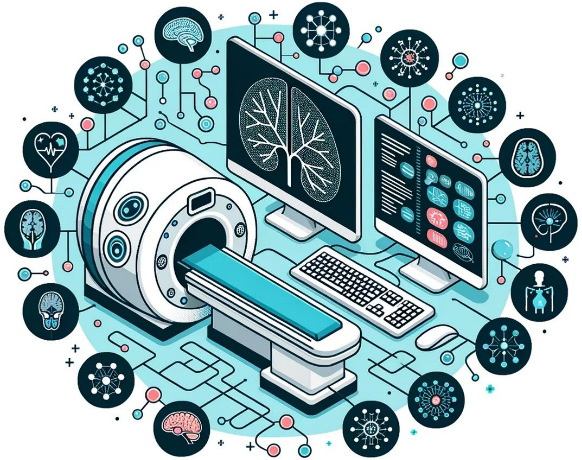
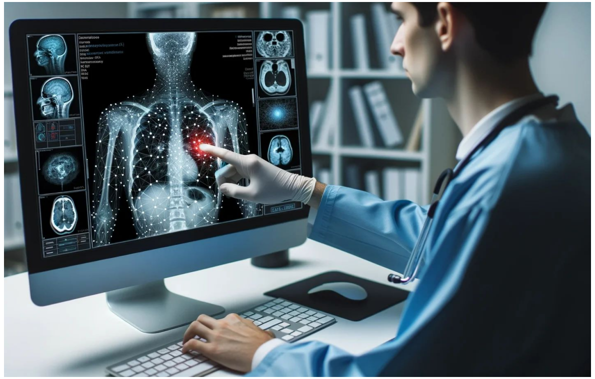
# Applied Deep Learning and Generative Models in Healthcare



