Introduction to Machine Learning

Attention Mechanism and Transformers

Prof. Subrahmanya Swamy Peruru
Department of Electrical Engineering
IIT Kanpur

Attention

I swam across the river to get to the other bank.
I walked across the road to get cash from the bank.

- Word embedding of the word ‘bank’ is the same in both the sentences. But they have different meaning.
- The correct interpretation can be understood if we pay attention to the other words in the sentence.

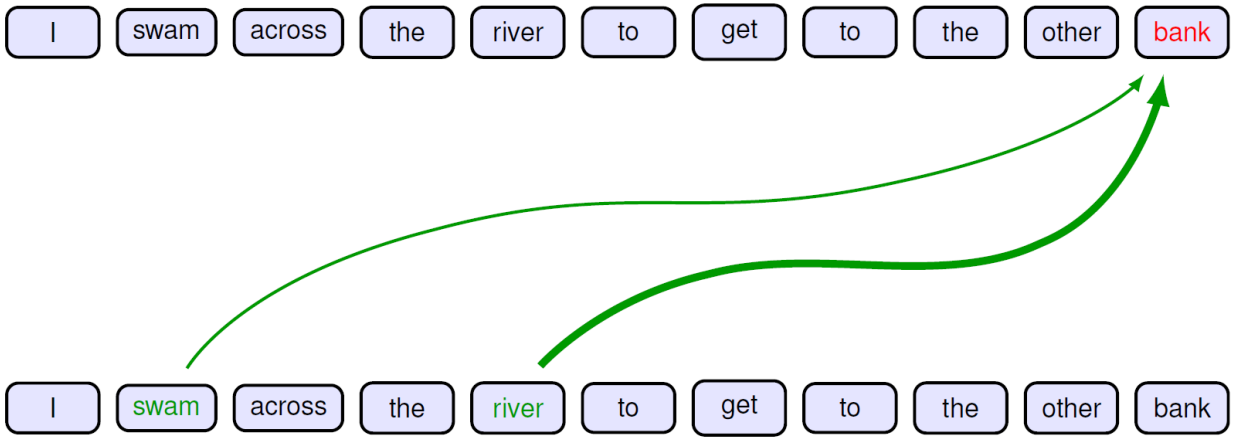
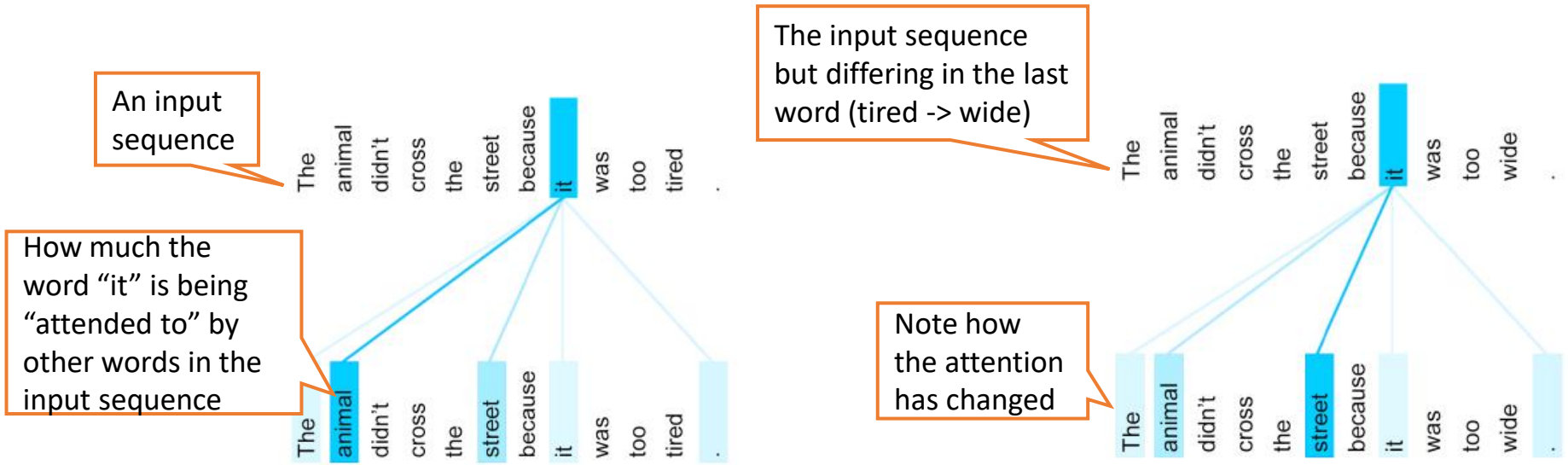


