.3 & \cancel{0.8} & \cancel{1.3} & \cancel{2.8} & \cancel{2.3} \\ 2.4 & 2.8 & \cancel{2.3} & \cancel{6.8} & \cancel{1.9} \\ 1.6 & 7.4 & 1.6 & \cancel{0.3} & \cancel{0.5} \\ 2.1 & 1.2 & 9.3 & 5.2 & 0.2 \\ 4.3 & 3.8 & 6.3 & 1.8 & 2.3 \end{bmatrix} \end{array}$$

Masked multi-head attention

$$\frac{Q_i K_i^T}{\sqrt{d_k}} = \begin{array}{c c} & \begin{array}{ccccc} \text{SOS} & \text{me} & \text{gustan} & \text{los} & \text{gatos} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} 1.3 & \cancel{0.8} & \cancel{1.3} & \cancel{2.8} & \cancel{2.3} \\ 2.4 & 2.8 & \cancel{2.3} & \cancel{6.8} & \cancel{1.9} \\ 1.6 & 7.4 & 1.6 & \cancel{0.3} & \cancel{0.5} \\ 2.1 & 1.2 & 9.3 & 5.2 & \cancel{0.2} \\ 4.3 & 3.8 & 6.3 & 1.8 & 2.3 \end{bmatrix} \end{array}$$

Masked multi-head attention

$$\frac{Q_i K_i^T}{\sqrt{d_k}} = \begin{array}{c} \\ \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} \begin{array}{c} \begin{array}{ccccc} \text{SOS} & \text{me} & \text{gustan} & \text{los} & \text{gatos} \end{array} \\ \begin{bmatrix} 1.3 & -\infty & -\infty & -\infty & -\infty \\ 2.4 & 2.8 & -\infty & -\infty & -\infty \\ 1.6 & 7.4 & 1.6 & -\infty & -\infty \\ 2.1 & 1.2 & 9.3 & 5.2 & -\infty \\ 4.3 & 3.8 & 6.3 & 1.8 & 2.3 \end{bmatrix} \end{array}$$

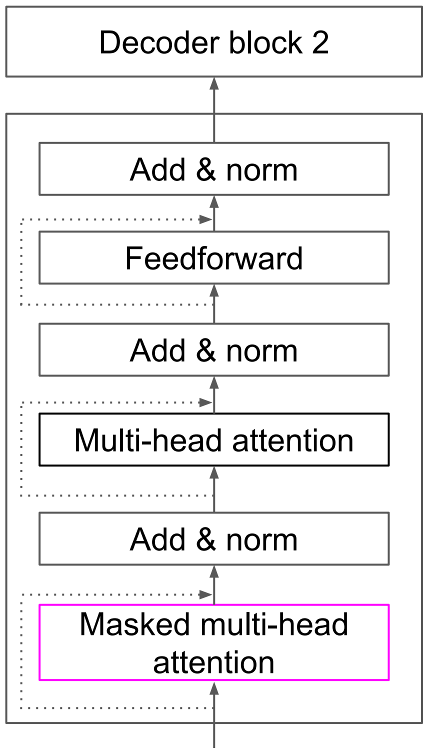
Masked multi-head attention

$$Z_i(Q_i, K_i, V_i) = \mathrm{softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_k}}\right) V_i$$

Masked multi-head attention

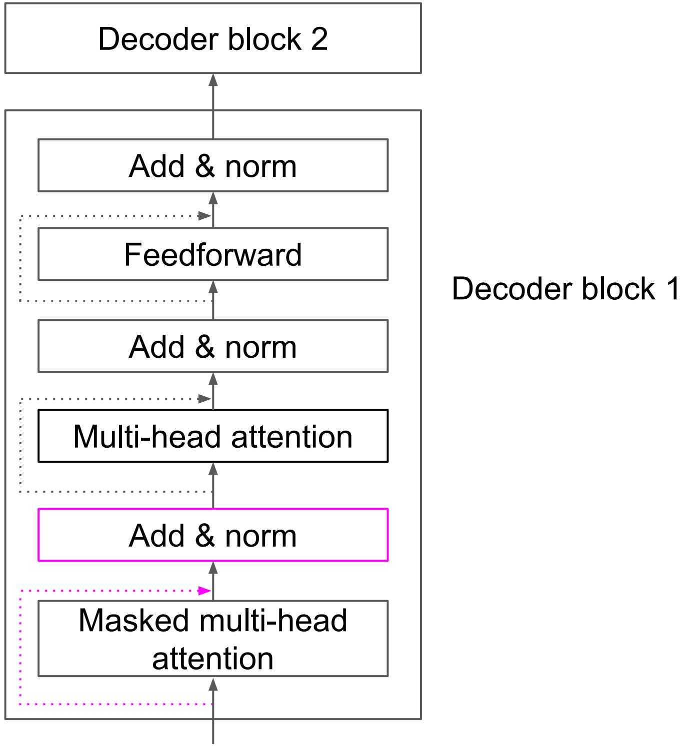
$$Z = concatenate(Z_1, Z_2, Z_3, ..., Z_n)W_0$$