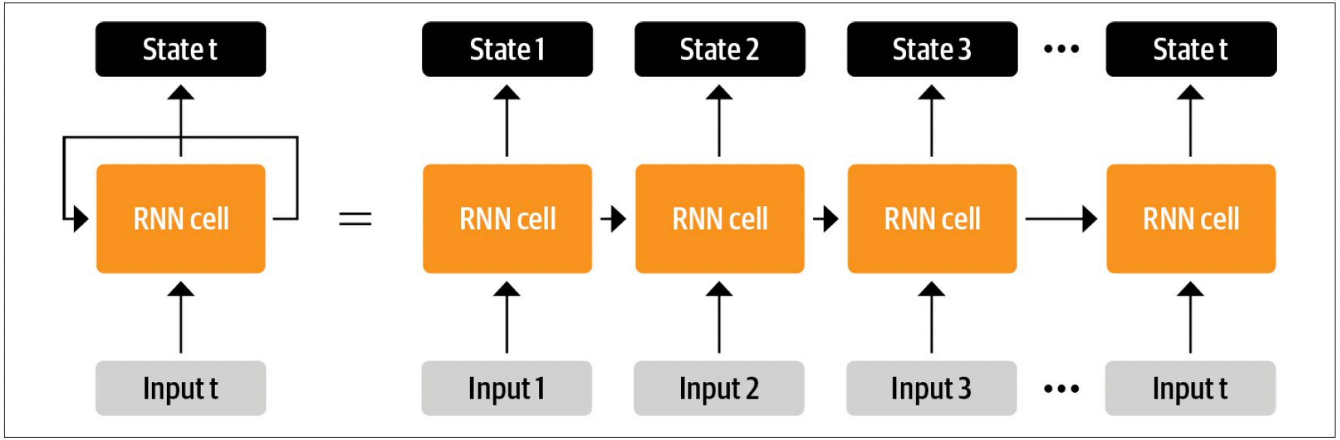


**Session 5:** Transformers
**Date:** Feb 08 2025

**Instructor:** Mahmoud E. Khani, Ph.D.

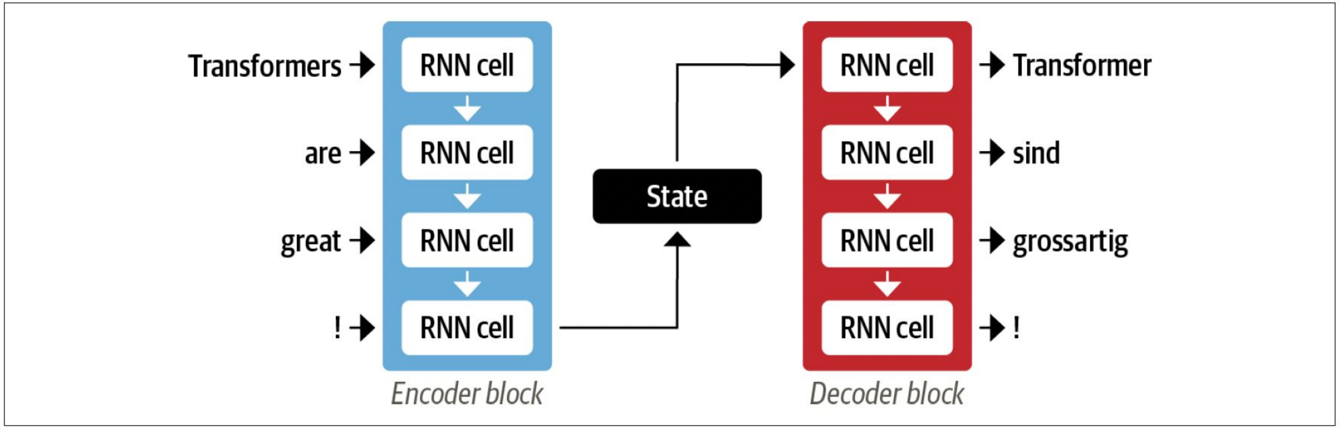
# Recap: Key Highlights from February 1, 2025

- **RNNs** showed unprecedented results in NLP applications.



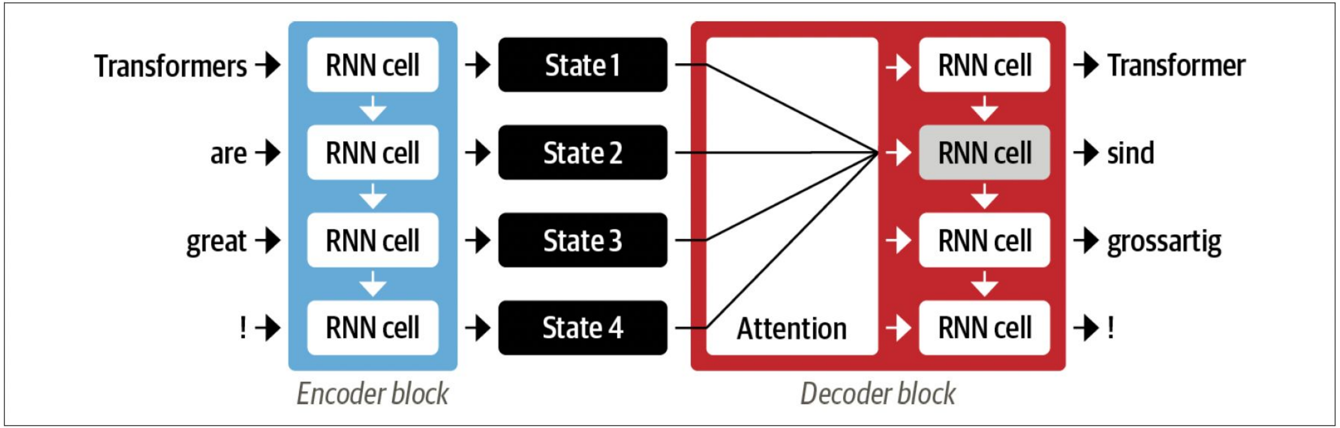
# Recap: Key Highlights from February 1, 2025

- An encoder-decoder architecture with a pair of RNNs was introduced for seq2seq application.