Fig credit: Bishop book on Deep Learning

Attention

- With self-attention, each token $\boldsymbol{x}_n$ can "attend to" all other tokens of the same sequence when computing this token's embedding $\boldsymbol{h}_n$

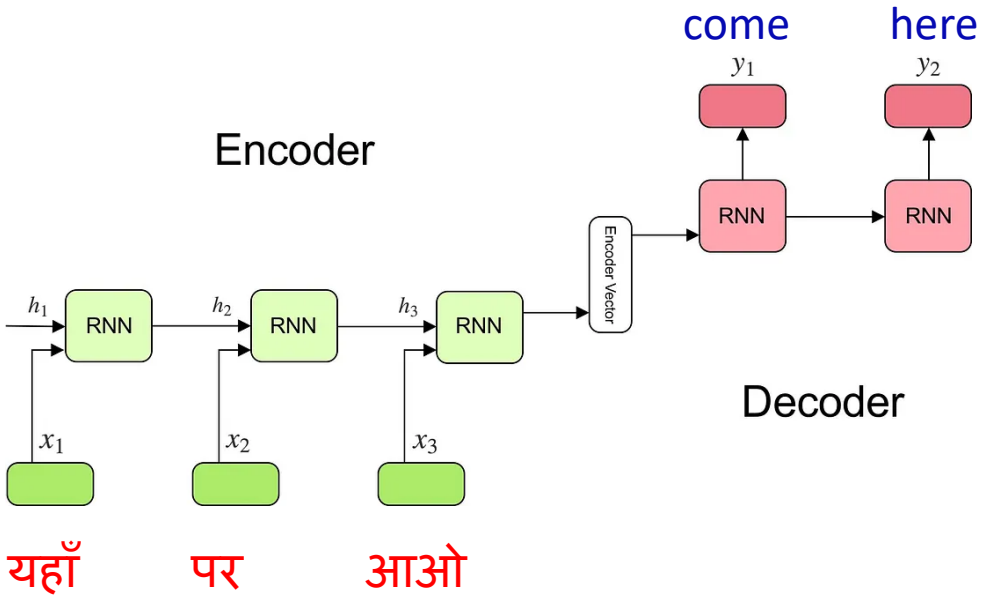


- Attention helps capture the context better and in a much more "global" manner
  - "Global": Long ranges captures and in both directions (previous and ahead)

Example credit: https://blog.research.google/2017/08/transformer-novel-neural-network.html

