Transformers and LLMs for Signal Processors (and Applied Mathematicians)

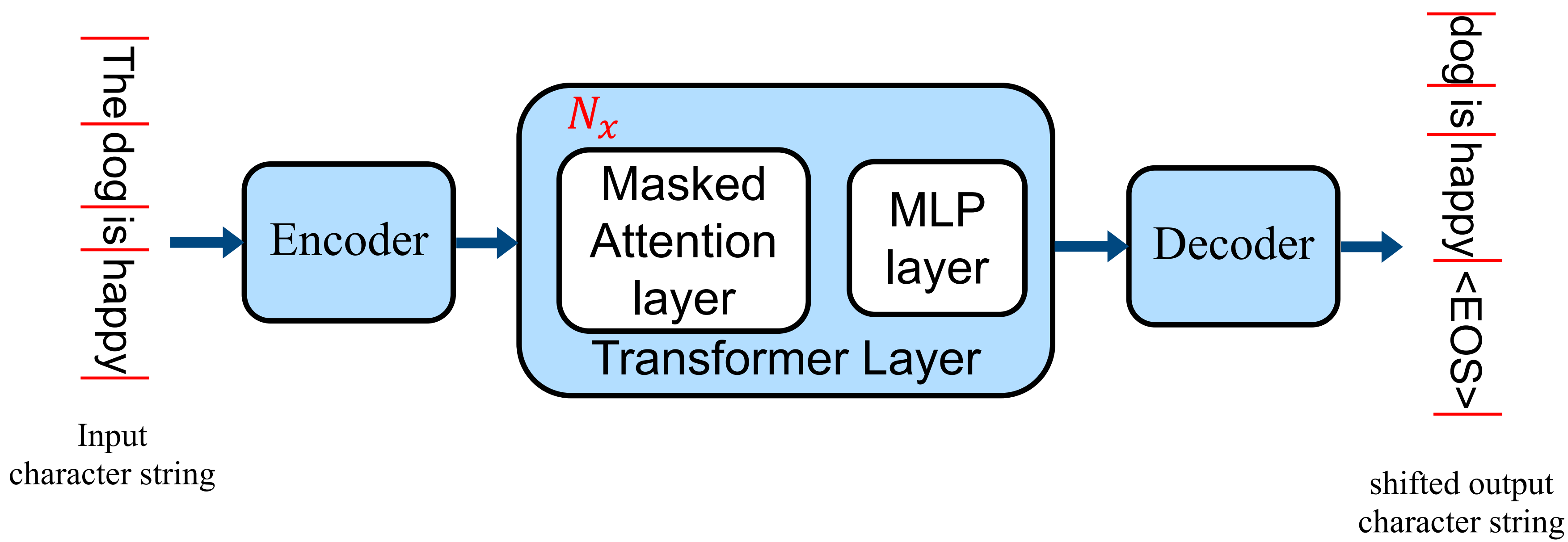
Charles A. Bouman
School of Electrical and Computer Engineering
Purdue University

Overview

- Key ideas:
  - All the basic concepts of Transformers and LLMs
  - Cover concepts in a hierarchical manner
  - Emphasis abstract/modular thinking
  - Draw flow diagrams from left to right!
- Topics:
  - Attention layers
  - Transformer layers
  - Encoding and decoding
  - Generative Pretraining
  - Inference

Overview of LLMs

- Model
  - Inputs character sequences
  - Output token probabilities
  - Uses masked attention to enforce causal prediction
  - Autoregressive model of language (or other things).

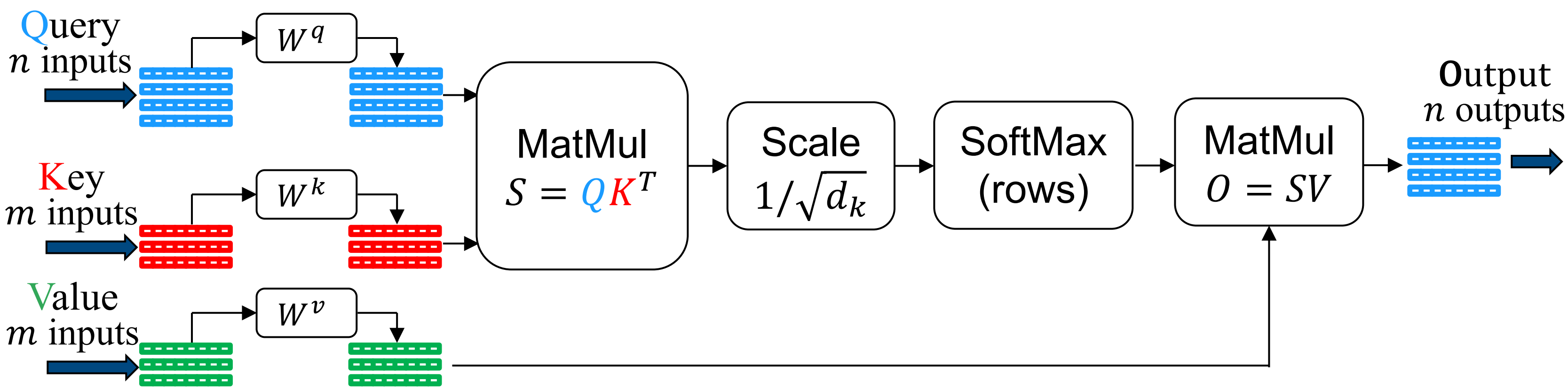


Transformer Architectures

- Single-Head Attention
- Score and Selection
- Multi-Head Attention
- Self Attention
- Masked Multi-Head Attention
- Pre-Normalization

Single-Head Attention Layer

- $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$



Input Shapes:

$Q \in \mathbb{R}^{n \times d_m}$

$K \in \mathbb{R}^{m \times d_m}$

$V \in \mathbb{R}^{m \times d_m}$

Model Parameters:

$W^q \in \mathbb{R}^{d_m \times d_k}$

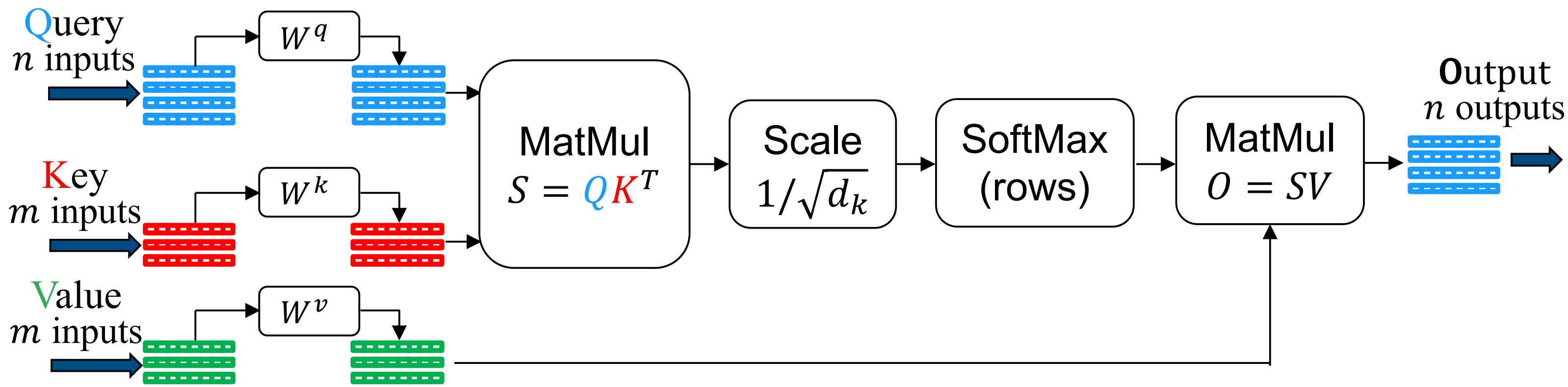
$W^k \in \mathbb{R}^{d_m \times d_k}$

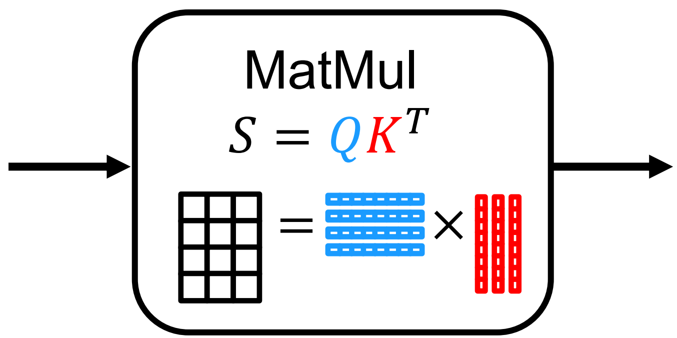
$W^v \in \mathbb{R}^{d_m \times d_h}$

Out Shape:

$O \in \mathbb{R}^{n \times d_h}$

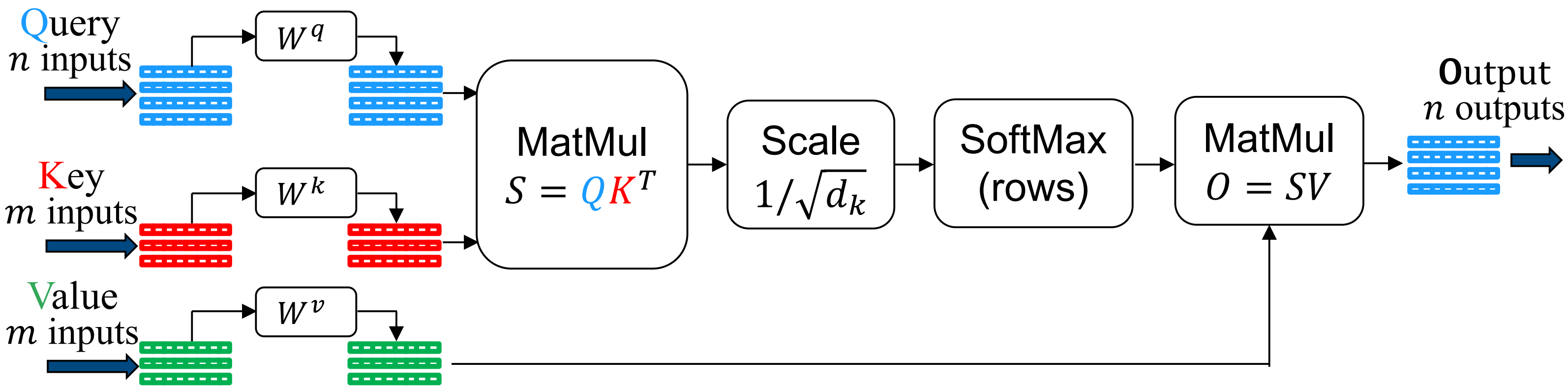
Attention Layer: Score Calculation

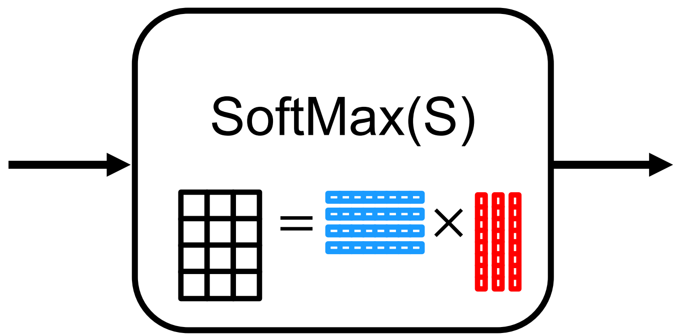




	a	b	c
a	96	66	12
b	66	92	10
c	12	10	89
d	30	16	22

Attention Layer: SoftMax



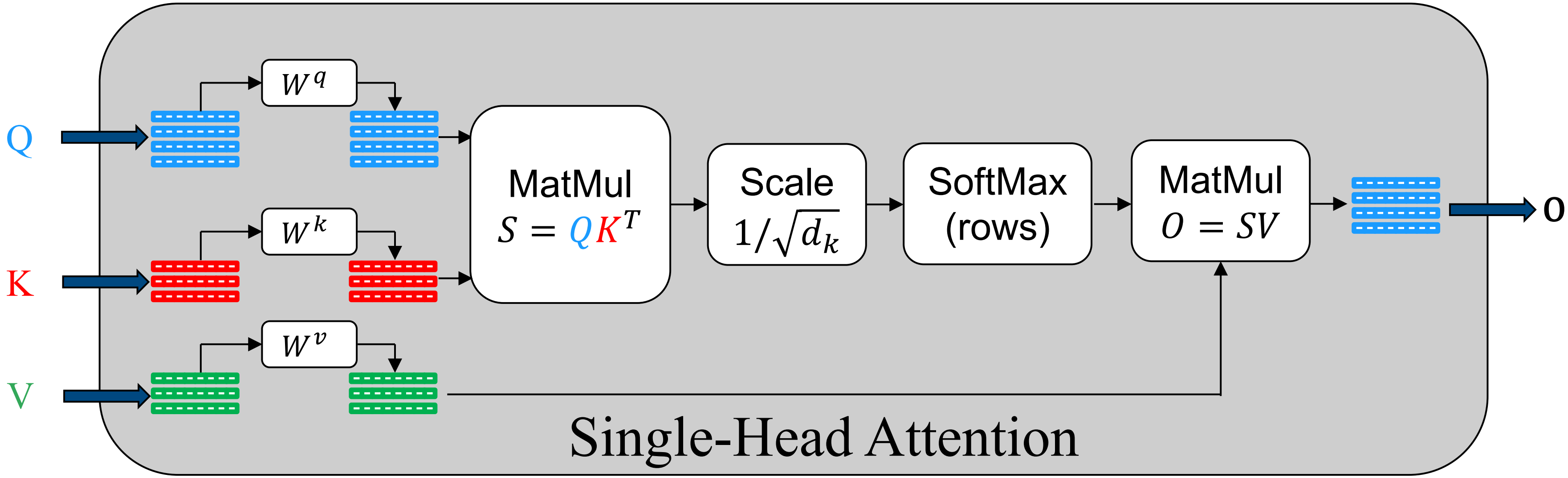


	a	b	c
a	0.80	0.10	0.10
b	0.10	0.80	0.10
c	0.10	0.10	0.80
d	0.33	0.33	0.33

$$\mathrm{SoftMax}(x)_{k,i} = \frac{\exp(x_{k,i})}{\sum_j \exp(x_{k,j})}$$

Single-Head Attention Layer

- Intuition:
  - $\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$
  - Finds the best matching key and selects the corresponding value.



Input Shapes:

$Q \in \mathbb{R}^{n \times d_m}$
$K \in \mathbb{R}^{m \times d_m}$
$V \in \mathbb{R}^{m \times d_m}$

Model Parameters:

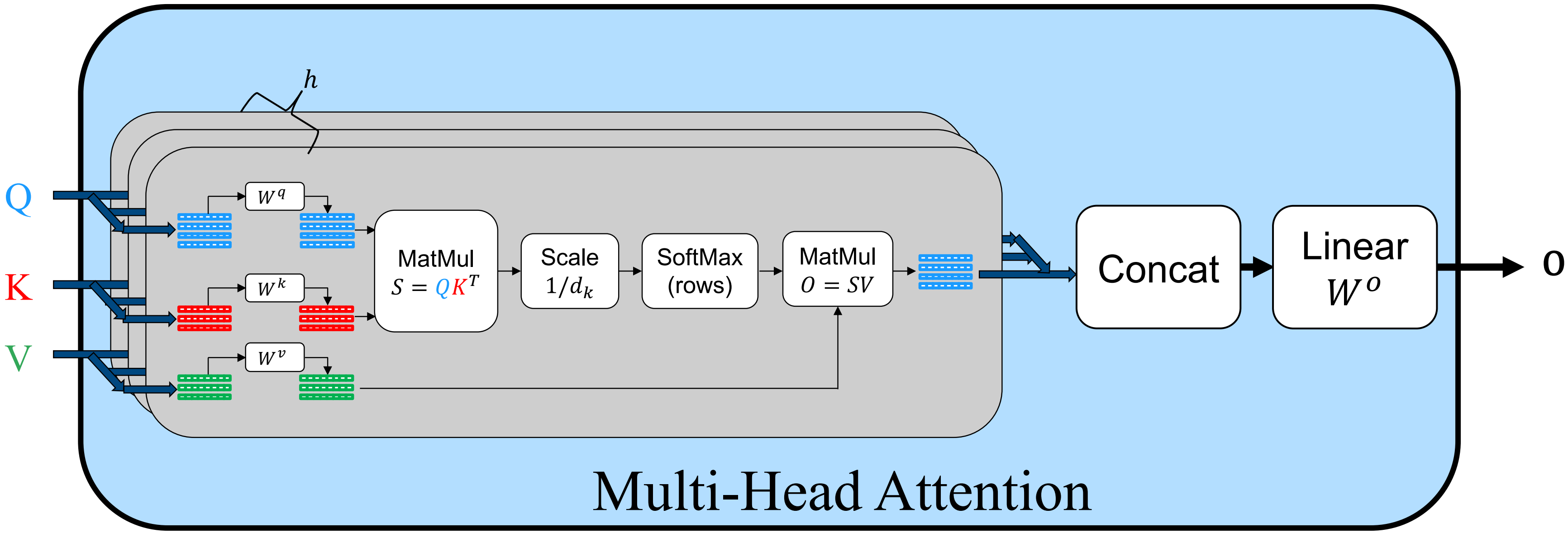
$W^q \in \mathbb{R}^{d_m \times d_k}$
$W^k \in \mathbb{R}^{d_m \times d_k}$
$W^v \in \mathbb{R}^{d_m \times d_h}$

Out Shape:

$O \in \mathbb{R}^{n \times d_h}$

Multi-Head Attention Layer

- $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$



Input Shapes:

$Q \in \mathbb{R}^{n \times d_m}$

$K \in \mathbb{R}^{m \times d_m}$

$V \in \mathbb{R}^{m \times d_m}$

Model Parameters:

$W_i^q \in \mathbb{R}^{d_m \times d_k}$

$W_i^k \in \mathbb{R}^{d_m \times d_k}$

$W_i^v \in \mathbb{R}^{d_m \times d_h}$

$W^o \in \mathbb{R}^{hd_h \times d_m}$

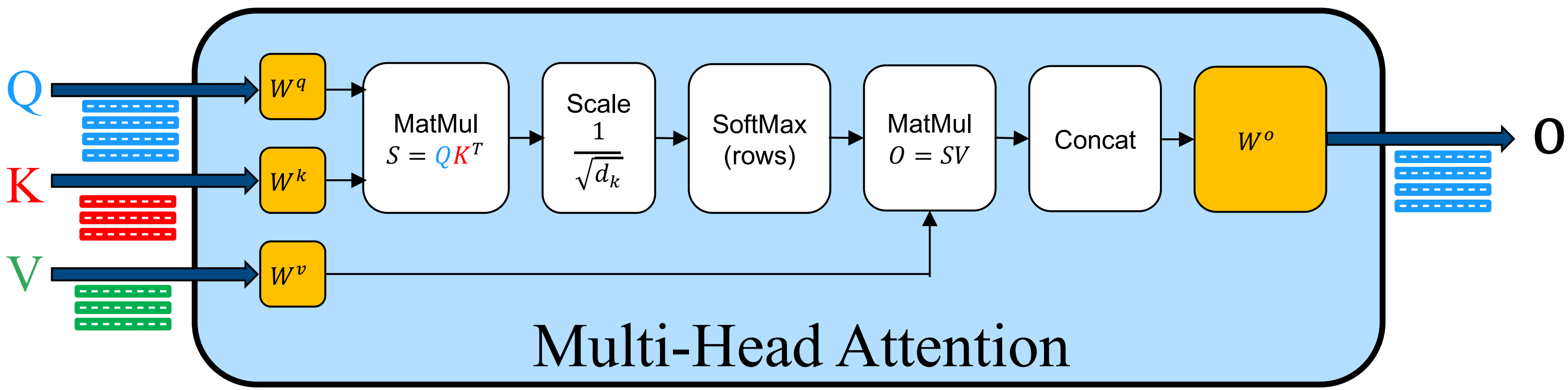
Out Shape:

$O \in \mathbb{R}^{n \times d_m}$

i – head index

Multi-Head Attention Layer

- Interpretation:
  - O has the same shape as Q
  - n can change
  - D_m determines the model order



Input Shapes:

$$Q \in \mathbb{R}^{n \times d_m}$$
$$K \in \mathbb{R}^{m \times d_m}$$
$$V \in \mathbb{R}^{m \times d_m}$$

Model Parameters:

$$W_i^q \in \mathbb{R}^{d_m \times d_k}$$
$$W_i^k \in \mathbb{R}^{d_m \times d_k}$$
$$W_i^v \in \mathbb{R}^{d_m \times d_h}$$
$$W^o \in \mathbb{R}^{hd_h \times d_m}$$

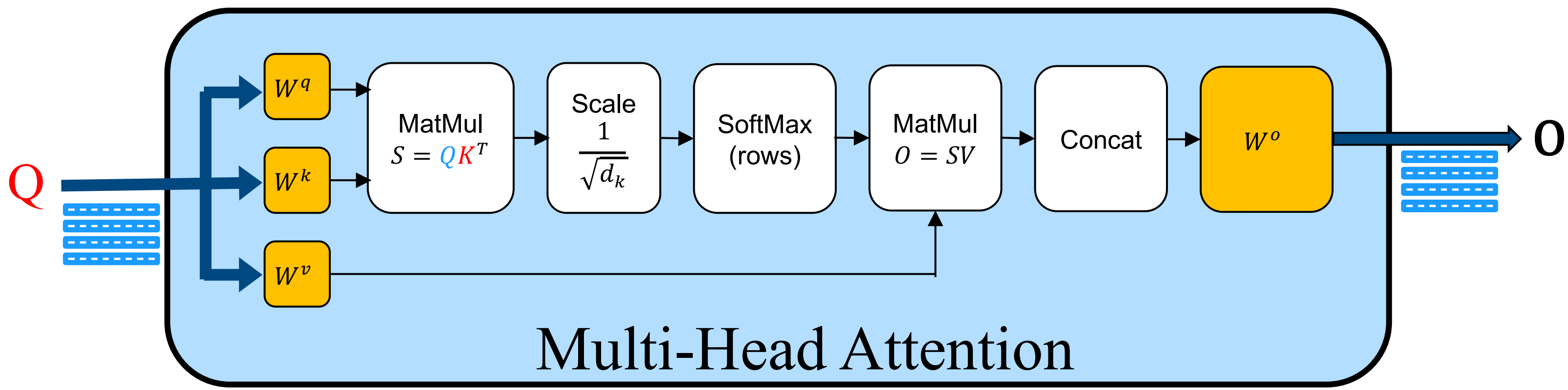
Out Shape:

$$O \in \mathbb{R}^{n \times d_m}$$

i – head index

Self Attention

- What is "Self Attention"?
  - Special case when $Q = K = V$
  - All inputs are the same.
  - S is a square matrix.



Input Shapes:

$Q \in \mathbb{R}^{n \times d_m}$

Model Parameters:

$W_i^q \in \mathbb{R}^{d_m \times d_k}$
$W_i^k \in \mathbb{R}^{d_m \times d_k}$
$W_i^v \in \mathbb{R}^{d_m \times d_h}$
$W^o \in \mathbb{R}^{hd_h \times d_m}$

Out Shape:

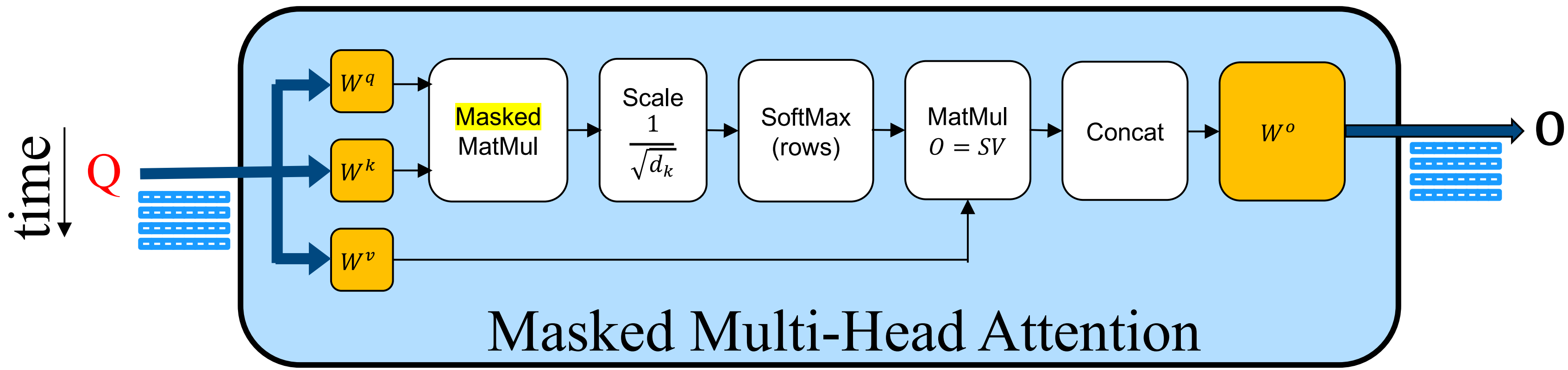
$O \in \mathbb{R}^{n \times d_m}$

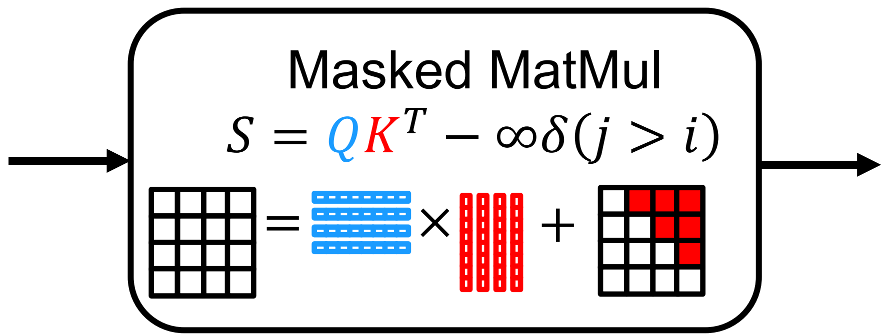
i – head index

Masked Multi-Head Attention

- What is it?
  - Used to enforce causal dependencies in data
  - The basis of autoregressive models
  - Crucial component of LLM
  - Only used with Self Attention, i.e., when $n = m$.
- How is it implemented?
  - Add $-\infty$ to upper triangular components of Score matrix