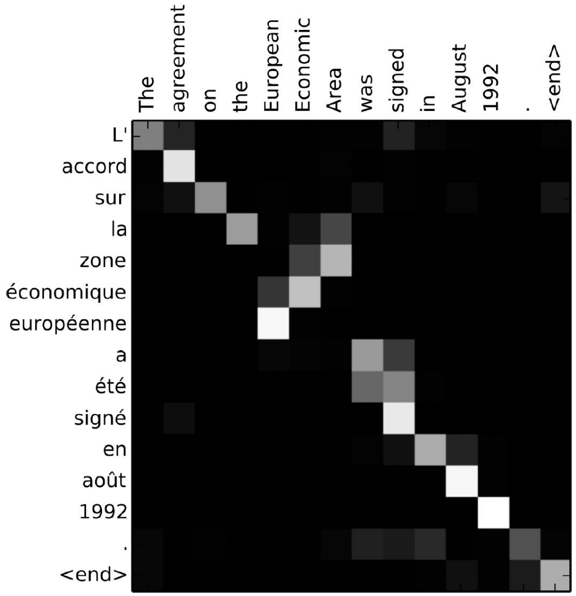
# Recap: Key Highlights from February 1, 2025

- Attention mechanism solved the bottleneck problems in RNNs.



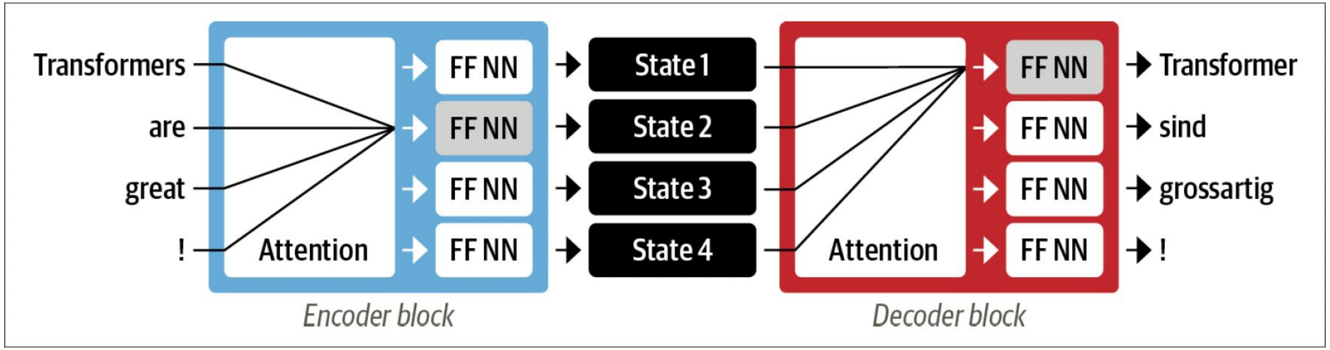
# Recap: Key Highlights from February 1, 2025

- Attention mechanism solved the bottleneck problems in RNNs.



# Recap: Key Highlights from February 1, 2025

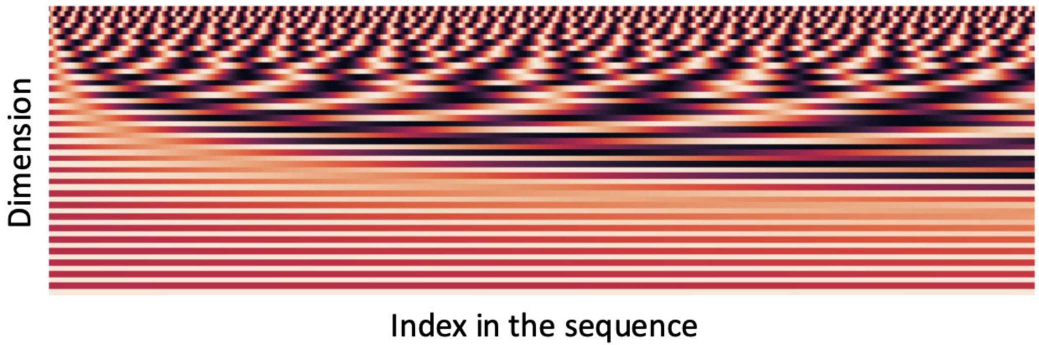
- **People soon realized Attention is all you need!**



# Sinusoidal position representation to add order

- **Sinusoidal position representations: concatenate sinusoidal functions of varying periods**
  - Periodicity indicates that maybe “absolute position” isn’t as important
  - can extrapolate to longer sequences as periods restart!