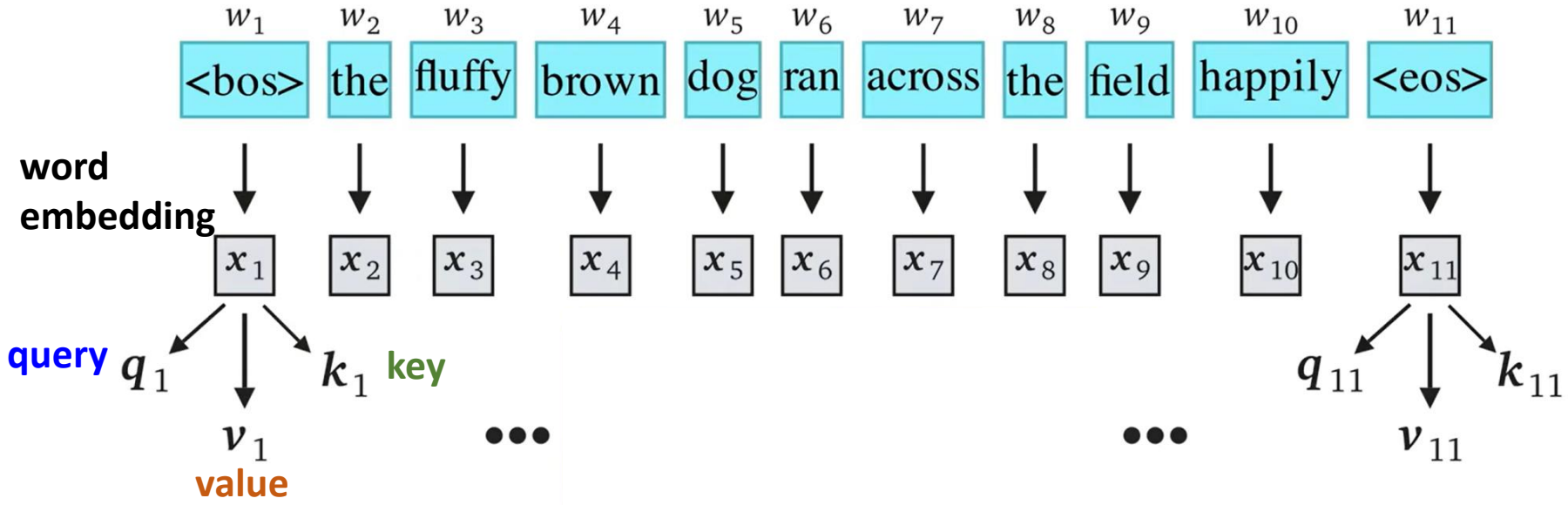
Recap: RNNs

- RNNs are used when each input or output or both are sequences of tokens



- Slow processing: Can't compute $\boldsymbol{h}_t$ before computing $\boldsymbol{h}_{t-1}$
- Information bottleneck:
  - Hidden state $\boldsymbol{h}_t$ is supposed to remember everything up to time $t-1$.
  - Variants such as LSTM, GRU, etc mitigate this issue to some extent

Attention

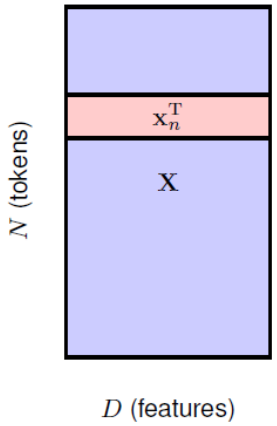


Learnable $D \times d$ weight matrix

$$q_n = \mathbf{W_Q}\, x_n$$
$$v_n = \mathbf{W_V}\, x_n$$
$$k_n = \mathbf{W_K}\, x_n$$

$d \times 1$ $D \times 1$

Can be computed parallelly for all N tokens using matrix multiplications



Row n is $\boldsymbol{q}_n$

$$\boldsymbol{Q} = \boldsymbol{X}\boldsymbol{W}_Q$$

$$\boldsymbol{K} = \boldsymbol{X}\boldsymbol{W}_K$$

$$\boldsymbol{V} = \boldsymbol{X}\boldsymbol{W}_V$$

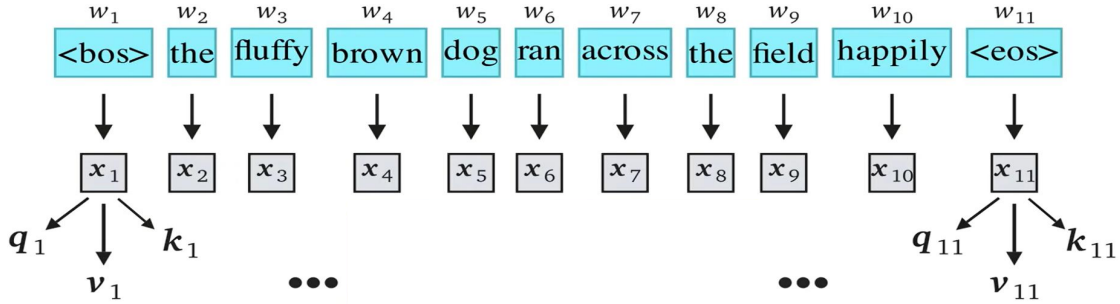
Fig Credits: https://www.youtube.com/watch?v=mmzRYGCfTzc