Masked Multi-Head Self Attention





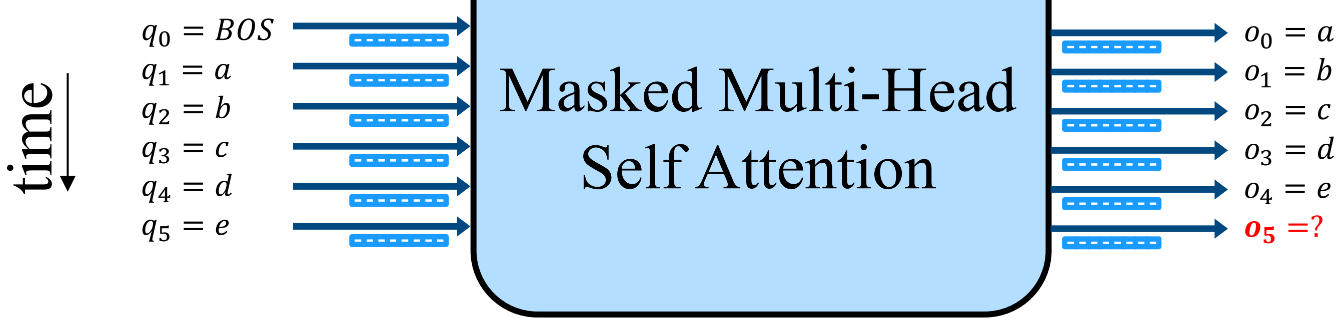
	a	b	c	d
a	96	$-\infty$	$-\infty$	$-\infty$
b	66	92	$-\infty$	$-\infty$
c	12	10	89	$-\infty$
d	30	16	22	22

- Intuition:
  - Only uses “past” values of Q in attention
  - Useful in speech and natural language

What is Masked Self Attention Good For?

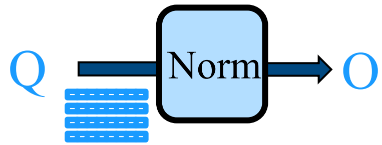
- Autoregressive prediction:
  - Train both the input and output using sentence sequences
  - The last token output o_{n-1} is a prediction of the next next symbol
  - Useful in speech and natural language
  - Has largely replaced RNNs

$$Q = \begin{bmatrix} q_0 \\ \vdots \\ q_{n-1} \end{bmatrix} \qquad O = \begin{bmatrix} o_0 \\ \vdots \\ o_{n-1} \end{bmatrix}$$



Normalization Layer

- Normalization



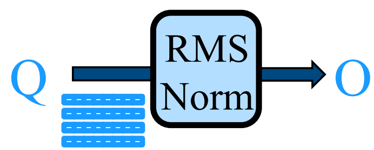
$$\mu_i \leftarrow \frac{1}{d_m}\sum_j Q_{i,j}$$

$$\sigma_i \leftarrow \sqrt{\frac{1}{d_m}\sum_j \left(Q_{i,j} - \mu_i\right)^2}$$

$$O_{i,j} \leftarrow \frac{Q_{i,j} - \mu_i}{\sigma_i}$$

- Important: These are not learned parameters.
- Each row is normalized

- RMS normalization



$$\mu_i \leftarrow \frac{1}{d_m}\sum_j Q_{i,j}$$

$$\sigma_i \leftarrow \sqrt{\frac{1}{d_m}\sum_j \left(Q_{i,j} - \mu_i\right)^2}$$

$$O_{i,j} \leftarrow \frac{Q_{i,j}}{\sigma_i}$$

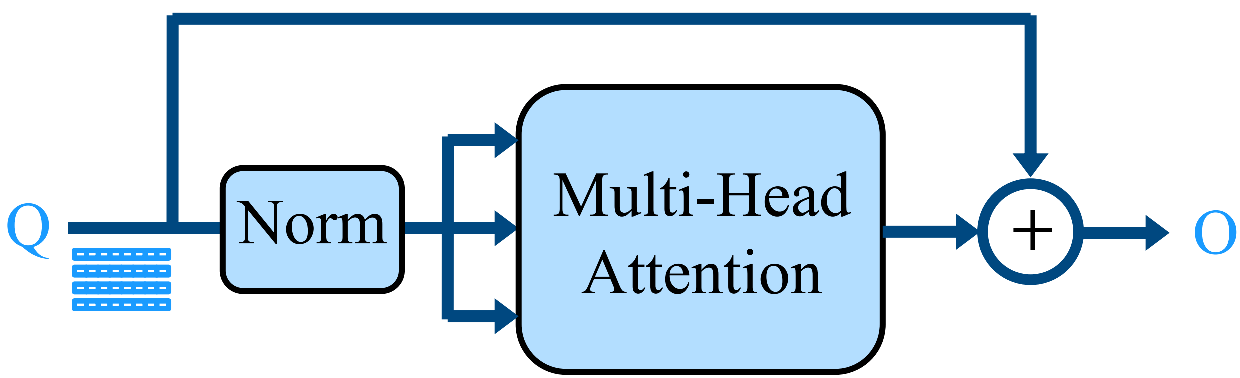
RMS normalization: Zhangyu Shi and Xiao Liu and Jianfei Chen and Jun Zhu, "RMSNorm: Simple Normalization for Transformers," arXiv preprint arXiv:1910.07467, 2019, https://arxiv.org/pdf/1910.07467 .

GPT pre-norm: Zhangyu Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, Tie-Yan Liu, "On Layer Normalization in the Transformer Architecture," arXiv preprint arXiv:2002.04745, 2020, https://arxiv.org/pdf/2002.04745 .

Layer normalization: Jimmy Lei Ba and Jamie Ryan Kiros and Geoffrey E Hinton, "Layer Normalization," arXiv preprint arXiv:1607.06450, 2026, https://arxiv.org/pdf/1607.06450 .

Pre-Normed Multi-Head Self Attention

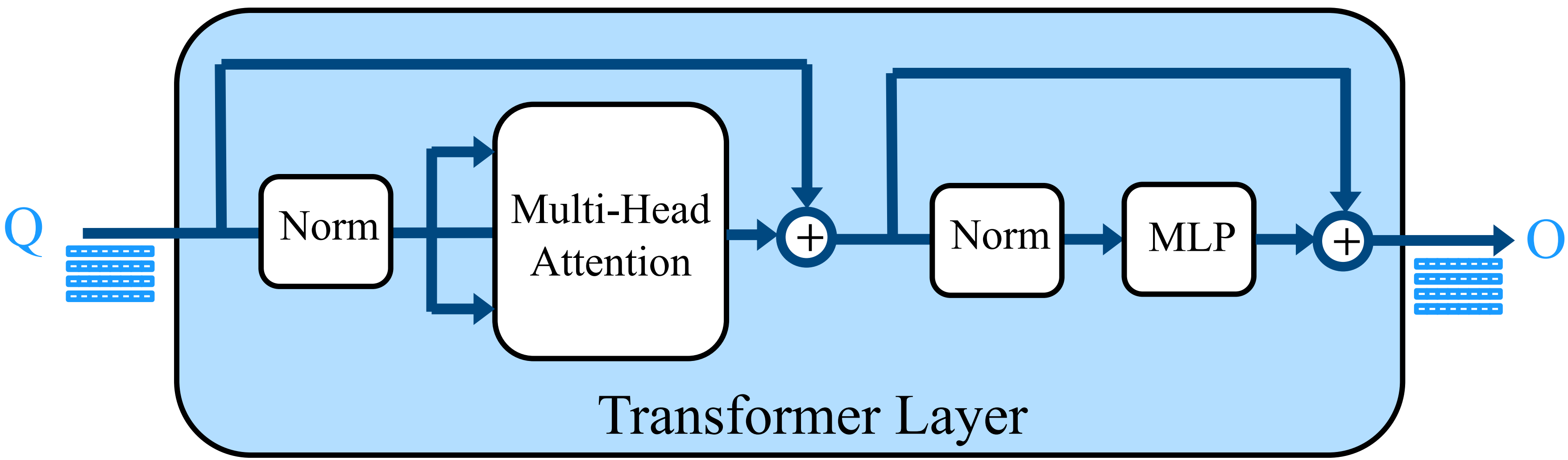
- MHA typically:
  - Is preceded by normalization
  - Uses a skipped connection
- Together, they improve convergence in training.



Pre-Normalization:
Yuzhao Xiong and Zhe Gan and Yi Cheng and Yunxin Zhao and Jianfeng Gao and Zicheng Liu and Michael Zeng, “On Layer Normalization in the Transformer Architecture,” Proceedings of the 37th International Conference on Machine Learning (ICML), 2020, https://arxiv.org/pdf/2002.04745 .