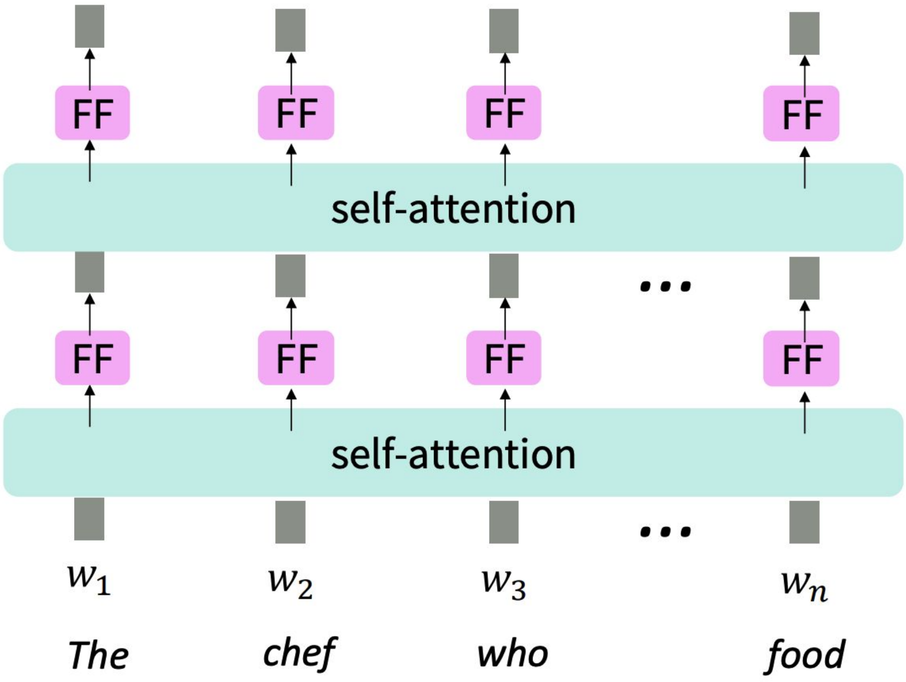
$$\boldsymbol{p}_i = \begin{pmatrix} \sin(i/10000^{2*1/d}) \\ \cos(i/10000^{2*1/d}) \\ \vdots \\ \sin(i/10000^{2*\frac{d}{2}/d}) \\ \cos(i/10000^{2*\frac{d}{2}/d}) \end{pmatrix}$$



# Adding nonlinearity to self-attention

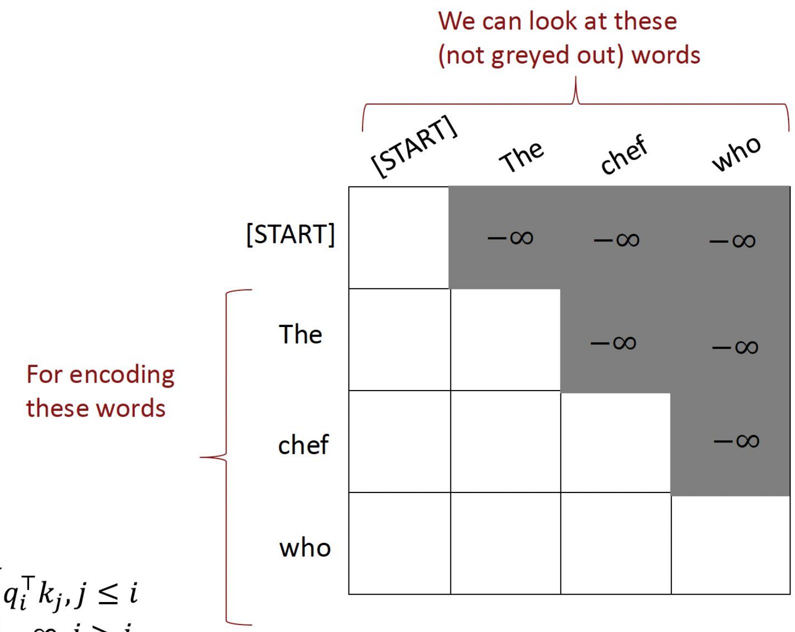
- Easy fix: add a **feed-forward network** to post-process each output vector.