Decoder block

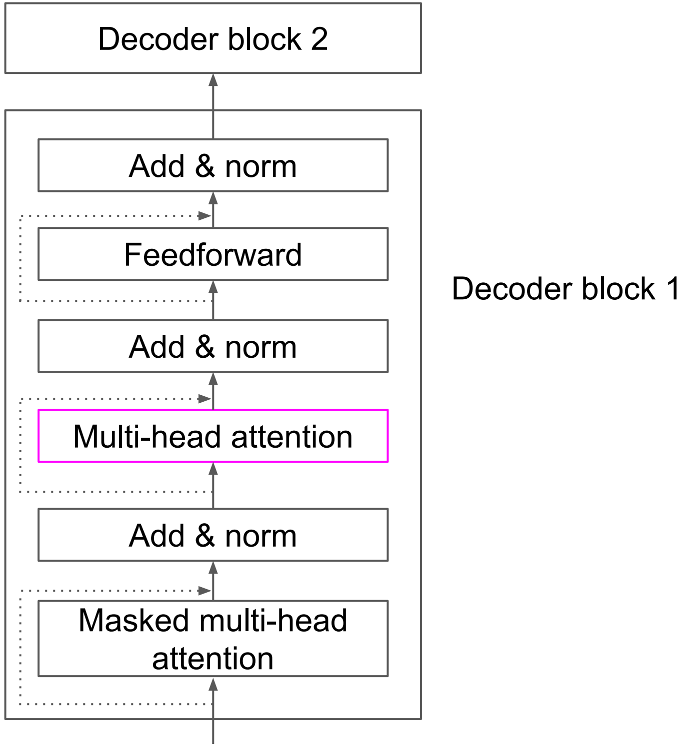


Decoder block 1

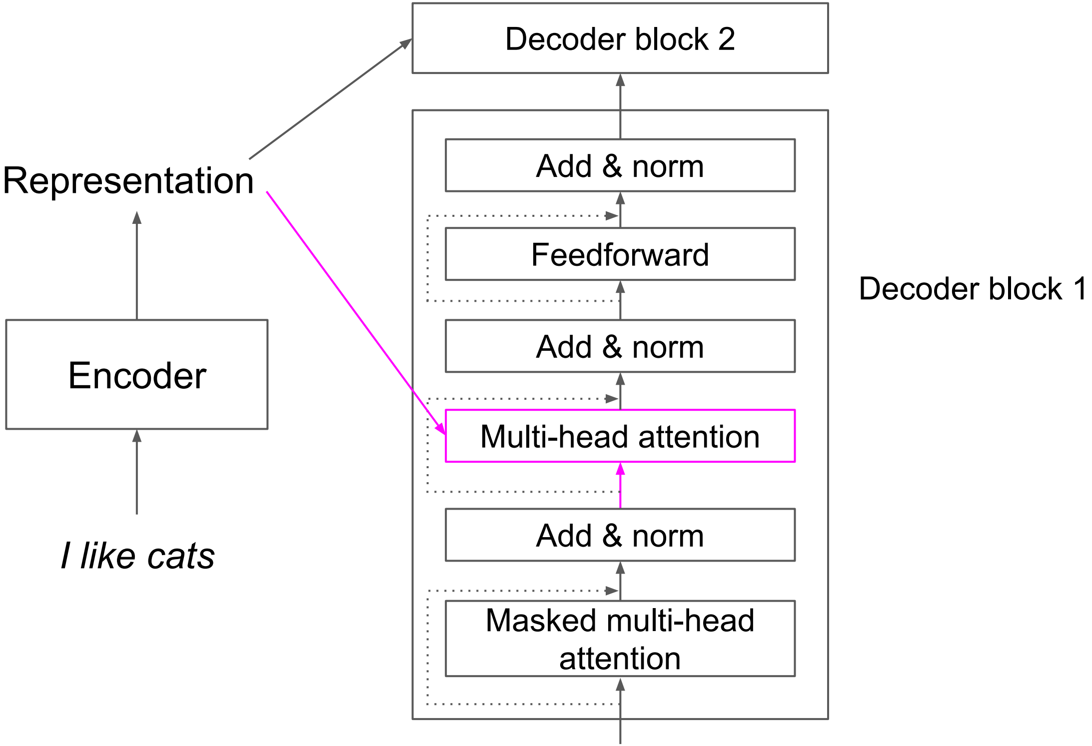
Decoder block



Decoder block



Decoder block



Multi-head attention

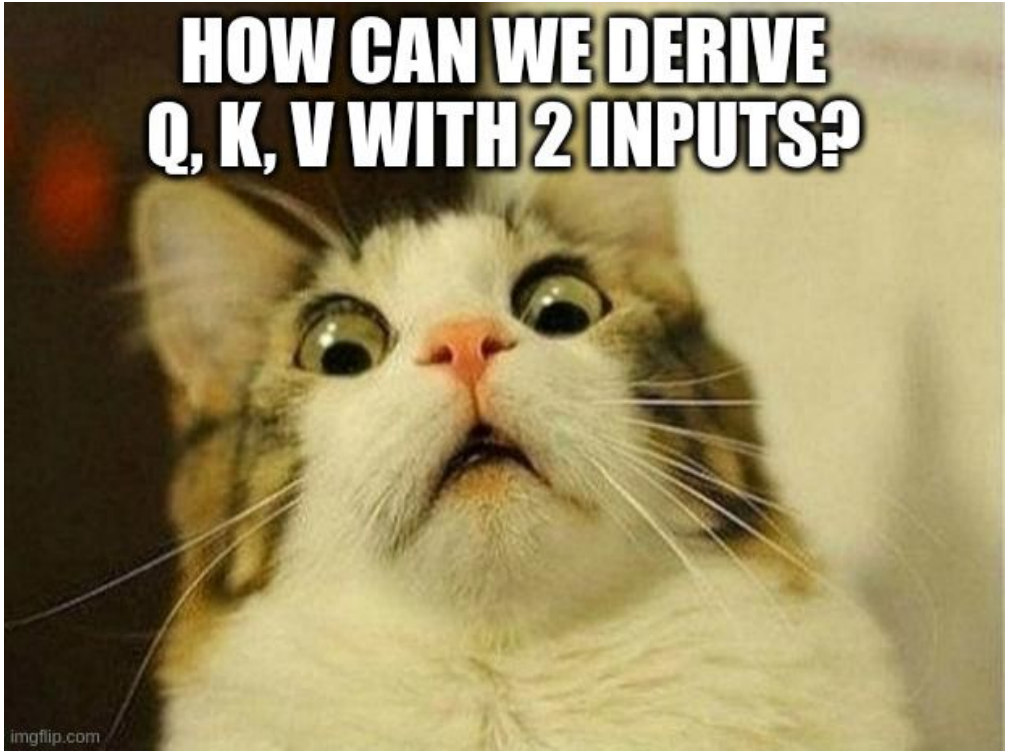
- Inputs: encoder representation R
  + masked attention M

Multi-head attention

- Inputs: encoder representation R
  + masked attention M
- AKA encoder-decoder attention layer

Multi-head attention

- Inputs: encoder representation R + masked attention M
- AKA encoder-decoder attention layer
- Normal multi-head attention layer


HOW CAN WE DERIVE
Q, K, V WITH 2 INPUTS?
imgflip.com

Deriving Q, K, V

- Query matrix (Q) from masked attention input
- Key (K) and value (V) matrices from encoder representation

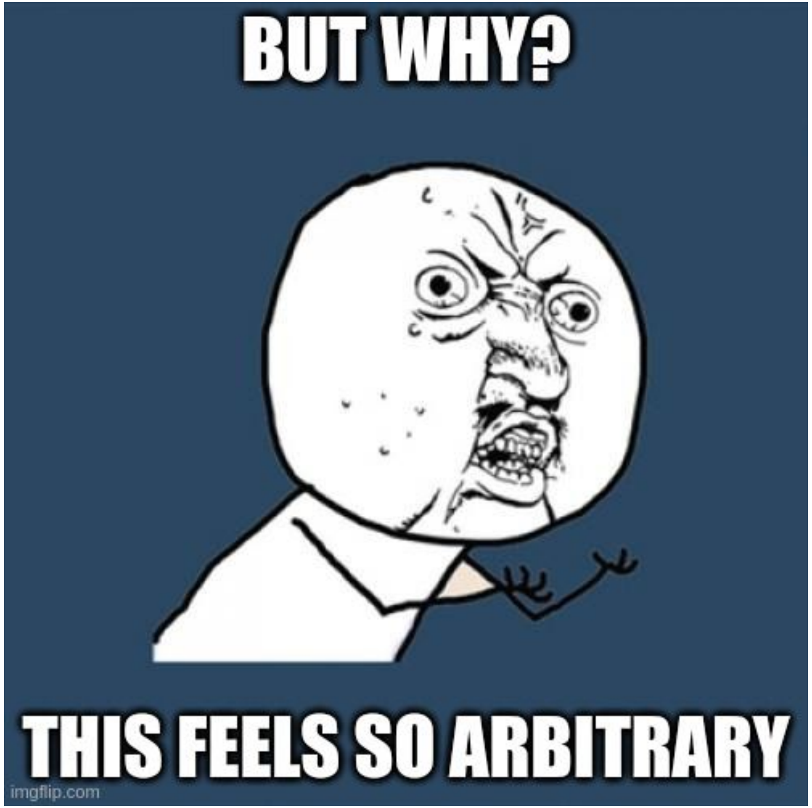
Deriving Q, K, V

- Query matrix (Q) from masked attention input
- Key (K) and value (V) matrices from encoder representation

$$MW_Q = Q$$
$$RW_K = K$$
$$RW_V = V$$

Deriving Q, K, V

- Q holds representation of target sentence
- K, V hold representation of source sentence


BUT WHY?
THIS FEELS SO ARBITRARY
imgflip.com

Deriving attention matrix

$$
Z = \begin{array}{c c}
 & \begin{array}{ccc} \text{I} & \text{like} & \text{cats} \end{array} \\
\begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} &
\begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}
\end{array}
\qquad
\begin{array}{c c}
\begin{array}{c} \text{I} \\ \text{like} \\ \text{cats} \end{array} &
\begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix}
\end{array}
$$

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \qquad V$$

Deriving attention matrix

$$Z = \underbrace{\begin{array}{c|c} & \begin{array}{ccc} \text{I} & \text{like} & \text{cats} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & \boxed{0.8} \end{bmatrix} \end{array}}_{\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)} \quad \underbrace{\begin{array}{c} \text{I} \\ \text{like} \\ \text{cats} \end{array} \begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix}}_{V}$$

Deriving attention matrix

$$Z = \begin{array}{c c} & \begin{array}{ccc} \text{I} & \text{like} & \text{cats} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & 0.8 \end{bmatrix} \end{array} \qquad \begin{array}{ccc} \begin{array}{c} \text{I} \\ \text{like} \\ \text{cats} \end{array} & \begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix} & \begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array} \end{array}$$

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \qquad V$$

Deriving attention matrix

$$Z = \begin{array}{c c} & \begin{array}{ccc} \text{I} & \text{like} & \text{cats} \end{array} \\ \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & 0.8 \end{bmatrix} \end{array} \begin{array}{c c c} \begin{array}{c} \text{I} \\ \text{like} \\ \text{cats} \end{array} & \begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix} & \begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array} \end{array} = \begin{array}{c c} \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} & \begin{bmatrix} \vec{z}_1 \\ \vec{z}_2 \\ \vec{z}_3 \\ \vec{z}_4 \\ \vec{z}_5 \end{bmatrix} \end{array}$$

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \qquad V$$

Deriving attention matrix

$$Z = \underbrace{\begin{array}{c|ccc} & \text{I} & \text{like} & \text{cats} \\ \text{SOS} & & & \\ \text{me} & & & \\ \text{gustan} & & & \\ \text{los} & & & \\ \text{gatos} & & & \end{array}\begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}}_{\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)} \underbrace{\begin{array}{c} \text{I} \\ \text{like} \\ \text{cats} \end{array}\begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix}\begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array}}_{V} = \begin{array}{c} \text{SOS} \\ \text{me} \\ \boxed{\text{gustan}} \\ \text{los} \\ \text{gatos} \end{array}\begin{bmatrix} \vec{z}_1 \\ \vec{z}_2 \\ \boxed{\vec{z}_3} \\ \vec{z}_4 \\ \vec{z}_5 \end{bmatrix}$$

Deriving attention matrix

$$Z = \begin{array}{r} \\ \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} \begin{array}{c} \begin{array}{ccc} \text{I} & \text{like} & \text{cats} \end{array} \\ \begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.1 & 0.8 & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & 0.8 \end{bmatrix} \end{array} \quad \begin{array}{r} \text{I} \\ \text{like} \\ \text{cats} \end{array} \begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix} \begin{array}{l} v_1 \\ v_2 \\ v_3 \end{array} = \begin{array}{r} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} \begin{bmatrix} \vec{z}_1 \\ \vec{z}_2 \\ \vec{z}_3 \\ \vec{z}_4 \\ \vec{z}_5 \end{bmatrix}$$

$$\underbrace{\vec{z}_3}_{\text{gustan}} = 0.1\underbrace{\vec{v}_1}_{\text{I}} + 0.8\underbrace{\vec{v}_2}_{\text{like}} + 0.1\underbrace{\vec{v}_3}_{\text{cats}}$$