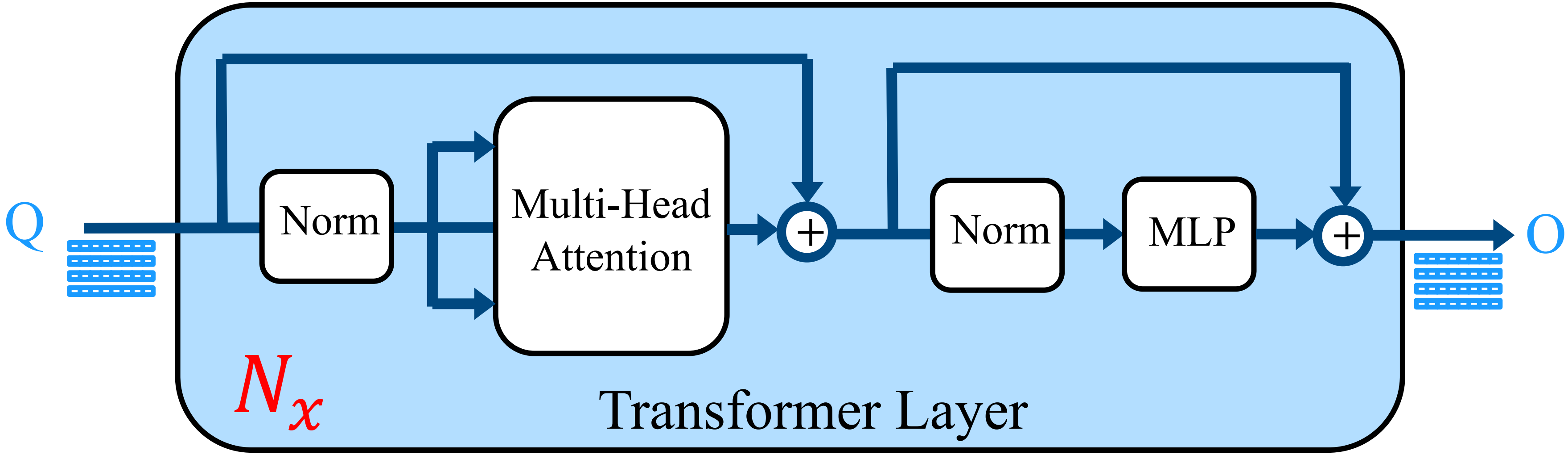
Transformer Layer

- A Modern Transformer layer typically consists of:
  - Pre-Norm and MHA with skip connection
  - Pre-Norm and MLP with skip connection
  - MLP is applied to each token ⇔ CNN with a 1x1 kernel
  - Shape of input and output are the same.



The Transformer

- A Transformer is then formed by a series of transformer layers



Input Shapes:

$Q \in \mathbb{R}^{n \times d_m}$

Model Parameters:

$W_{i,j}^{q} \in \mathbb{R}^{d_m \times d_k}$

$W_{i,j}^{k} \in \mathbb{R}^{d_m \times d_k}$

$W_{i,j}^{v} \in \mathbb{R}^{d_m \times d_h}$

$W_{j}^{o} \in \mathbb{R}^{h d_h \times d_m}$

$W_{i,j}^{m1} \in \mathbb{R}^{d_m \times d_{mlp}}$

$W_{i,j}^{m1} \in \mathbb{R}^{d_{mlp} \times d_m}$

Out Shape:

$O \in \mathbb{R}^{n \times d_m}$

i – head index

j – layer index

Typical Model Sizes

- GPT-2 117M Small
  - $N_x = 12$, $d_m = 768$, $h = 12$, $d_h = \frac{d_m}{h} = 64$, d_{mlp} =3072, context window = 1024 tokens
  - Number of parameters = 117M
- GPT-2 1.5B XL
  - $N_x = 64$, $d_m = 1600$, $h = 25$, $d_h = \frac{d_m}{h} = 64$, d_{mlp} =6400, context window = 1024 tokens
  - Number of parameters = 1.5B
- LLAMA-1 65B (2023)
  - $N_x = 80$, $d_m = 8192$, $h = 64$, $d_h = \frac{d_m}{h} = 128$, d_{mlp} =22016, context window = 2048 tokens
  - Number of parameters = 65B
- LLAMA-2 70B (2023)
  - $N_x = 80$, $d_m = 8192$, $h = 64$, $d_h = \frac{d_m}{h} = 128$, d_{mlp} =28672, context window = 4096 tokens
  - Number of parameters = 65B
- LLAMA-3 70B (2024)
  - $N_x = 80$, $d_m = 8192$, $h = 64$, $d_h = \frac{d_m}{h} = 128$, d_{mlp} =28672, context window = 8192 tokens
  - Number of parameters = 65B

Citations for Attention and Transformers

- Early Attention:
  - Samuel R Bowman and Gabor Angeli and Christopher Potts and Christopher D Manning, “A Decomposable Attention Model for Natural Language Inference,” Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2016, https://arxiv.org/pdf/1606.01933 .
- Modern Attention and Transformers:
  - Ashish Vaswani and Noam Shazeer and Niki Parmar and Jakob Uszkoreit and Llion Jones and Aidan N Gomez and Lukasz Kaiser and Illia Polosukhin, “Attention Is All You Need,” Advances in Neural Information Processing Systems (NeurIPS), 2017, https://arxiv.org/pdf/1706.03762 .
- Improved normalization
  - Layer normalization: Jimmy Lei Ba and Jamie Ryan Kiros and Geoffrey E Hinton, “Layer Normalization," arXiv preprint arXiv:1607.06450, 2026, https://arxiv.org/pdf/1607.06450 .
  - RMS normalization: Zhangyu Shi and Xiao Liu and Jianfei Chen and Jun Zhu, “RMSNorm: Simple Normalization for Transformers,” arXiv preprint arXiv:1910.07467, 2019, https://arxiv.org/pdf/1910.07467 .
  - Pre-Normalization: Yuzhao Xiong and Zhe Gan and Yi Cheng and Yunxin Zhao and Jianfeng Gao and Zicheng Liu and Michael Zeng, “On Layer Normalization in the Transformer Architecture,” Proceedings of the 37th International Conference on Machine Learning (ICML), 2020, https://arxiv.org/pdf/2002.04745 .
- Visual Transformer:
  - Vision Transformer: Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby, “An Image is Worth 16*16 Words: Transformers for Image Recognition at Scale," ICLR 2021, https://arxiv.org/pdf/2010.11929 .

GPT Style LLM Models*

- Tokenization
- Embedding
- Autoregressive prediction
- Pretraining

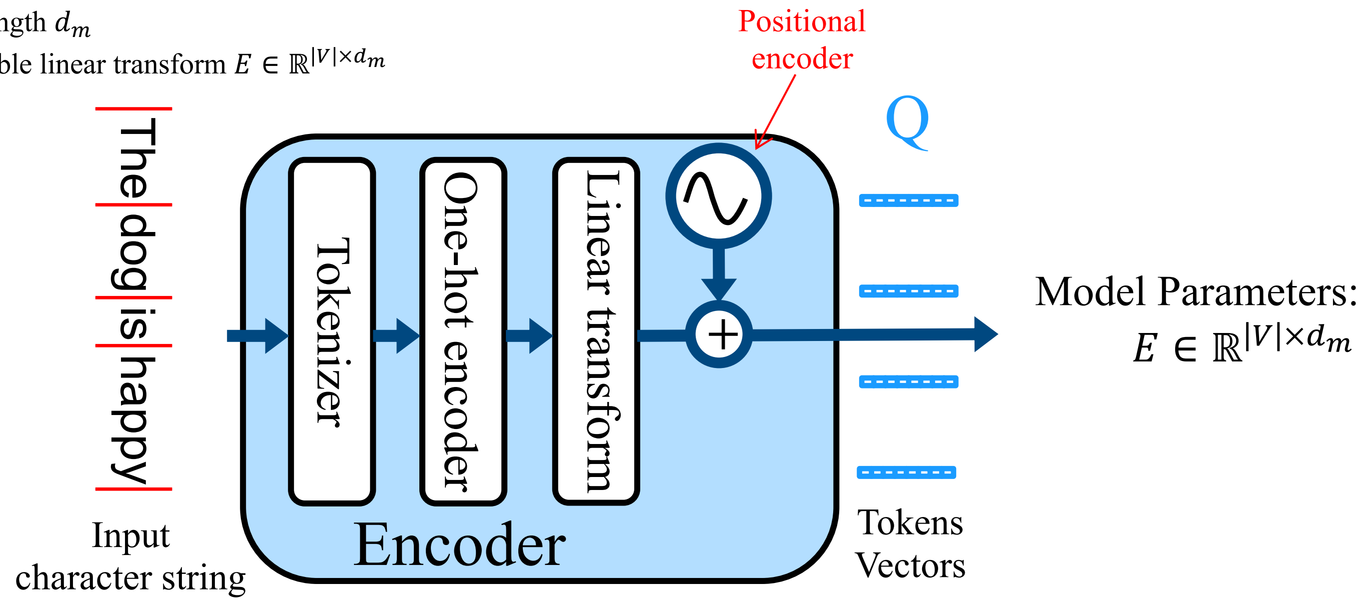
Vision Transformer:
Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby, "An Image is Worth 16*16 Words: Transformers for Image Recognition at Scale," ICLR 2021, https://arxiv.org/pdf/2010.11929 .

