INF721

2024/2

UFV

Deep Learning

L19: Multimodal Learning

Logistics

Announcements

- Midterm Exam II has been pushed back to Nov 27th

Last Lecture

- GPT
- BERT

Lecture Outline

- Multimodal Learning
- Vision Transformers (ViT)
- Audio Data
- Text-to-Speech
- Text-to-Image
- CLIP (Contrastive Language-Image Pre-training)

Multimodal Learning

Transformers make very few assumptions about input data, so they have become state-of-the-art in many different modalities:

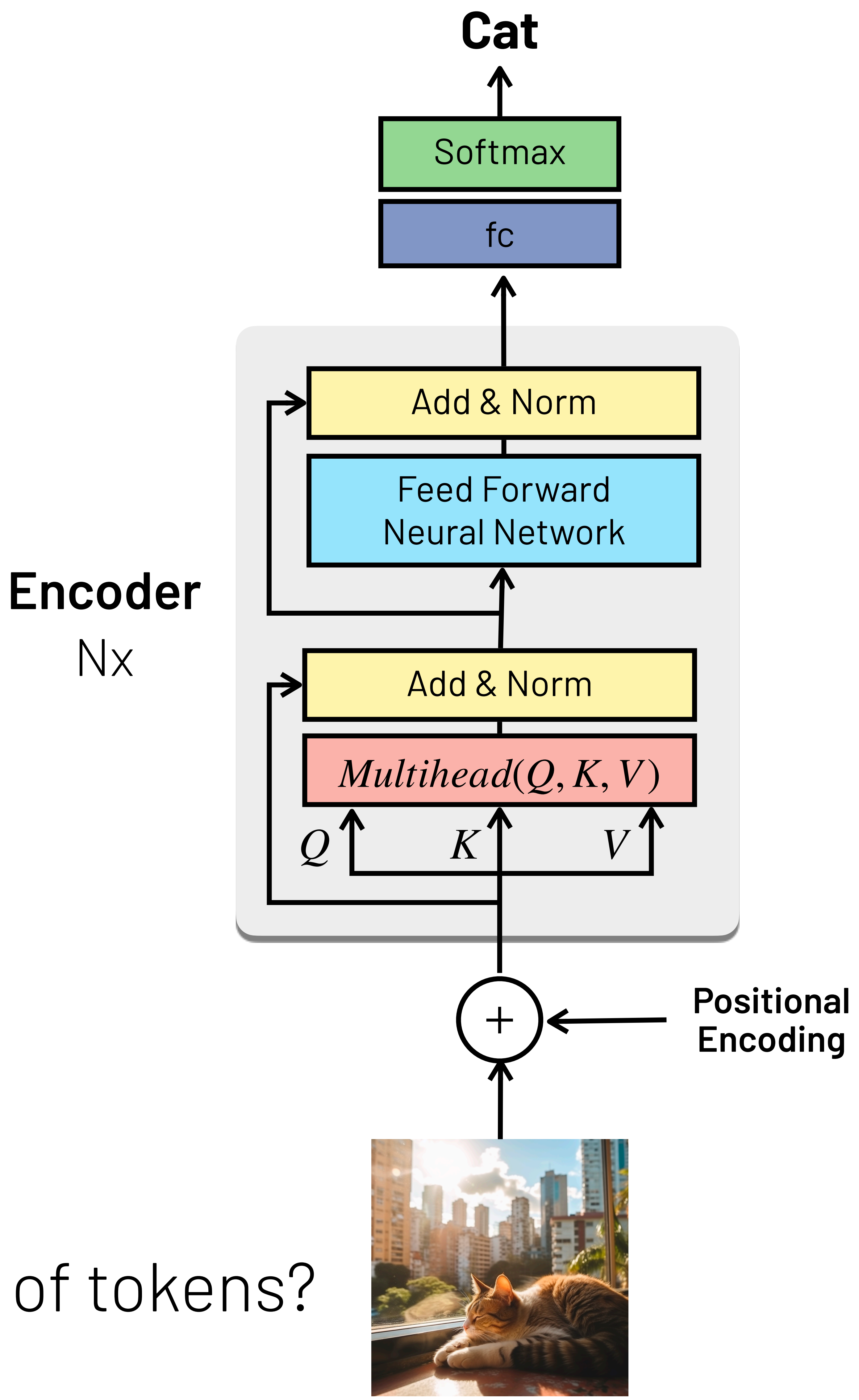
- Text, Image , Video, Audio…

The core architecture has remained relatively constant across aplications, what has changed is the **representation** and **encoding** of inputs and outputs