Deriving attention matrix

$$Z = \begin{array}{c} \\ \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} \begin{array}{c} \begin{array}{ccc} \text{I} & \text{like} & \text{cats} \end{array} \\ \begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ \boxed{0.1} & 0.8 & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & 0.8 \end{bmatrix} \end{array} \begin{array}{c} \text{I} \\ \text{like} \\ \text{cats} \end{array} \begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix} \begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array} = \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} \begin{bmatrix} \vec{z}_1 \\ \vec{z}_2 \\ \vec{z}_3 \\ \vec{z}_4 \\ \vec{z}_5 \end{bmatrix}$$

$$\underbrace{\vec{z}_3}_{\text{gustan}} = \boxed{0.1}\underbrace{\vec{v}_1}_{\text{I}} + 0.8\underbrace{\vec{v}_2}_{\text{like}} + 0.1\underbrace{\vec{v}_3}_{\text{cats}}$$

Deriving attention matrix

$$
Z = \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array}
\overset{\begin{array}{ccc} \text{I} & \text{like} & \text{cats} \end{array}}{\begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.1 & \boxed{0.8} & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}}
\begin{array}{c} \text{I} \\ \text{like} \\ \text{cats} \end{array}
\begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix}
\begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array}
= \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array}
\begin{bmatrix} \vec{z}_1 \\ \vec{z}_2 \\ \vec{z}_3 \\ \vec{z}_4 \\ \vec{z}_5 \end{bmatrix}
$$

$$
\underset{\text{gustan}}{\vec{z}_3} = 0.1\underset{\text{I}}{\vec{v}_1} + \boxed{0.8}\underset{\text{like}}{\vec{v}_2} + 0.1\underset{\text{cats}}{\vec{v}_3}
$$

Deriving attention matrix

$$Z = \begin{array}{c|ccc} & \text{I} & \text{like} & \text{cats} \\ \hline \text{SOS} & & & \\ \text{me} & & & \\ \text{gustan} & & & \\ \text{los} & & & \\ \text{gatos} & & & \end{array} \begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.1 & 0.8 & \boxed{0.1} \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & 0.8 \end{bmatrix} \begin{array}{c} \text{I} \\ \text{like} \\ \text{cats} \end{array} \begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix} \begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array} = \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} \begin{bmatrix} \vec{z}_1 \\ \vec{z}_2 \\ \vec{z}_3 \\ \vec{z}_4 \\ \vec{z}_5 \end{bmatrix}$$

$$\vec{z}_3 = 0.1\vec{v}_1 + 0.8\vec{v}_2 + \boxed{0.1}\vec{v}_3$$

gustan I like cats

Deriving attention matrix

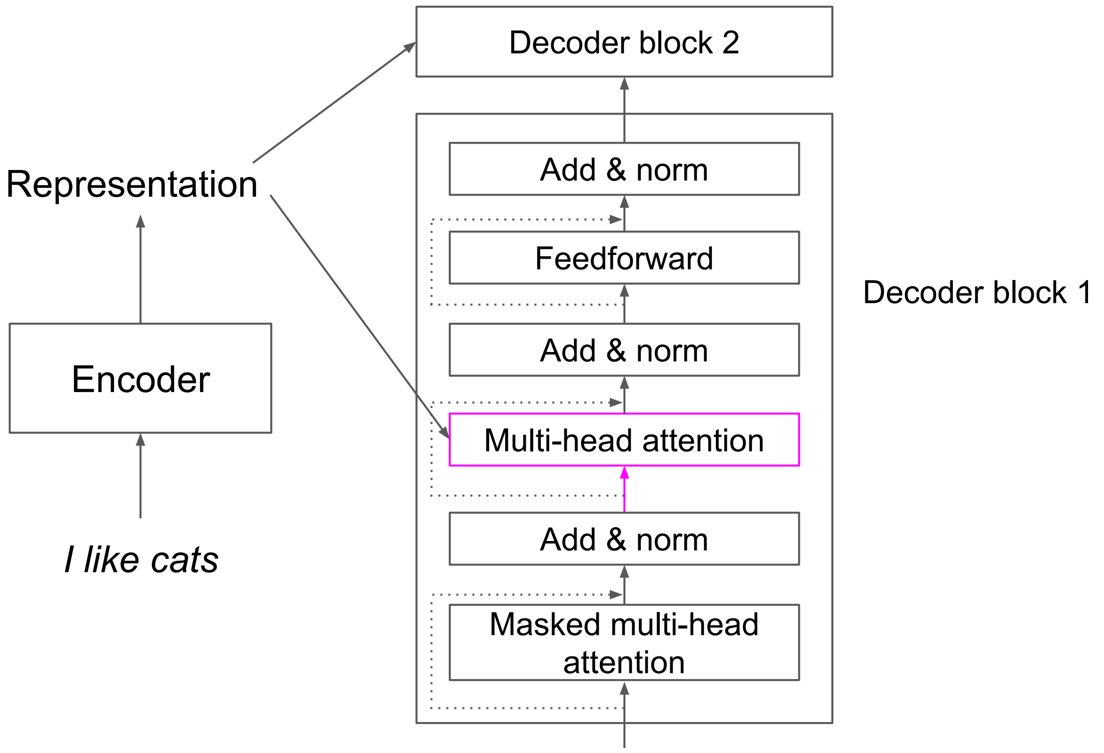
$$Z = \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} \overset{\begin{array}{ccc} \text{I} & \text{like} & \text{cats} \end{array}}{\begin{bmatrix} 0.7 & 0.2 & 0.1 \\ 0.6 & 0.3 & 0.1 \\ 0.1 & \boxed{0.8} & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}} \begin{array}{c} \text{I} \\ \text{like} \\ \text{cats} \end{array} \begin{bmatrix} 0.4 & 1.0 \\ 1.2 & 2.8 \\ 1.7 & 0.2 \end{bmatrix} \begin{array}{c} v_1 \\ v_2 \\ v_3 \end{array} = \begin{array}{c} \text{SOS} \\ \text{me} \\ \text{gustan} \\ \text{los} \\ \text{gatos} \end{array} \begin{bmatrix} \vec{z}_1 \\ \vec{z}_2 \\ \vec{z}_3 \\ \vec{z}_4 \\ \vec{z}_5 \end{bmatrix}$$

$$\underbrace{\vec{z}_3}_{\text{gustan}} = 0.1\underbrace{\vec{v}_1}_{\text{I}} + \boxed{0.8}\underbrace{\vec{v}_2}_{\text{like}} + 0.1\underbrace{\vec{v}_3}_{\text{cats}}$$