Citations for LLMs

- GPT papers:
  - Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, “Improving Language Understanding by Generative Pre-Training,” OpenAI Technical Report, 2018. https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf .
  - Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever, “Language Models are Unsupervised Multitask Learners”, OpenAI Technical Report, 2019, https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf .
  - Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan Pranav, Shyam Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei, “Language Models are Unsupervised Multitask Learners”, NerIPS, 2020, https://arxiv.org/pdf/2005.14165 .

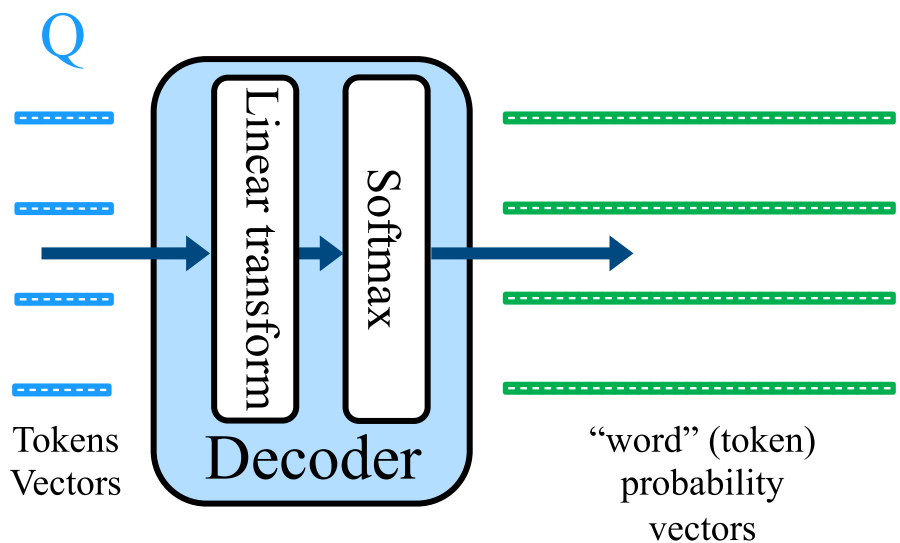
LLM Encoders

- What are tokens?
  - Sequencies of characters are grouped together based on the probability of occurrence into a large vocabulary V of size $|V| \sim 50K$.
    - V includes all characters
    - Every entry of entry of V is assigned a unique integer token. $T: V \to \{0, \dots, |V|-1\}$
- What is tokenization?
  - Every character string can be encoded as a tokens sequence.
  - Every token sequence can be decoded as character string.
    - Tutorial: https://sebastianraschka.com/blog/2025/bpe-from-scratch.html
    - Example: https://tiktokenizer.vercel.app/?model=gpt-4
- What is the output of the encoder?
  - Token vectors are of length d_m
  - Computed using learnable linear transform $E \in \mathbb{R}^{|V| \times d_m}$



LLM Decoder

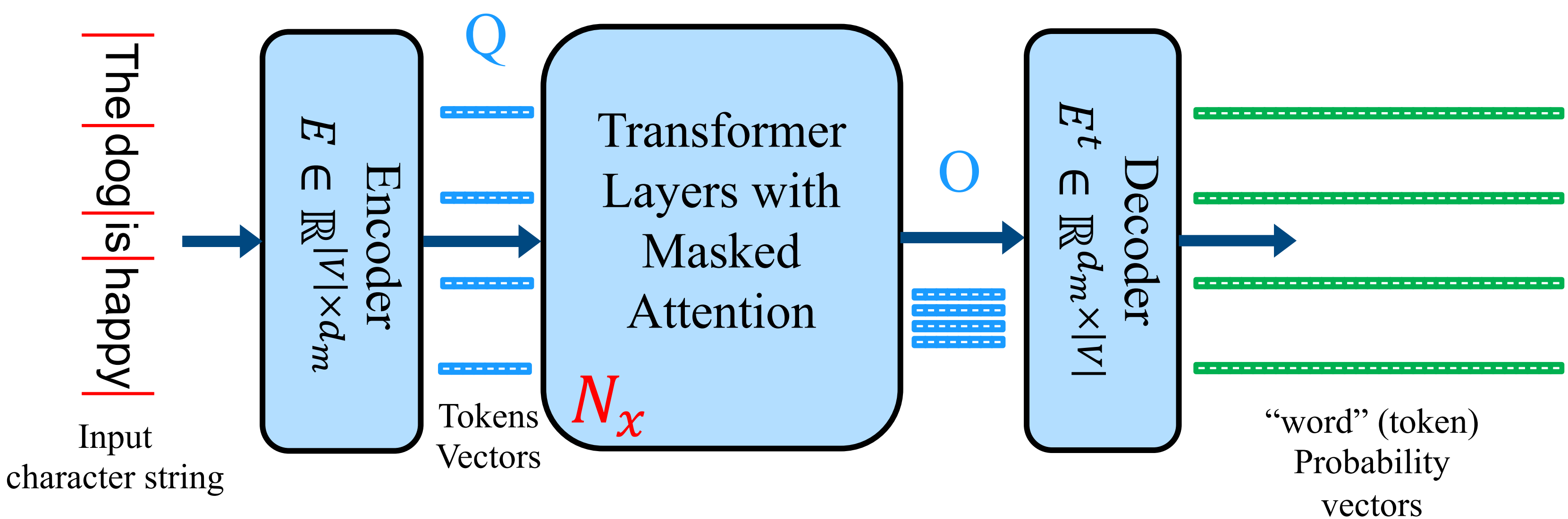
- Decoder
  - Reverses encoding process
  - But produces token probabilities, rather than tokens



Model Parameters:
$E^t \in \mathbb{R}^{d_m \times |V|}$

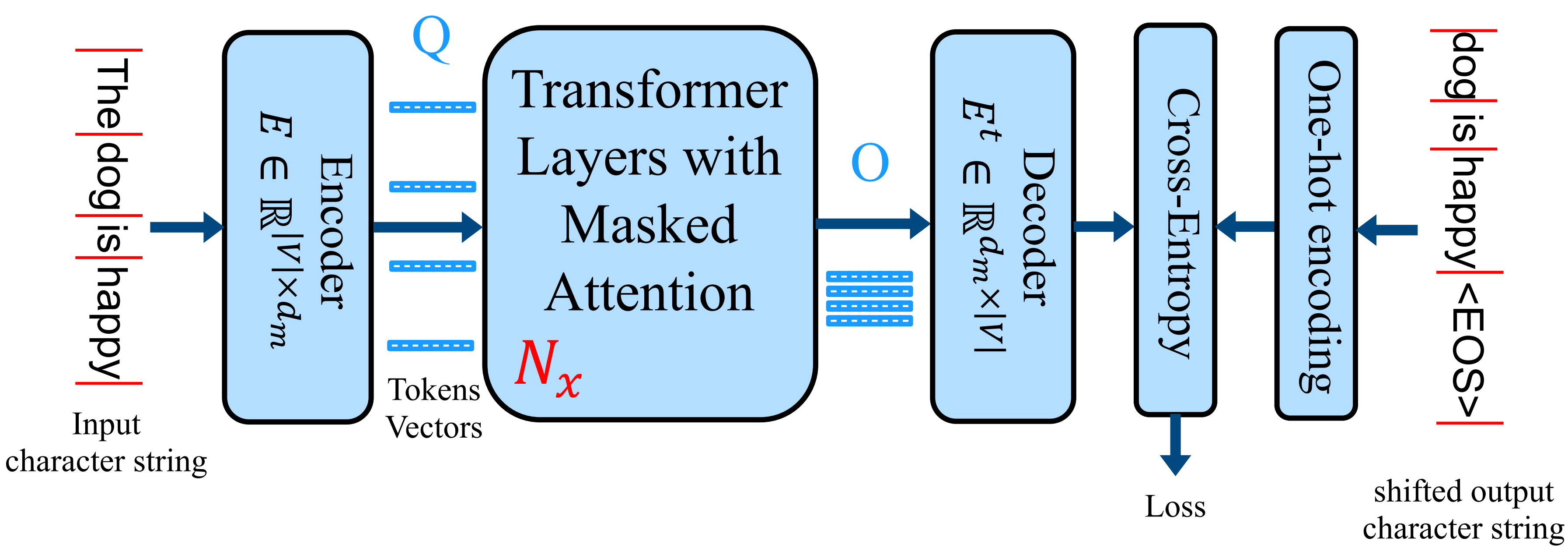
Full LLM Model

- Model
  - Inputs character sequences
  - Output token probabilities
  - Uses masked attention to enforce causal prediction