Self-Attention

- Attention scores

How much token i attends to token n

$$\alpha_{n,i} = \frac{\exp(\boldsymbol{q}_n^\top \boldsymbol{k}_i)}{\sum_{j=1}^{N} \exp(\boldsymbol{q}_n^\top \boldsymbol{k}_j)}$$



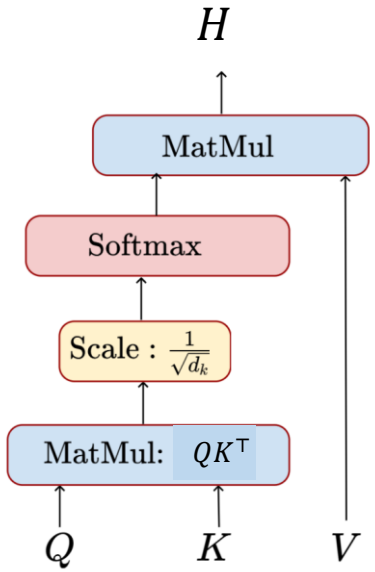
- Contextual Embedding (Hidden state) for $\boldsymbol{x}_n$ is

$$\boldsymbol{h}_n = \sum_{i=1}^{N} \alpha_{n,i}\, \boldsymbol{v}_i$$

Scaled Dot Product

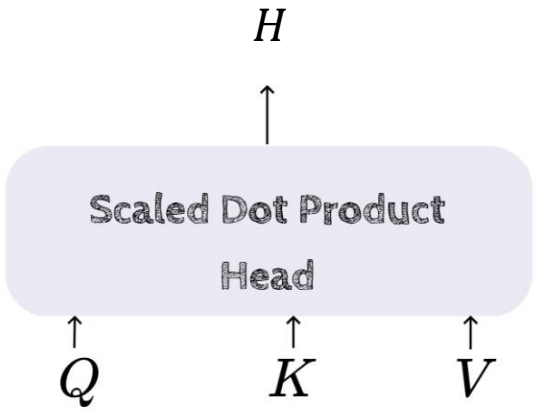
$$\boldsymbol{H} = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}}\right)\boldsymbol{V}$$

Dividing by $\sqrt{d}$ ensures variance of the dot product is 1



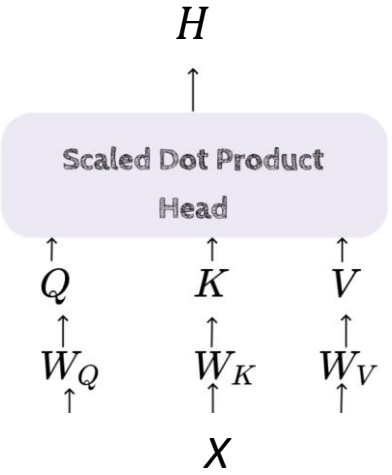
- Matrix form

$$\boldsymbol{H} = \text{softmax}(\boldsymbol{Q}\boldsymbol{K}^\top)\,\boldsymbol{V}$$



Compactly

Attention Head



Multi-Head Attention

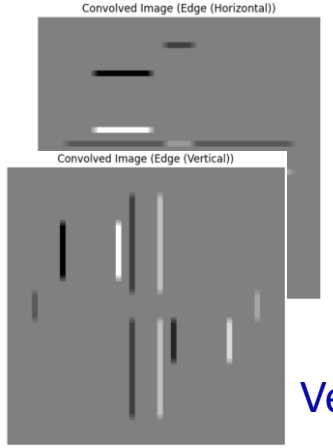
Recall from CNNs that multiple filters allow us to extract different kinds of features