Vision Transformers

Vision transformers typically use an **transformer encoder** for image classification tasks

The overall architeture remains the same:



Main problems:

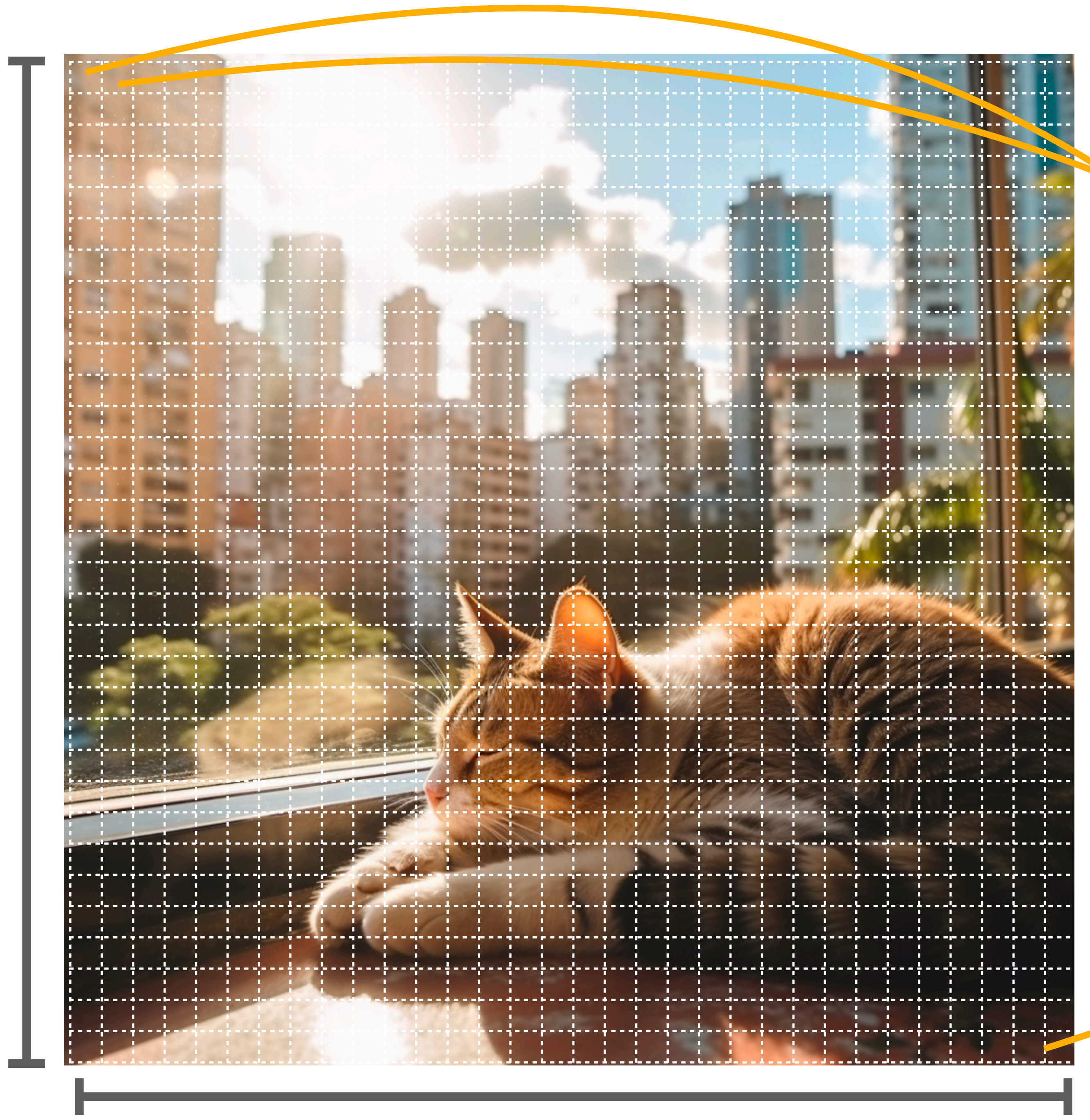
2. How to encode positional information?