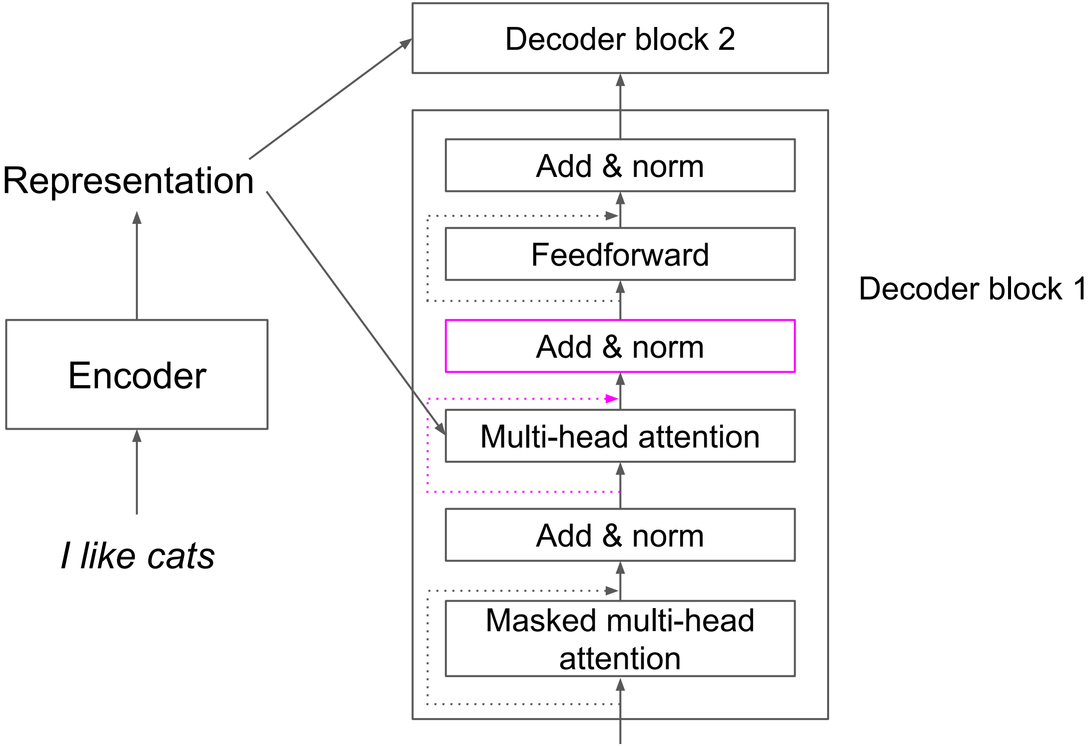
Multi-head attention

$$Z = concatenate(Z_1, Z_2, Z_3, ..., Z_n)W_0$$

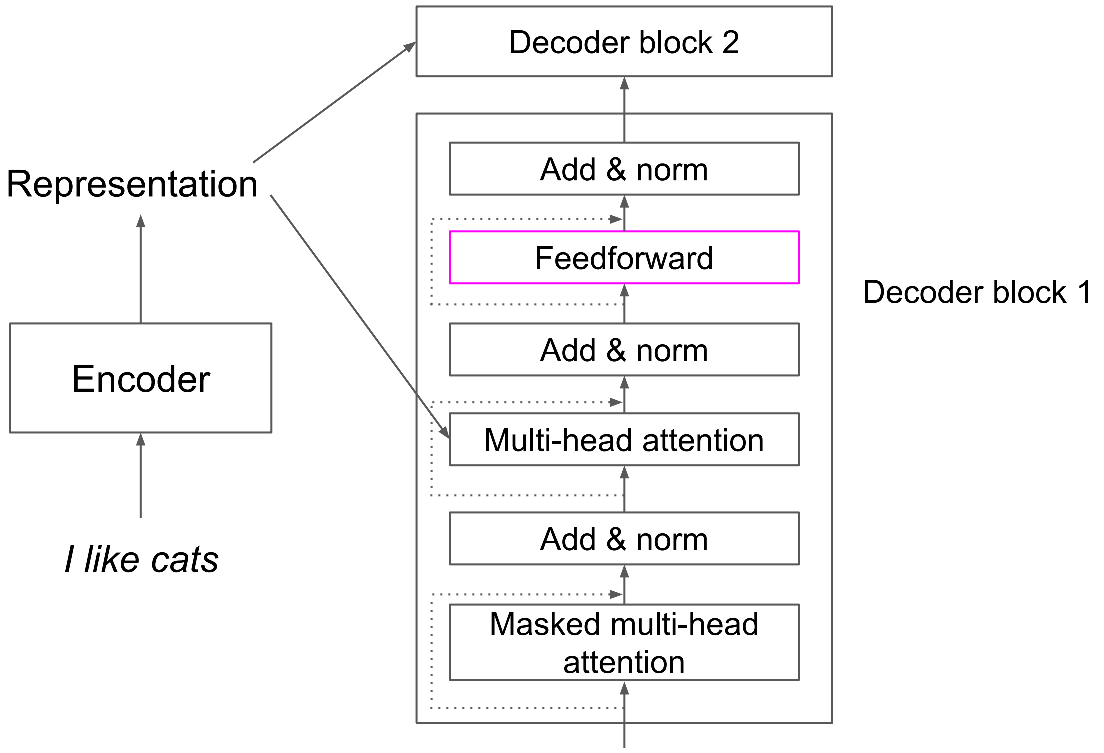
Decoder block



Decoder block



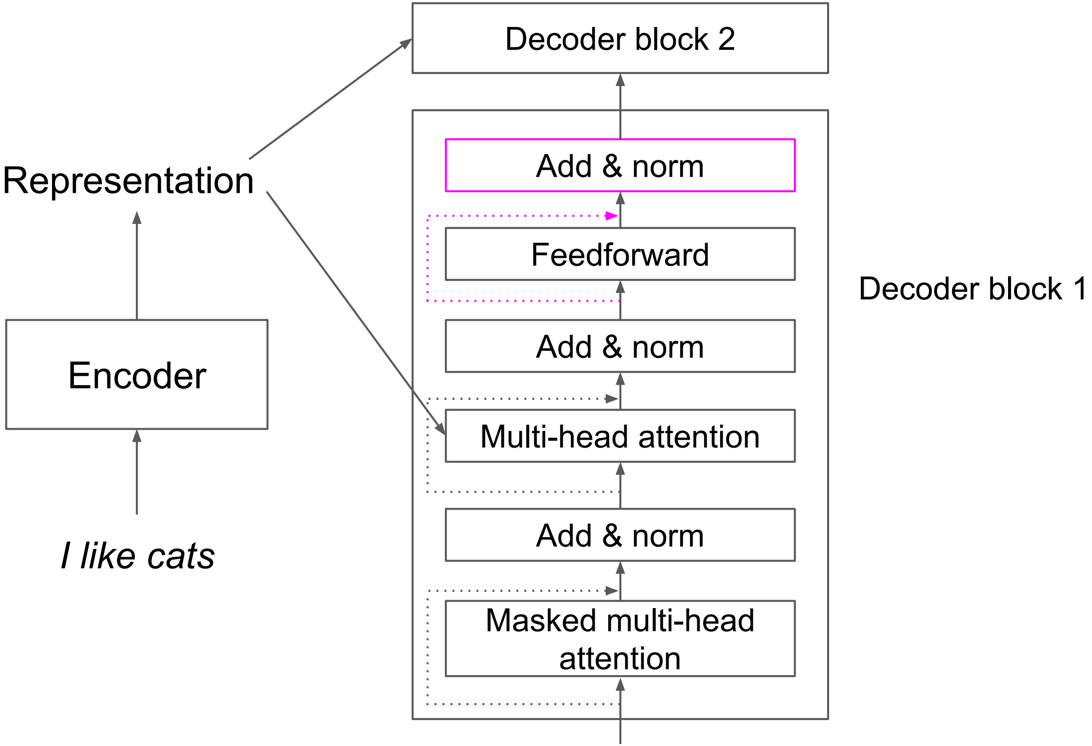
Decoder block



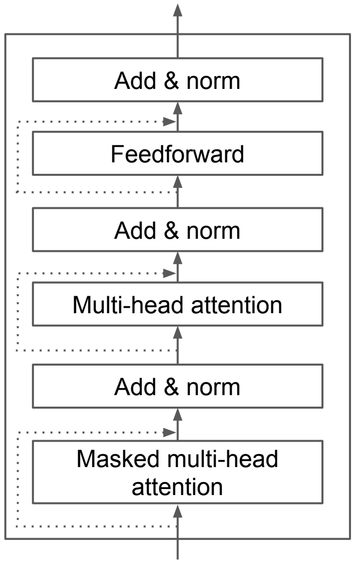
Feedforward layer

- Same as encoder
- 2 fully connected layers
- Process each position data separately
- ReLU activation

Decoder block

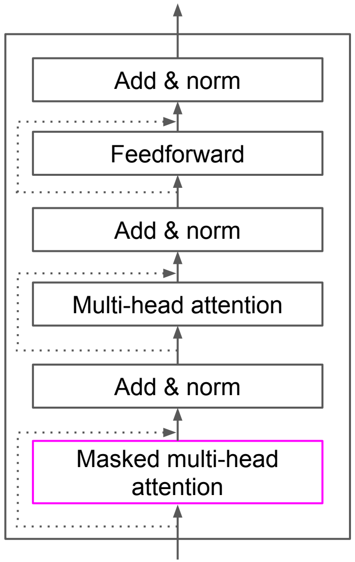


Decoder block: Deeper meaning



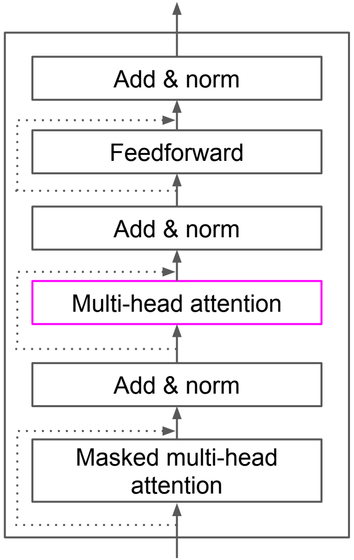
Decoder block: Deeper meaning

- Masked multi-head attention -> autoregressive-aware target sentence context



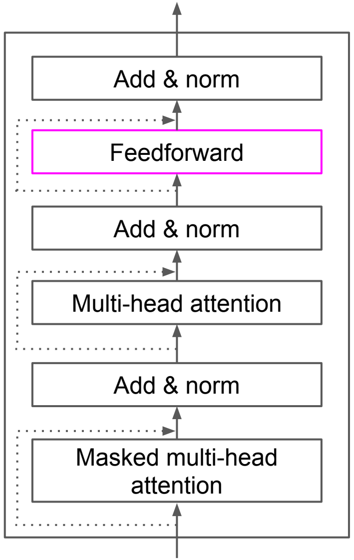
Decoder block: Deeper meaning

- Masked multi-head attention -> autoregressive-aware target sentence context
- Multi-head attention -> context combining target and source



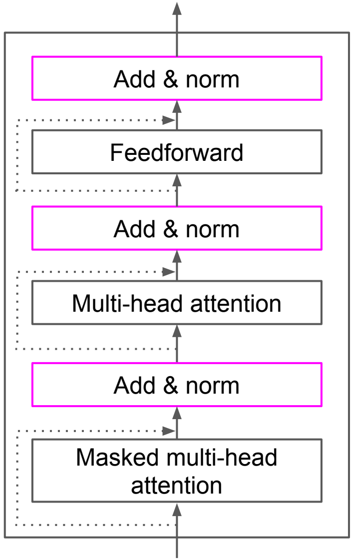
Decoder block: Deeper meaning

- Masked multi-head attention -> autoregressive-aware target sentence context
- Multi-head attention -> context combining target and source
- Feedforward -> nuance

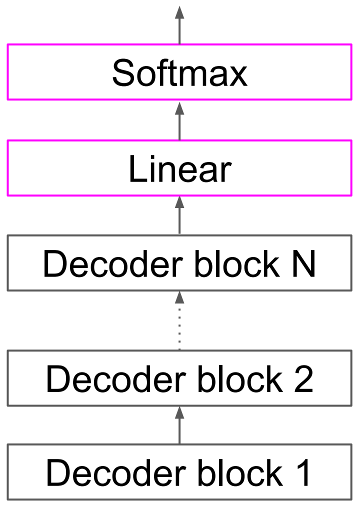


Decoder block: Deeper meaning

- Masked multi-head attention -> autoregressive-aware target sentence context
- Multi-head attention -> context combining target and source
- Feedforward -> nuance
- Add & norm -> streamline learning