$$m_i = MLP(\text{output}_i)$$
$$= W_2 * \text{ReLU}(W_1 \text{ output}_i + b_1) + b_2$$



# Masking the future in self-attention

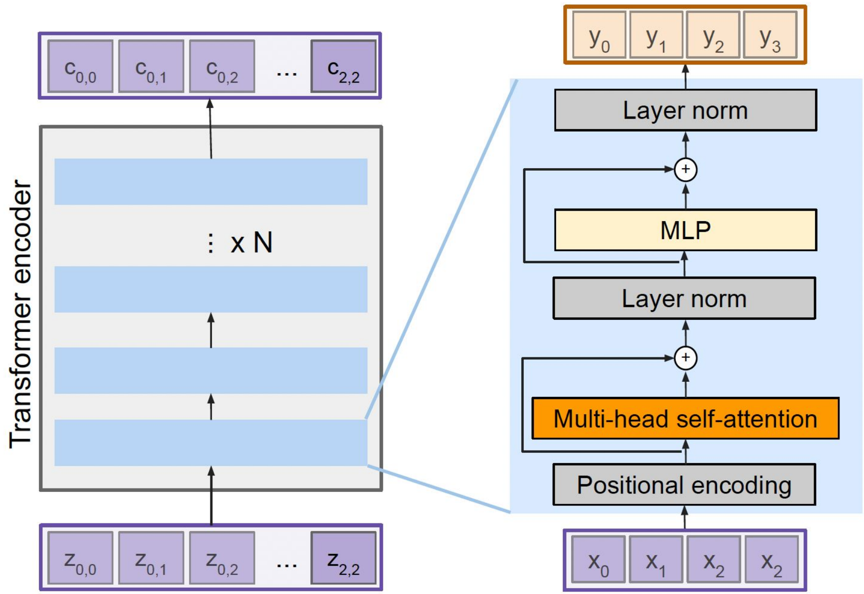
- To use self-attention in **decoders**, we need to ensure not to peek at the future.
- At each timestep, we could change the set of **keys and queries** to only include past words!