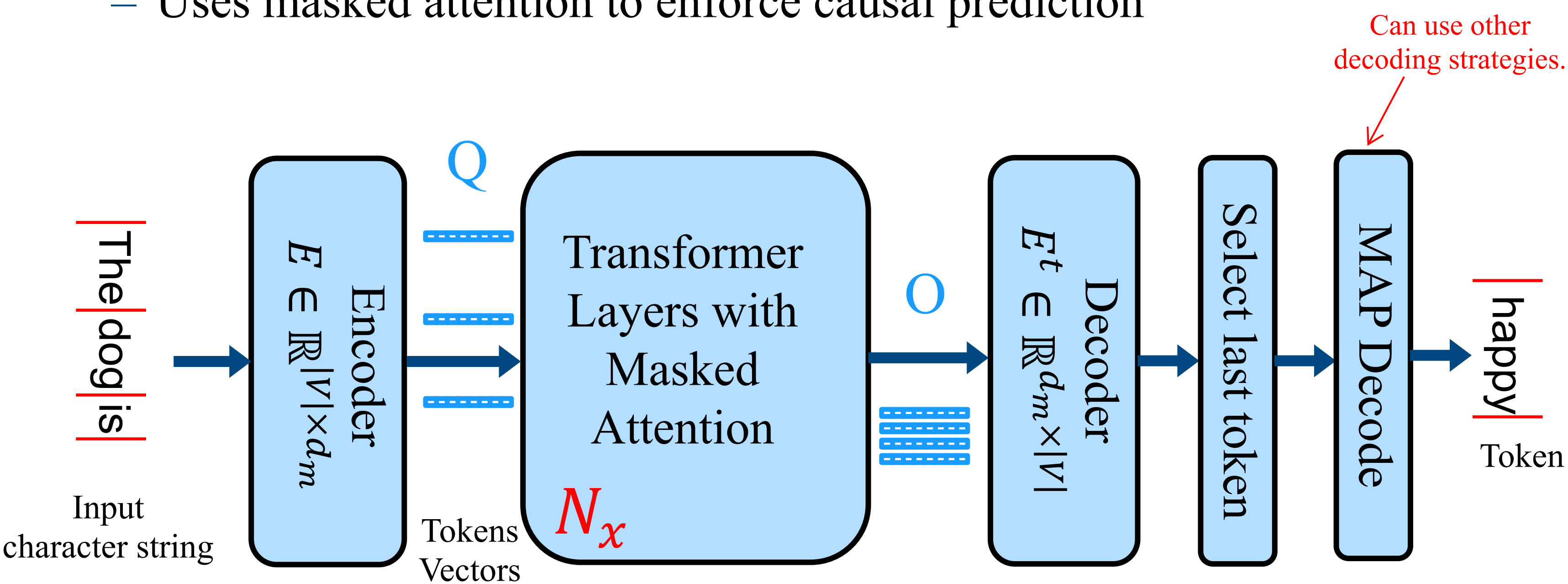
Generative Pre-Training

- Un-label character sequences are used for both input and target sequences
- Target sequence is shifted one token to the left.
- <EOS> tokens are put at the ends of documents.
- Cross entropy loss compares softmax prediction to one-hot encoding



Autoregressive Sequence Generation

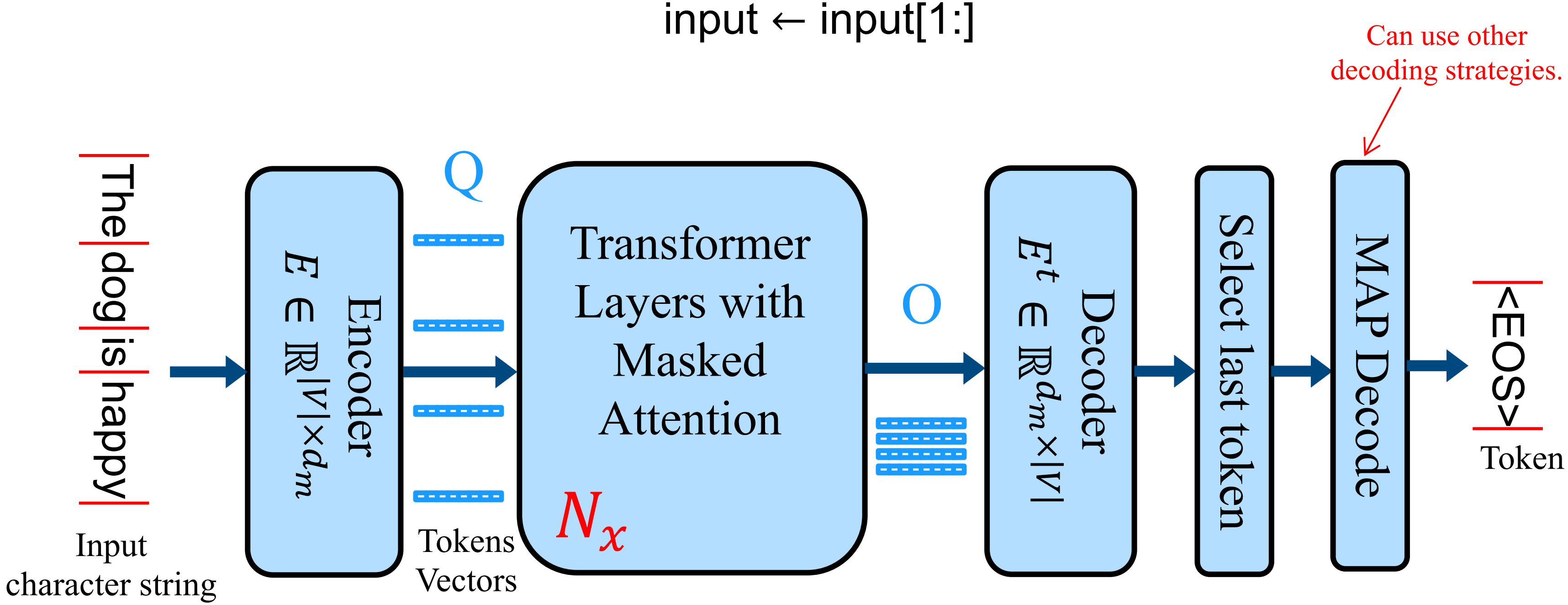
- Model
  - Inputs character sequences
  - Output token probabilities
  - Uses masked attention to enforce causal prediction



Sequence Termination

- LLM algorithm:

Input = <User prompt>
Repeat until output=<EOS>:
    output ← LLM(input)
    input = [input, output]
    If length(input) > context_length
    input ← input[1:]



Embedding

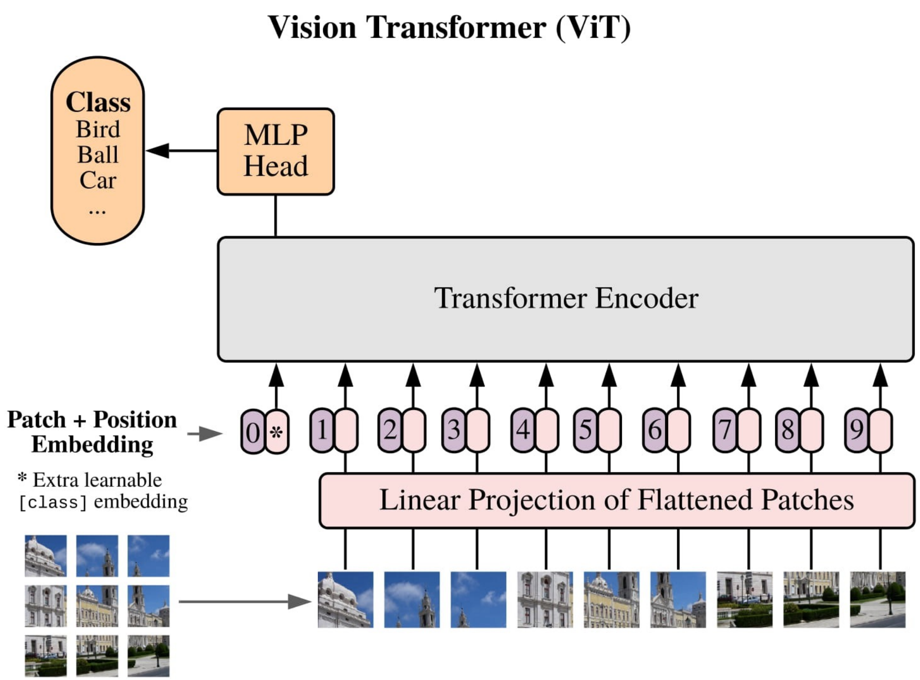
- **Tokenization**
  - Sequencies of symbols are grouped together based on the probability of occurrence into a vocabulary of size |V|=50K.
  - Tutorial: https://sebastianraschka.com/blog/2025/bpe-from-scratch.html
  - Example: https://tiktokenizer.vercel.app/?model=gpt-4
- **Embedding matrix**
  - $E \in \mathbb{R}^{|V| \times d_m}$ is learned during training.
  - The same matrix is also used for decoding.
- **Position encoding**
  - The position encoding is added to the embedded token vector.
  - This encodes the position of each token in the sequence.
- **Special tokens**
  - There is one special token <EOS> that denotes the end of the sequence.
  - Input sequence:
    - The dog is a pet
  - Target sequence:
    - The dog is a pet <EOS>
  - Loss function is difference between softmax probabilities and target sequence using the cross-entropy loss.
- **Context window**
  - Length of the sequences that are used in training.
  - If it is too long, then the computation for the transformer will be too large.