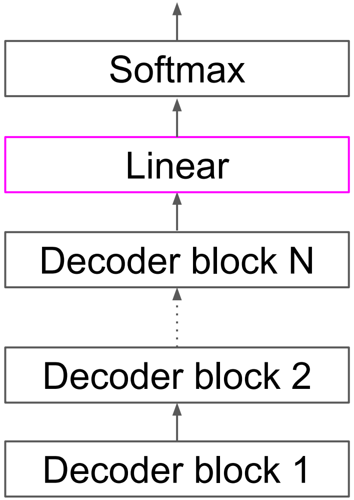


Linear & softmax layers



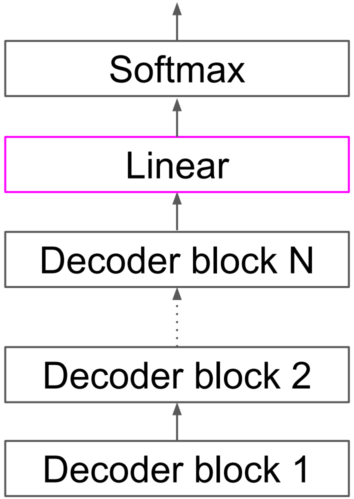
Linear & softmax layers

- Linear layer generates logits



Linear & softmax layers

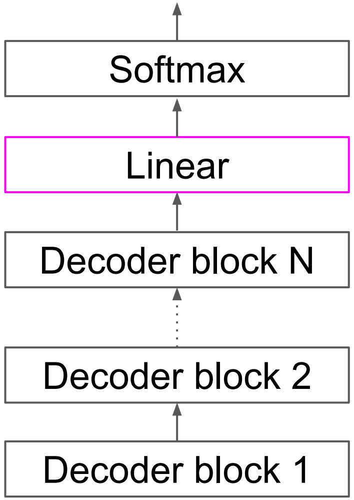
- Linear layer generates logits
- As many logits as size of vocabulary



Linear & softmax layers

- Linear layer generates logits
- As many logits as size of vocabulary

$$\boxed{Vocabulary = [me, gustan, los, gatos]}$$

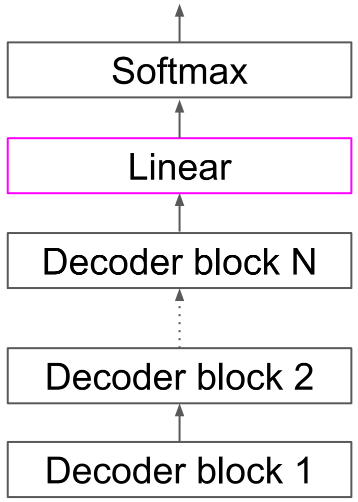


Linear & softmax layers

- Linear layer generates logits
- As many logits as size of vocabulary

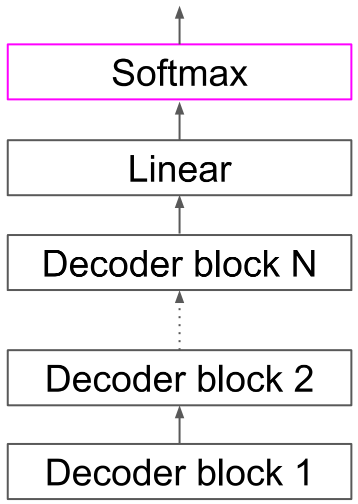
$$Vocabulary = [me, gustan, los, gatos]$$

$$\boxed{Logits = [32, 44, 55, 21]}$$



Linear & softmax layers

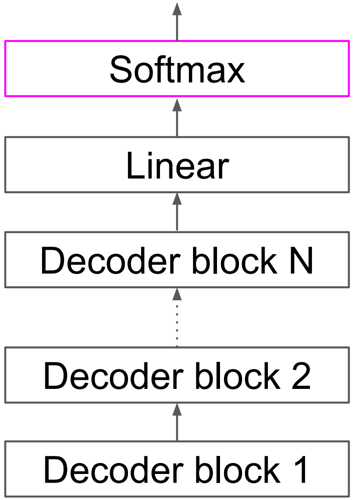
- Linear layer generates logits
- As many logits as size of vocabulary
- Softmax generates distribution probability over the logits



Linear & softmax layers

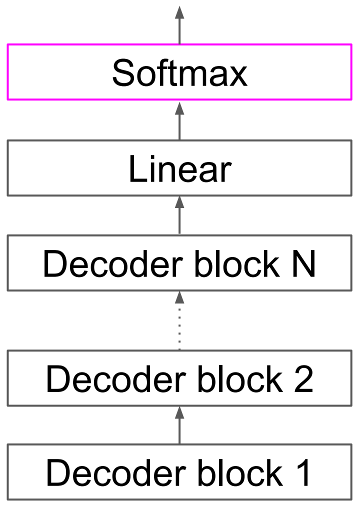
- Linear layer generates logits
- As many logits as size of vocabulary
- Softmax generates distribution probability over the logits