$$\begin{cases} q_i^\top k_j, j \leq i \\ -\infty, j > i \end{cases}$$

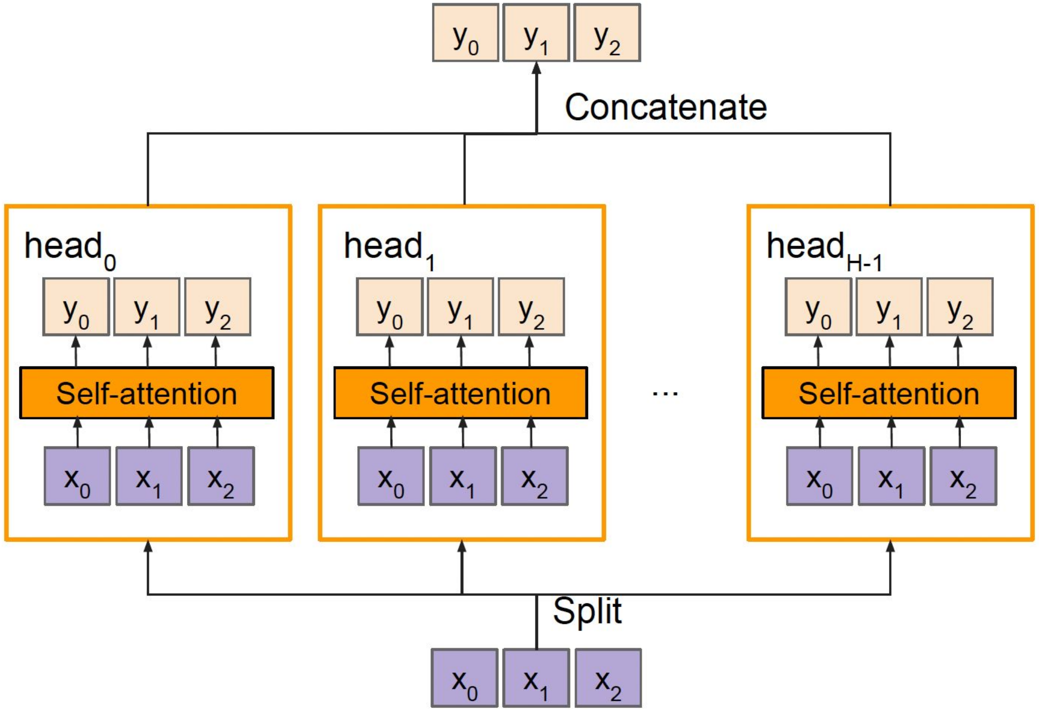
# Encoder: Put everything together

- **Position representation:**
  - Specify the sequence order, since self-attentiis an unordered function of its inputs.
- **Nonlinearities**
  - Frequently implemented as a simple feedforward network.
- **Masking**
  - Keep information about the future from “leaking” to the past.



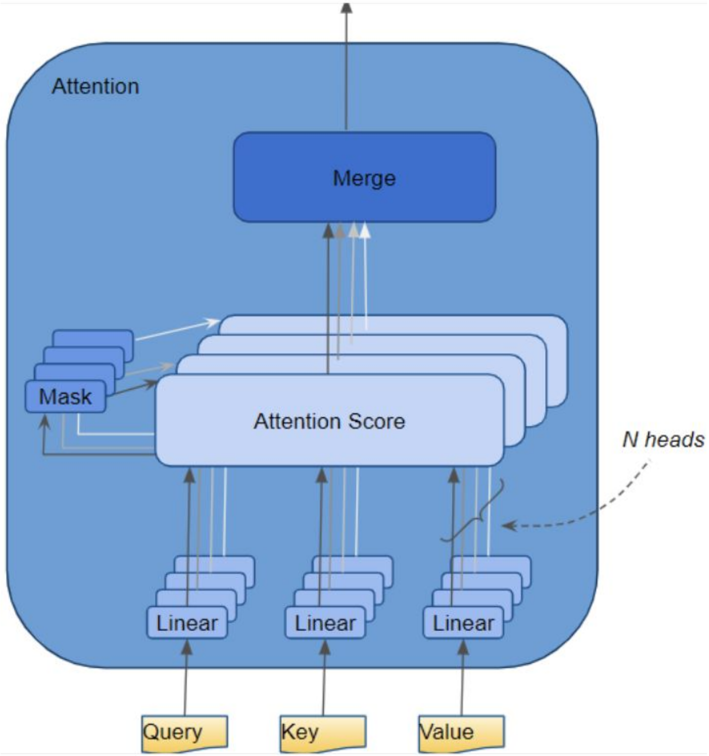
# Multi-head self attention layer

- Multiple self-attention heads in parallel



# Multi-head self attention layer

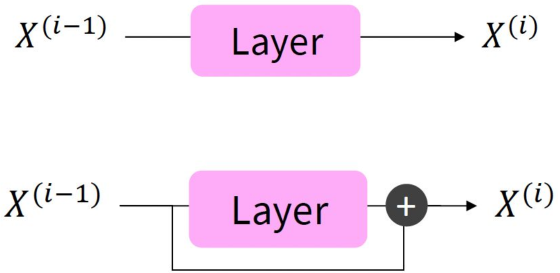
- The Attention module splits its Query, Key, and Value parameters N-ways and passes each split independently through a separate Head. All of these similar Attention calculations are then combined together to produce a final Attention score. This is called Multi-head attention and gives the Transformer greater power to encode multiple relationships and nuances for each word.