Filter 1

$$\begin{matrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix}$$

Filter 2

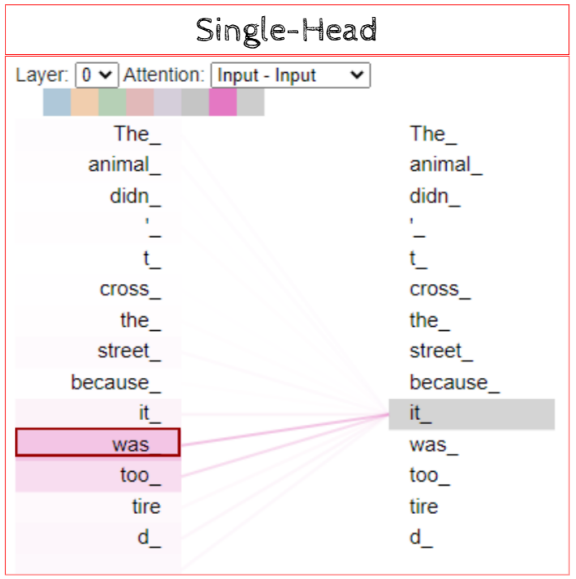
$$\begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}$$

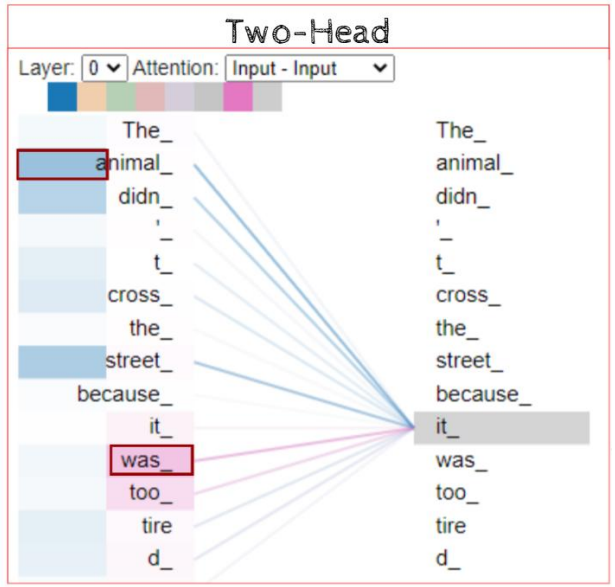


Horizontal edge features

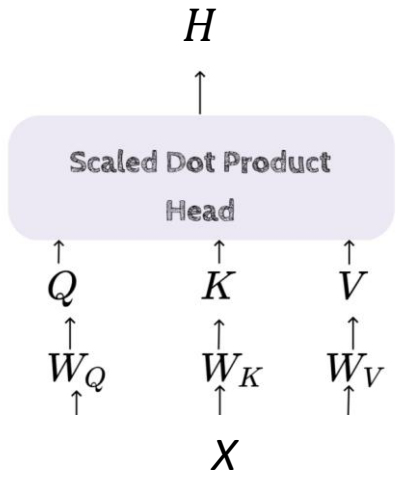
Vertical edge features

Similarly multiple attention heads allow us to learn different kinds of embeddings