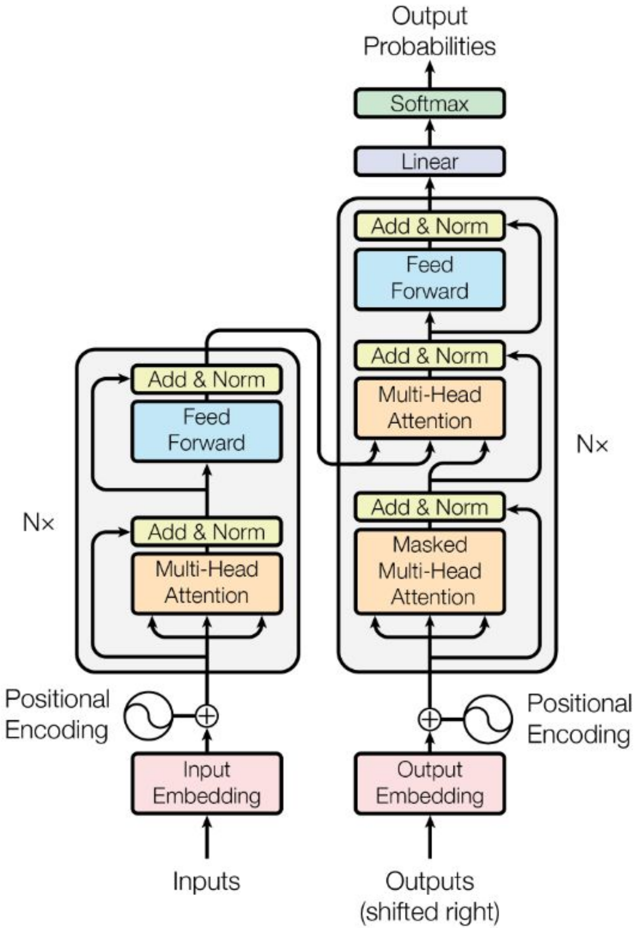
$$\boxed{p = [0.2, 0.1, 0.6, 0.1]}$$

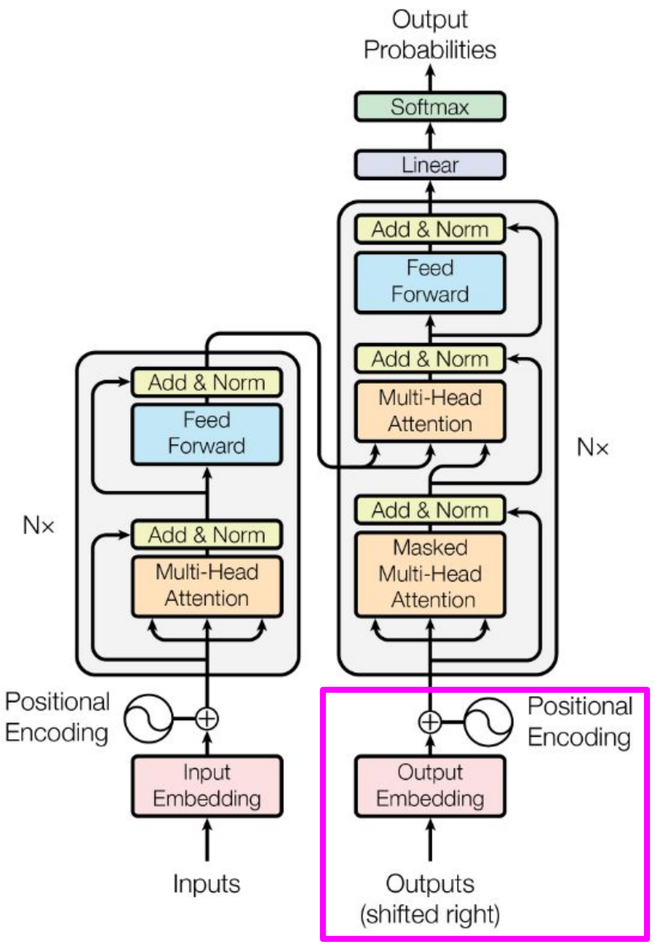


Linear & softmax layers

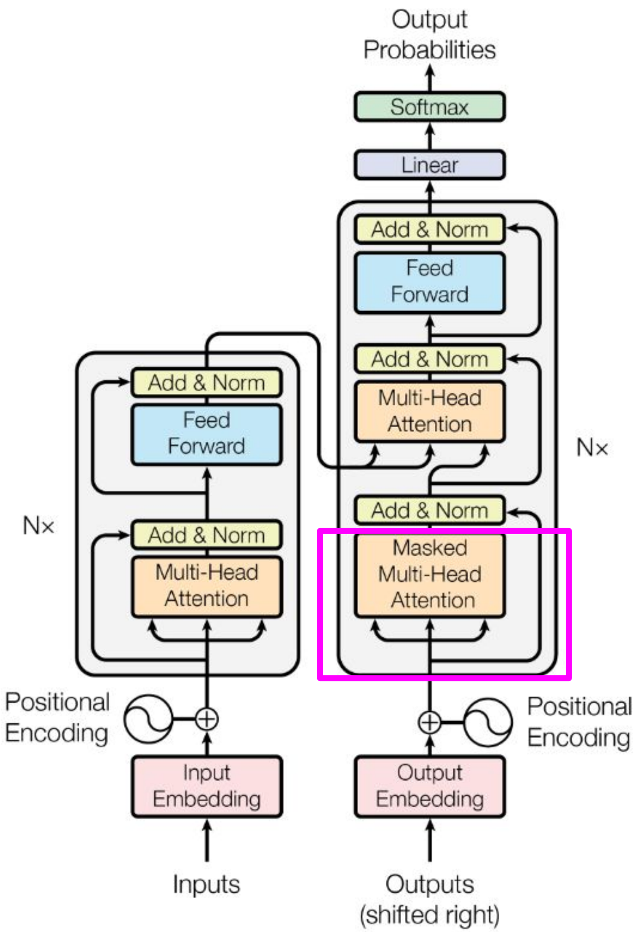
- Linear layer generates logits
- As many logits as size of vocabulary
- Softmax generates distribution probability over the logits
- Select word with highest probability



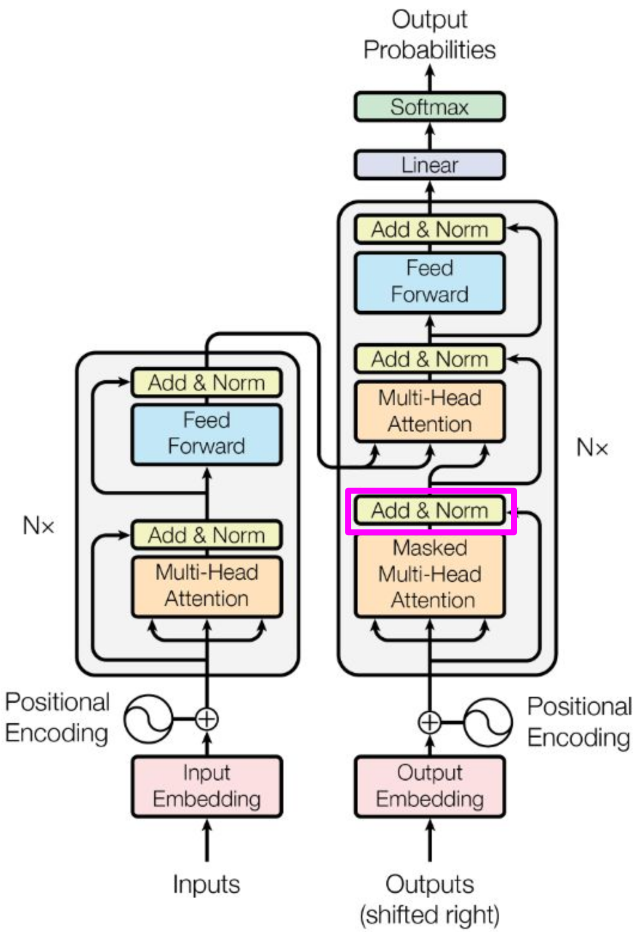

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Add & Norm
Masked
Multi-Head
Attention
Add & Norm
Feed
Forward
Nx
Add & Norm
Multi-Head
Attention
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

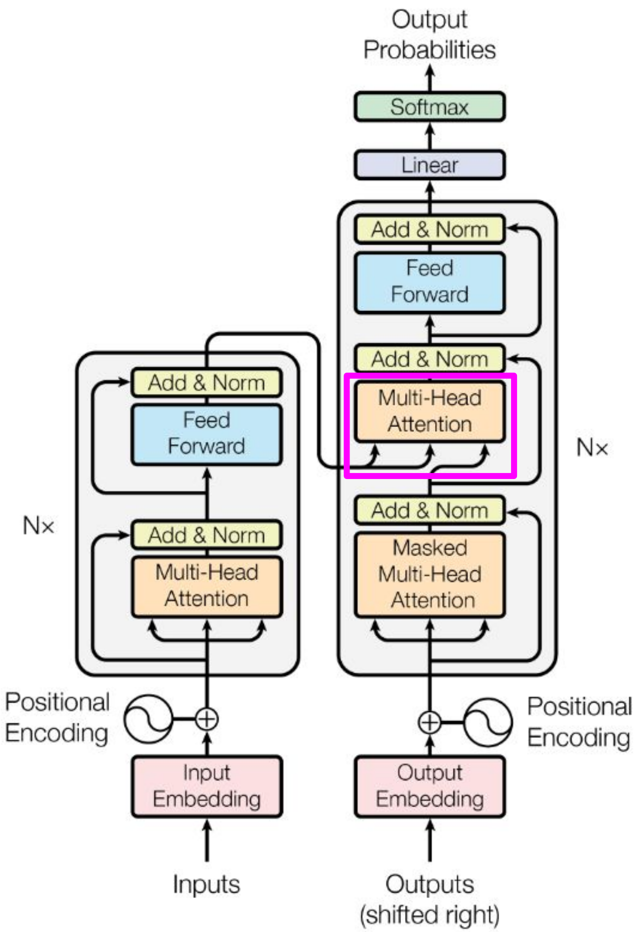


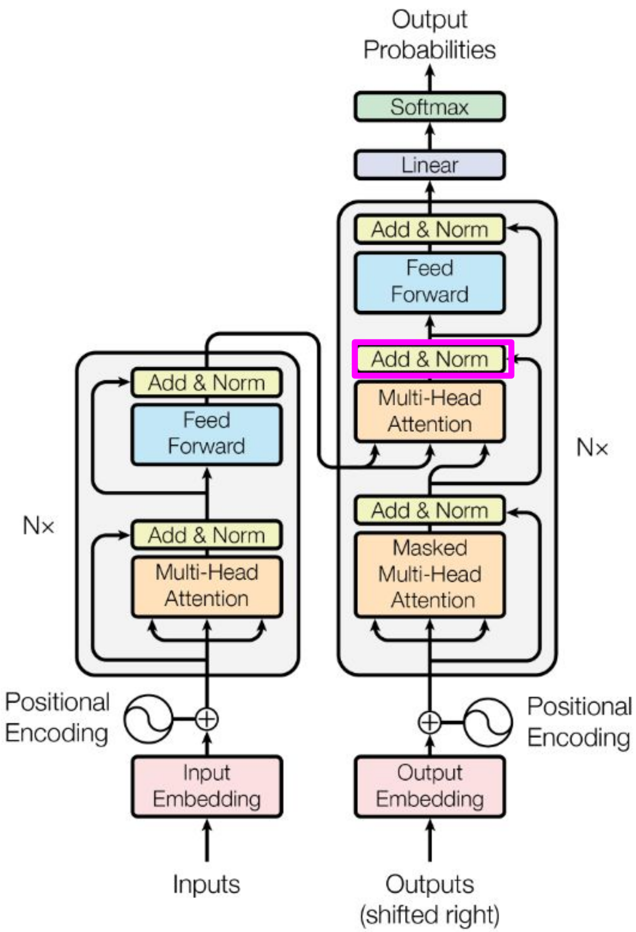
Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Input
Embedding
Inputs
Positional
Encoding
Output
Embedding
Outputs
(shifted right)