Vision Transformer*

- Single-Head Attention
- Score and Selection
- Multi-Head Attention
- Self Attention
- Masked Multi-Head Attention

Vision Transformer:
Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby, "An Image is Worth 16*16 Words: Transformers for Image Recognition at Scale," ICLR 2021, https://arxiv.org/pdf/2010.11929 .

ViT Model for Image Classification

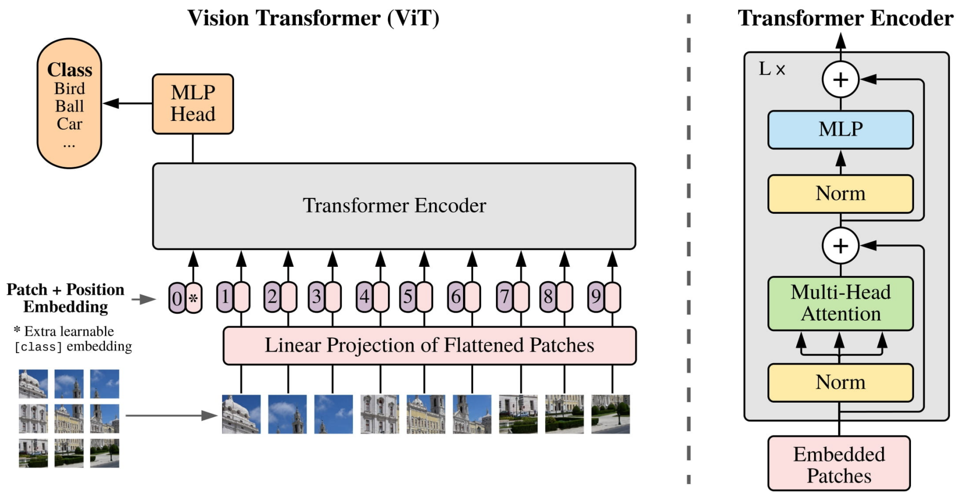


- ❑ MLP head
- ❑ Transformer Encoder
- ❑ Input embedding

* Image source: Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." *arXiv preprint arXiv:2010.11929* (2020).

31

MLP Head and Transformer Encoder



- ✓ MLP head
- ✓ Transformer Encoder
- ❑ Input embedding

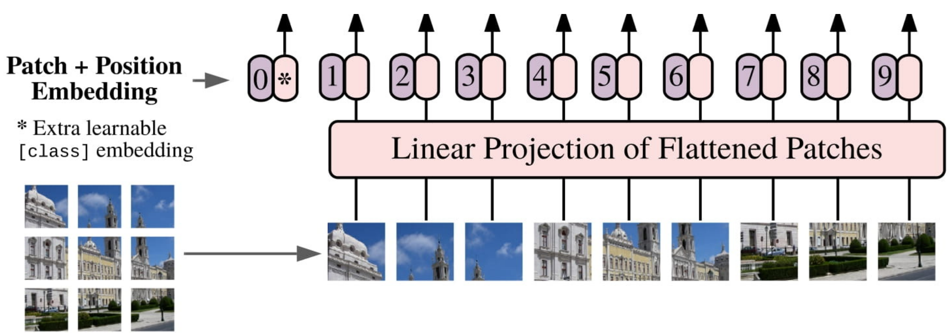
* Image source: Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." *arXiv preprint arXiv:2010.11929* (2020).

32

Input Embedding

To convert the image into a format that the model can process

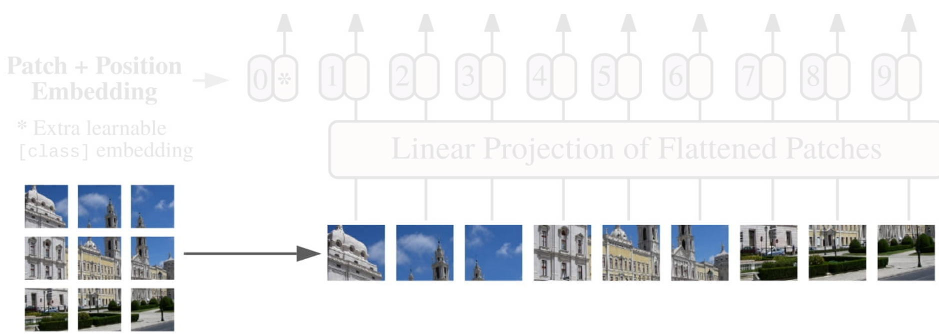
- Patch splitting
- Patch flattening and linear projection
- Positional encoding
- Class token (optional)



33

Patch Splitting

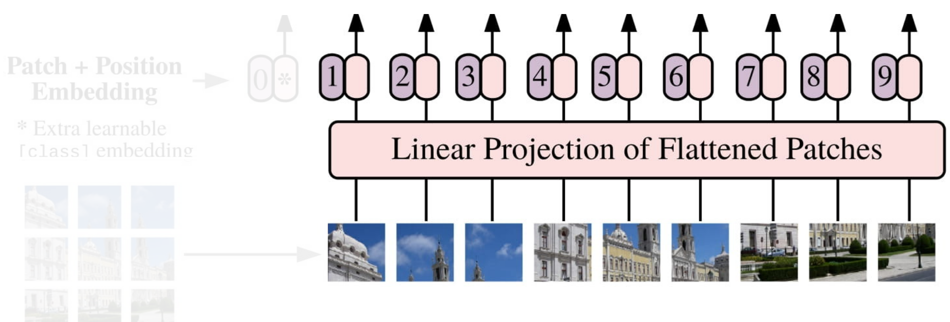
- Split input image into fixed-size non-overlapping patches
  1) Reshape image to 224x224
  2) Divide image input patches of size 16x16



34

Patch Flattening and Linear Projection

- Transform each patch into a fixed-size embedding vector
  1) Flatten each patch into a 1D vector
  2) Pass through a linear projection



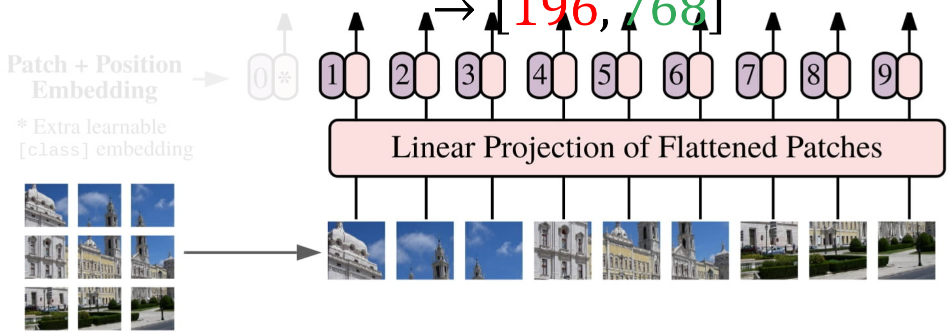
35

Dimensionality of Tokens

$$Image\ data \Rightarrow [{\color{red}num\_token}, {\color{green}token\_dim}]$$

$$[224, 224, 3] \rightarrow [{\color{red}14, 14}, {\color{green}768}]$$

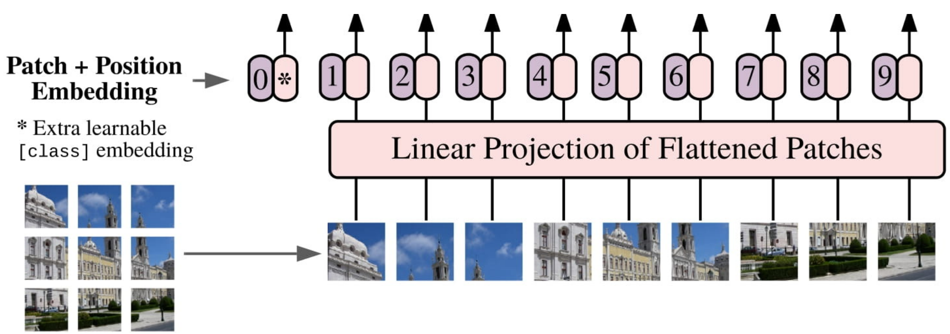
$$\rightarrow [{\color{red}196}, {\color{green}768}]$$



36

Positional Encoding

- Provide position information of patches
    1) A set of learned vectors for each patch location
    2) Add to the patch embeddings



37

Class token (optional)

- Provide position information of patches
  1) Randomly initialized
  2) Prepended to the beginning of input sequence

$$concat([1, 768], [196, 768]) \rightarrow [197, 768]$$

num_token token_dim