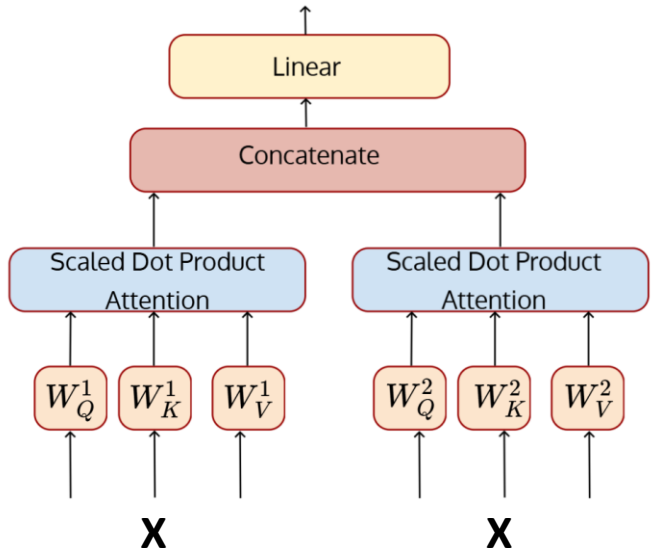




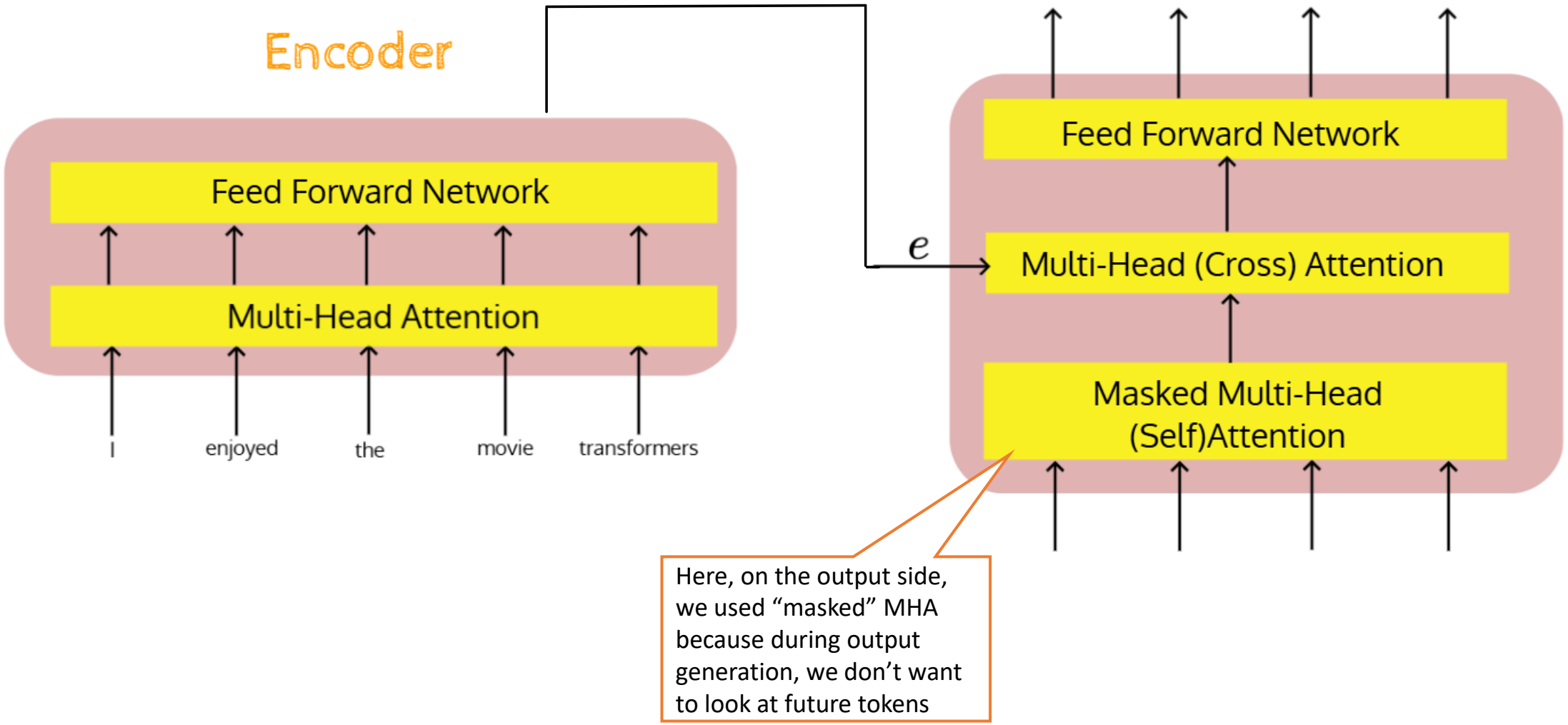
Single Attention Head



Multi Attention Head

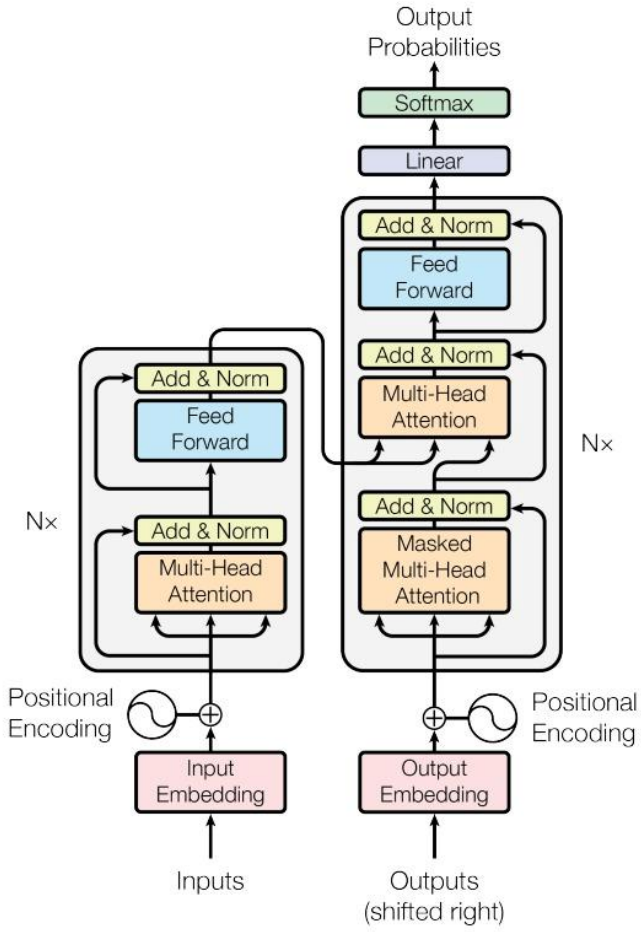


Encoder-Decoder Architecture



Transformers Architecture

- Transformers use the idea of attention
  - "self-attention" over all input tokens
  - Masked "self-attention" over each output token and previous tokens
  - "cross-attention" between output tokens and input tokens