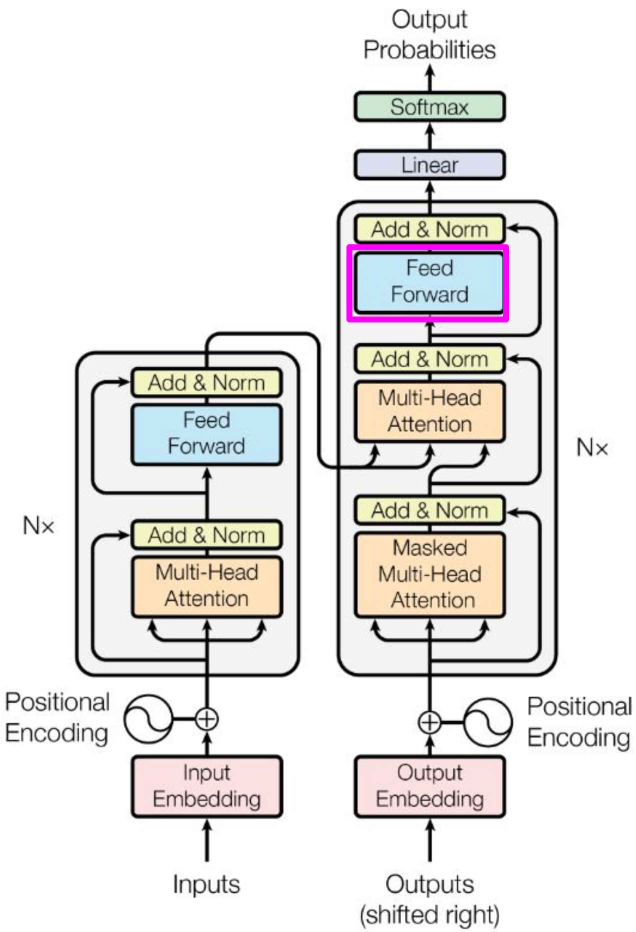


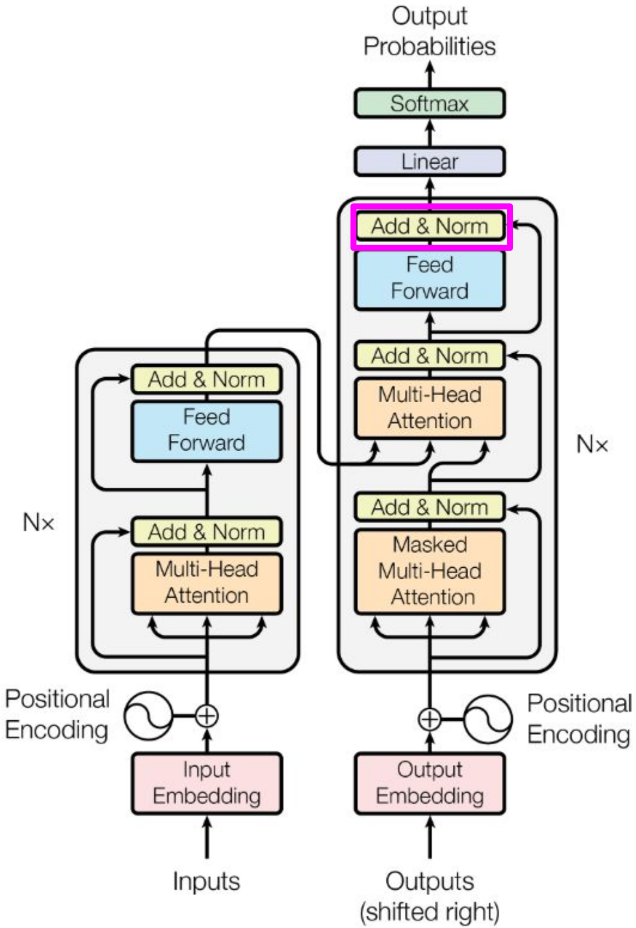
Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

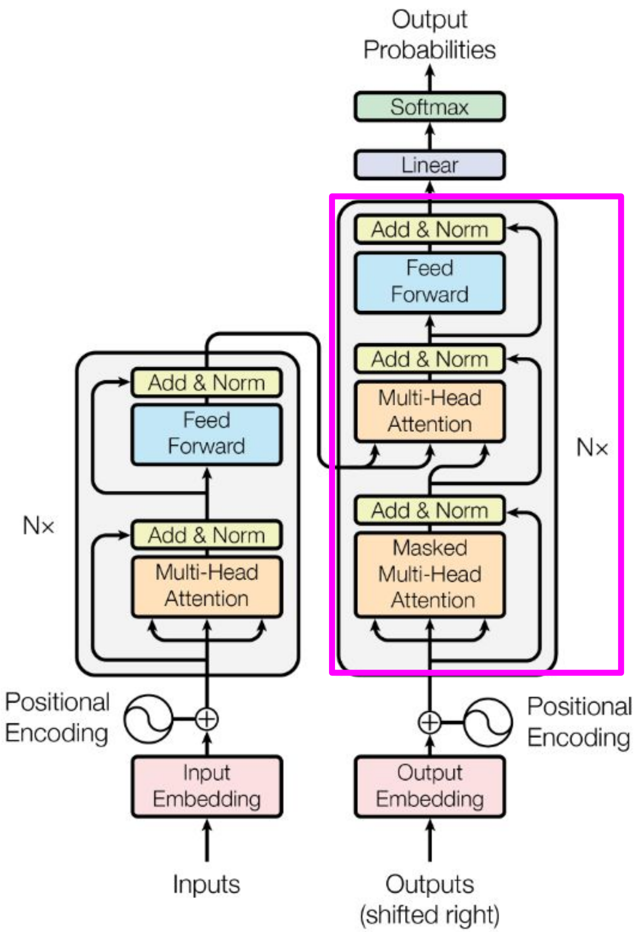

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

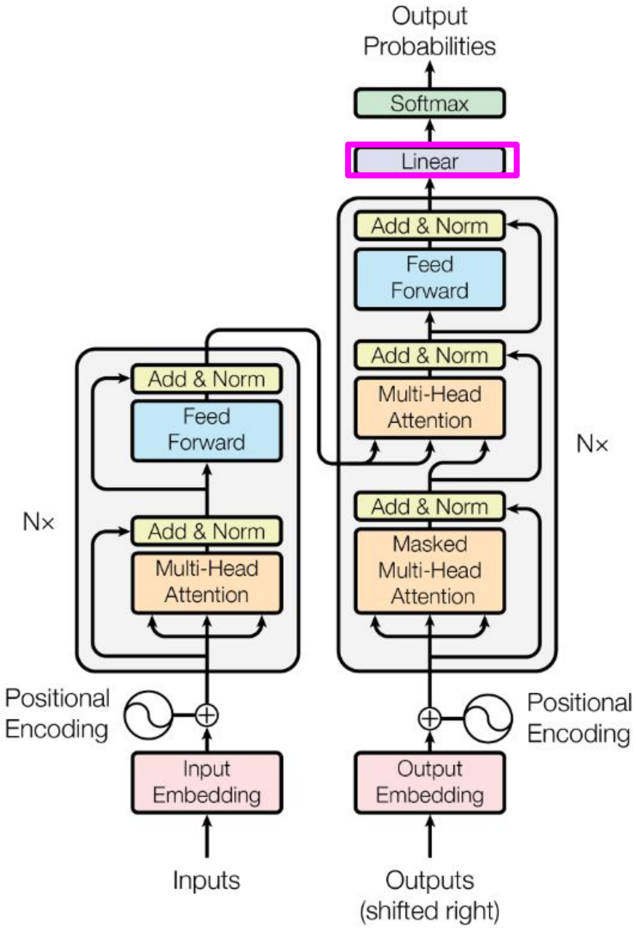

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

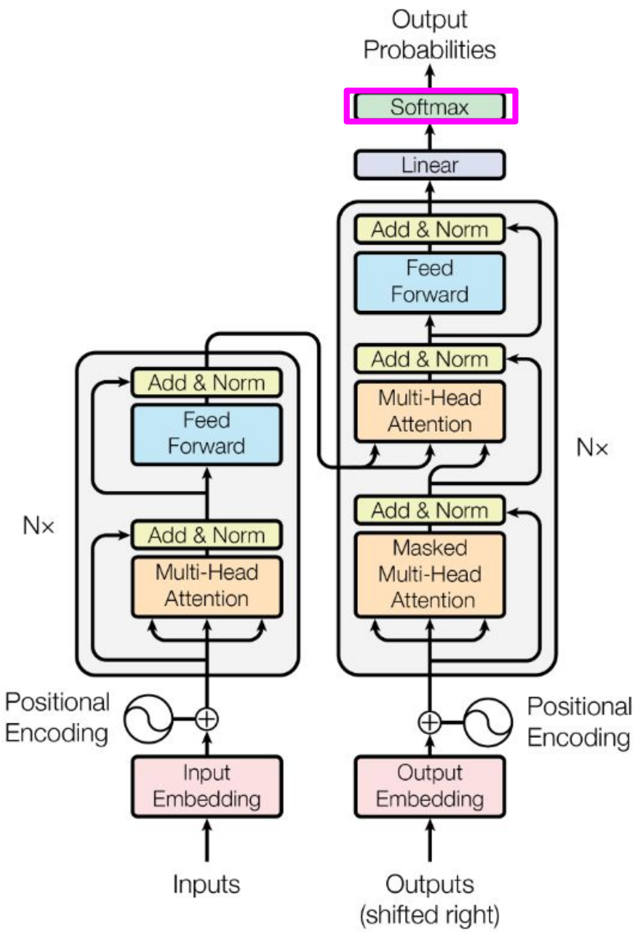

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

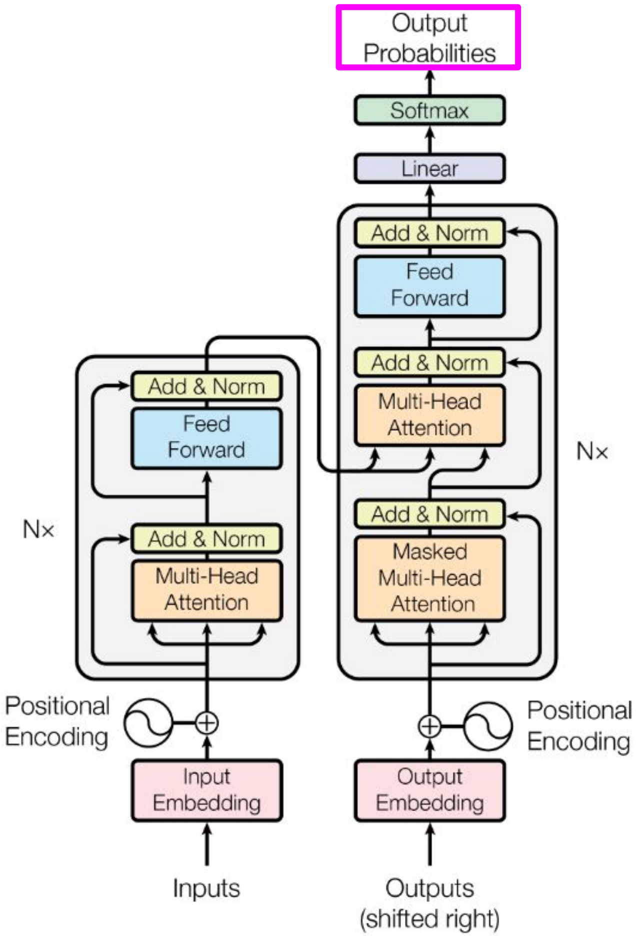


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
N×
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
N×
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)