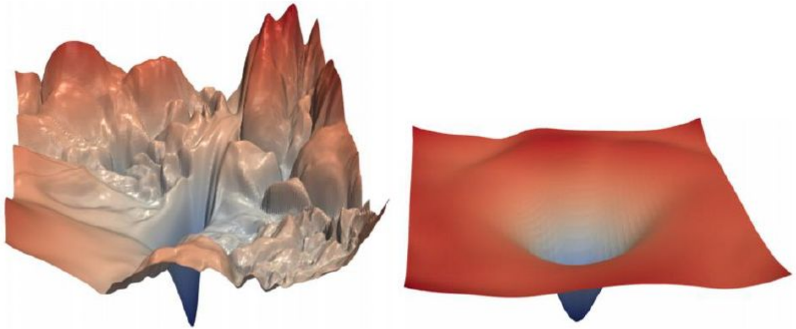



https://towardsdatascience.com/transformers-explained-visually-part-3-multi-head-attention-deep-dive-1c1ff1024853/

# Residual connection

- A trick to help models learn better!
- Gradient is 1 through residual connection
- Bias toward identity function.



[no residuals] [residuals]

[Loss landscape visualization,
Li et al., 2018, on a ResNet]

# Layer normalization

- A trick to help models train faster.
- Cut down on uninformative variation in hidden vectors by normalizing to unit mean and standard deviation within each layer