- Transformer also compute embeddings of all tokens in parallel
- Transformers are based on the following key ideas*
  - "Self-attention" and "cross-attention" for computing the hidden states
  - Positional encoding (to encode relative distances between words)
  - Residual connections
- Attention helps capture the context better and in a much more "global" manner in sequence data



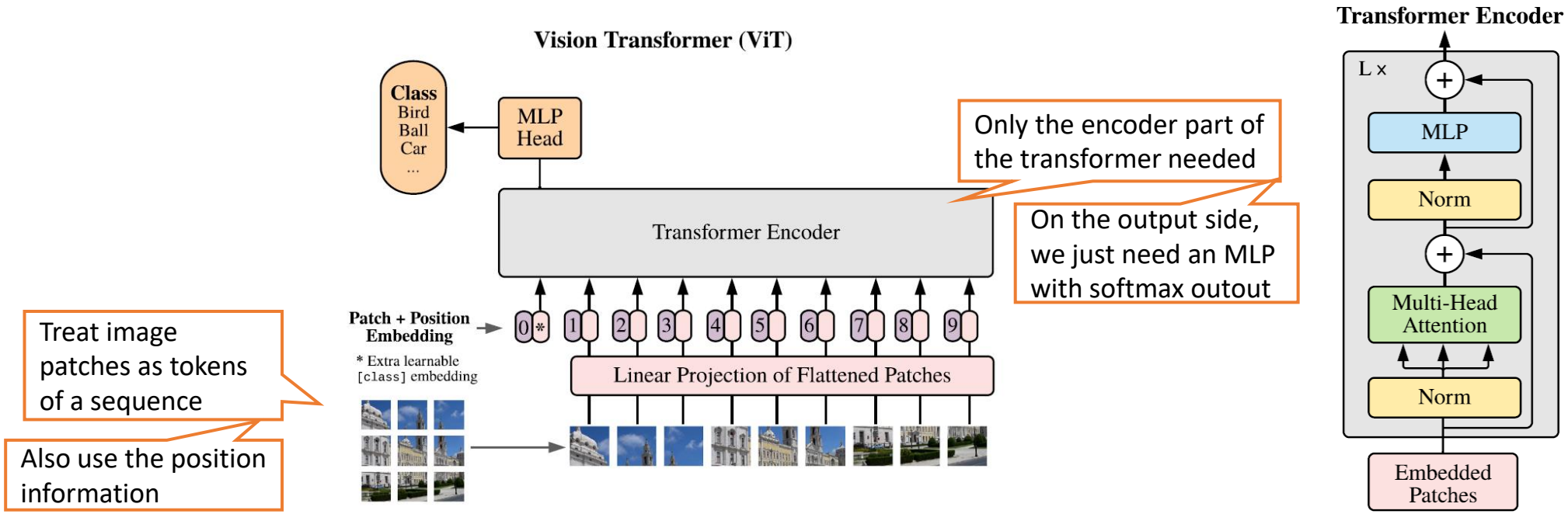
"Attention is all you need" (Vaswani et al, 2017)