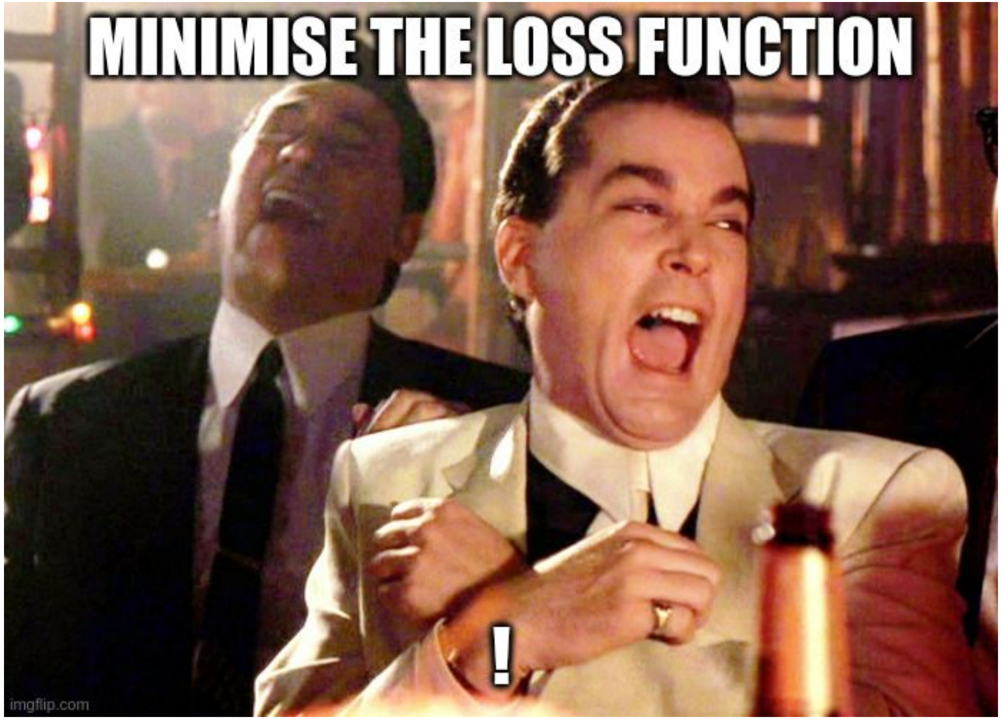

Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)


Output
Probabilities
Softmax
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Add & Norm
Masked
Multi-Head
Attention
Nx
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Positional
Encoding
Positional
Encoding
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)


Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)


YOU'VE MADE IT
CONGRATS!
imgflip.com

Training a transformer


MINIMISE THE LOSS FUNCTION
!
imgflip.com

Training a transformer

- Minimize difference between predicted and expected probability distributions

Training a transformer

- Minimize difference between predicted and expected probability distributions
- Cross-entropy

Training a transformer

- Minimize difference between predicted and expected probability distributions
- Cross-entropy
- Optimize with Adam

Training a transformer

- Minimize difference between predicted and expected probability distributions
- Cross-entropy
- Optimize with Adam
- Dropout to avoid overfitting

Music generation with transformers

- Treat music as a sequence of tokens

Music generation with transformers

- Treat music as a sequence of tokens
- Music representation is key
    - How do you encode pitch?
    - How do you encode time?
    - How do you encode polyphony?

Music generation with transformers

- Treat music as a sequence of tokens
- Music representation is key
  - How do you encode pitch?
  - How do you encode time?
  - How do you encode polyphony?
- Once you have a mapping, run transformer as is

Transformers work for both symbolic and audio generation

My music generation transformer routine

My music generation transformer routine

1. Decide music mapping

My music generation transformer routine

1. Decide music mapping
2. Compile music dataset

My music generation transformer routine

1. Decide music mapping
2. Compile music dataset
3. Choose pre-trained model (BERT, GPT2, LLAMA2)

My music generation transformer routine

1. Decide music mapping
2. Compile music dataset
3. Choose pre-trained model (BERT, GPT2, LLAMA2)
4. Fine-tune model with your data

My music generation transformer routine

1. Decide music mapping
2. Compile music dataset
3. Choose pre-trained model (BERT, GPT2, LLAMA2)
4. Fine-tune model with your data

If the loop above doesn't work:

- improve data
- create custom architecture

Music data is key

- Data >> architecture

Music data is key

- Data >> architecture
- Quality >> quantity

Music data is key

- Data >> architecture
- Quality >> quantity
- Focus on small (e.g., 10K melodies), but consistent dataset (e.g., 1 sub-genre)

Pros and cons of transformers

+

- Capture phrase-level dependencies
- Flexible
- Conditioning on text, chords, …
- Style transfer

Pros and cons of transformers

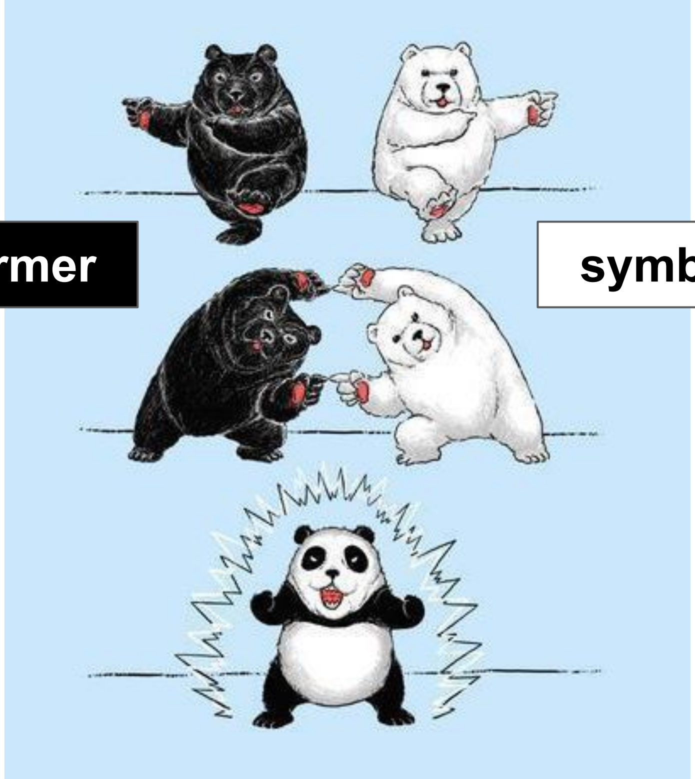
+

- Capture phrase-level dependencies
- Flexible
- Conditioning on text, chords, …
- Style transfer

-

- Long-term dependencies
- Massive computation
- Massive datasets
- Black box
- Copyright issues

Most promising research



transformer

symbolic AI

Most promising research

Music representation with rich music theory information

Key takeaways

- Decoder generates output word by word

Key takeaways

- Decoder generates output word by word
- Decoder block is made up of masked multi-head attention, multi-head attention, and feedforward

Key takeaways

- Decoder generates output word by word
- Decoder block is made up of masked multi-head attention, multi-head attention, and feedforward
- Maksed multi-head attention solves information leakage

Key takeaways

- Decoder generates output word by word
- Decoder block is made up of masked multi-head attention, multi-head attention, and feedforward
- Maksed multi-head attention solves information leakage
- Train transformer minimizing difference between predicted and expected probabilities

Key takeaways

- Decoder generates output word by word
- Decoder block is made up of masked multi-head attention, multi-head attention, and feedforward
- Maksed multi-head attention solves information leakage
- Train transformer minimizing difference between predicted and expected probabilities
- Treat music as a sequence of tokens

Key takeaways

- Decoder generates output word by word
- Decoder block is made up of masked multi-head attention, multi-head attention, and feedforward
- Maksed multi-head attention solves information leakage
- Train transformer minimizing difference between predicted and expected probabilities
- Treat music as a sequence of tokens
- Music data is key for generation

Key takeaways

- Decoder generates output word by word
- Decoder block is made up of masked multi-head attention, multi-head attention, and feedforward
- Maksed multi-head attention solves information leakage
- Train transformer minimizing difference between predicted and expected probabilities
- Treat music as a sequence of tokens
- Music data is key for generation
- Transformer captures phrase-level dependencies, but struggles with longer structures

Key takeaways

- Decoder generates output word by word
- Decoder block is made up of masked multi-head attention, multi-head attention, and feedforward
- Maksed multi-head attention solves information leakage
- Train transformer minimizing difference between predicted and expected probabilities
- Treat music as a sequence of tokens
- Music data is key for generation
- Transformer captures phrase-level dependencies, but struggles with longer structures
- Combine transformer + symbolic AI with richer music representation

What’s up next?

Chord progression generation with transformer