$$\text{output} = \frac{x - \mu}{\sqrt{\sigma} + \epsilon} * \gamma + \beta$$

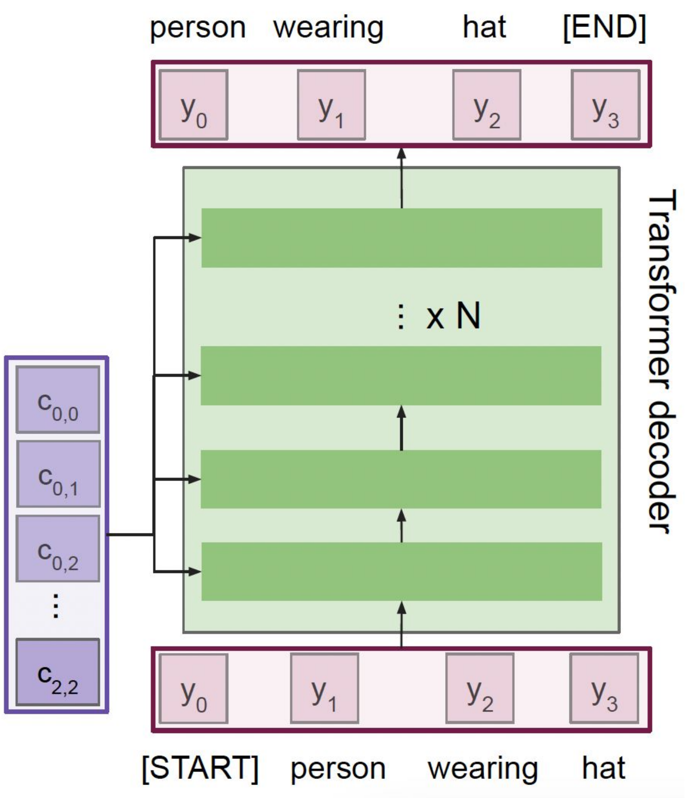
Normalize by scalar mean and variance

Modulate by learned elementwise gain and bias

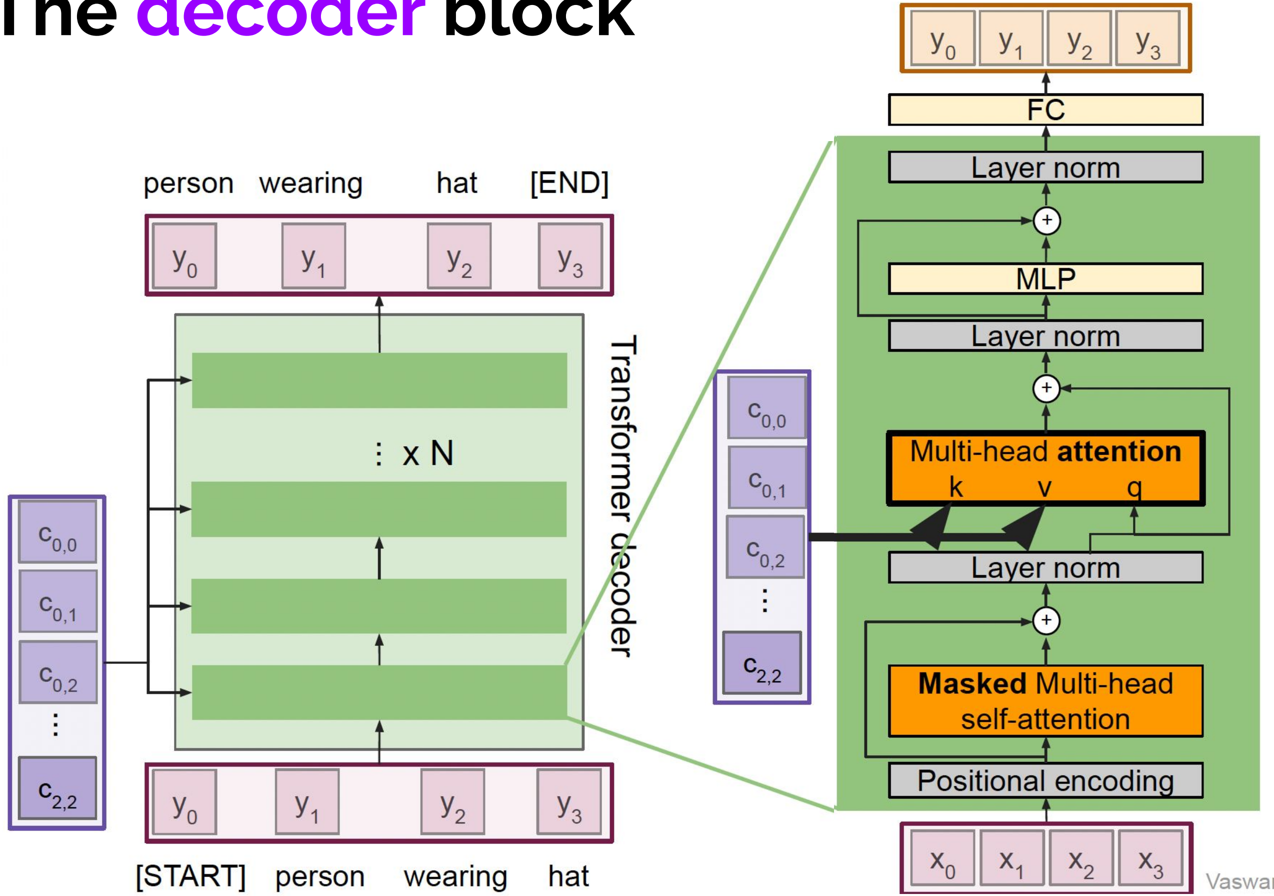

https://youtu.be/wjZofJX0v4M?si=1n1d-Y39sjElHpjC

# The decoder block

- **Made of N decoder blocks**
- **In Vaswani et al.**
  - N = 6
  - D = 512



# The decoder block




Multi-head attention block attends over the transformer encoder outputs.

For image captions, this is how we inject image features into the decoder.

Vaswani et al, “Attention is all you need”, NeurIPS 2017

# Cross-attention

- **Self-attention:**
  - Keys, queries, and values from same source
- **Cross-attention**
  - The keys and values are from encoder (like a memory)
  - The queries are from the decoder