Popular Transformer Variants: BERT and GPT

- The standard transformer architecture is an encoder-decoder model
- Some models use just the encoder or the decoder of the transformer
- BERT (Bidirectional Encoder Representations from Transformers)
  - Basic BERT can be learned to encode token sequences
- GPT (Generative Pretrained Transformer)
  - Basic GPT can be used to generate token sequences similar to its training data

Transformers for Images: ViT

- Transformers can be used for images as well. For image classification, it looks like this



- Early work showed ViT can outperform CNNs given very large amount of training data
- However, recent work* has shown that good old CNNs still rule! ViT and CNN perform comparably at scale, i.e., when both given large amount of compute and training data

An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale (Dosovitskiy et al, 2020)

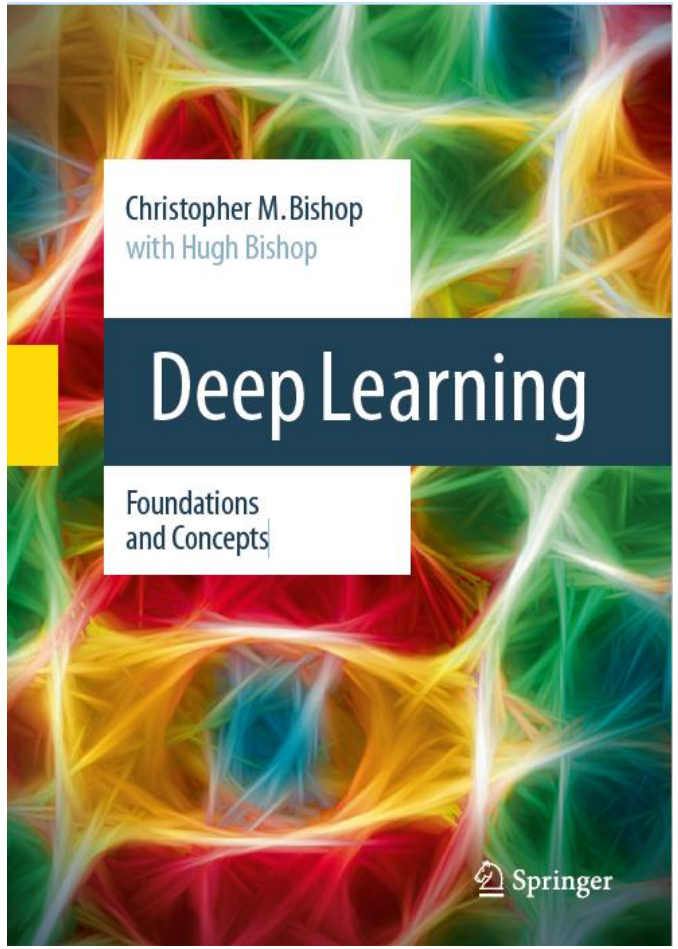
*ConvNets Match Vision Transformers at Scale (Smith et al, 2023)

Acknowledgments

- Slides are adapted from
  - Prof. Piyush Rai's course at IITK
  - Prof. Mitesh Khapra's course at IITM

References

Chapter 12



Thank you