1. How to transform an image into a sequence of tokens?

Images as Sequences (Classification): Naive

The simplest approach to represent images as sequences is to consider every pixel as an element:

h



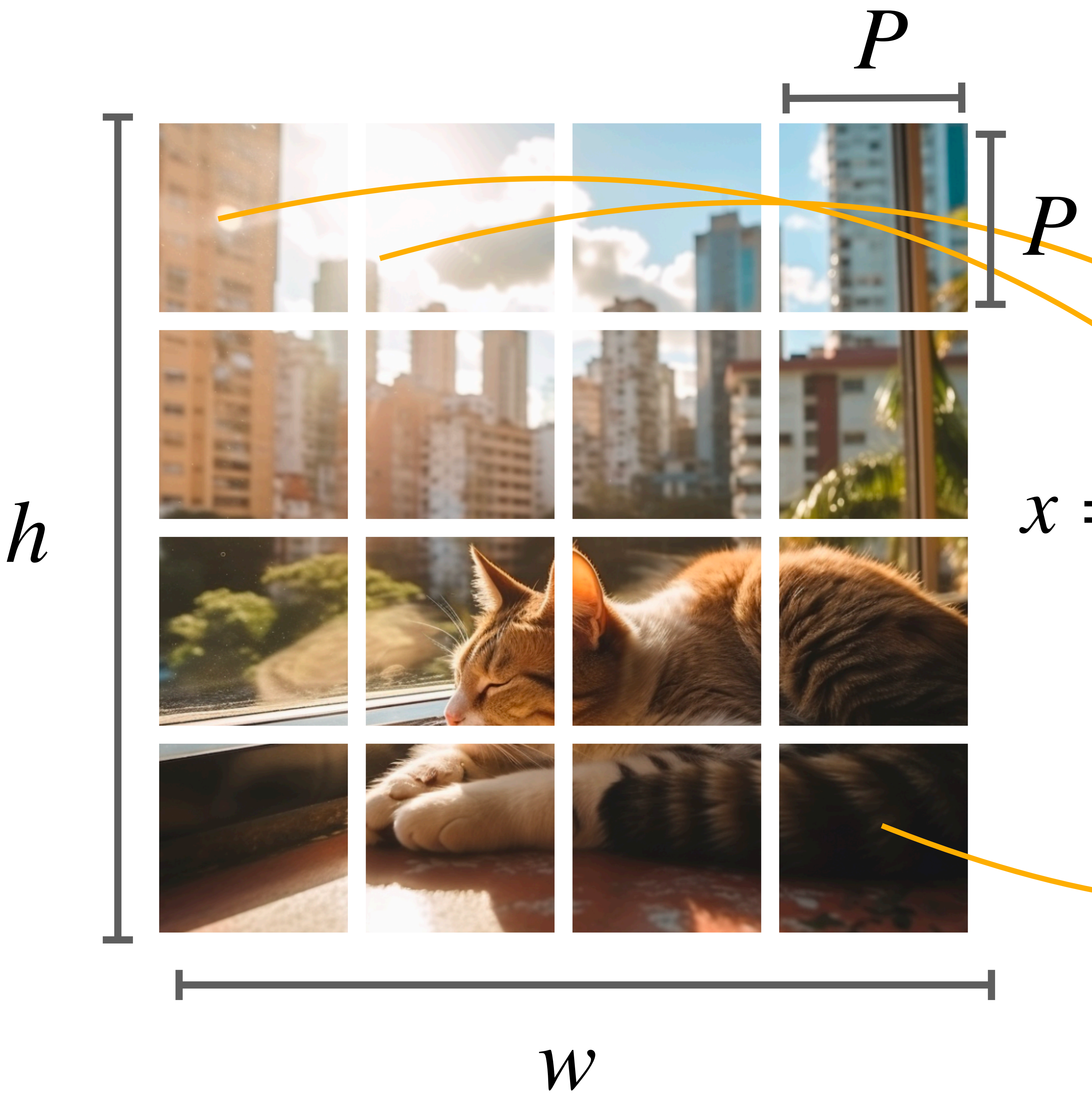
w

$$x = \{x^{<1>}, x^{<2>}, \ldots, x^{<n>}\}$$

The problem is that the memory required by the transformer grows quadratically $O(n^2)$ with the number of input tokens $n = wh$

Images as Sequences (Classification): Patches

The most common approach is to split the input image into a sequence of non-overlapping patches of size $P \times P$

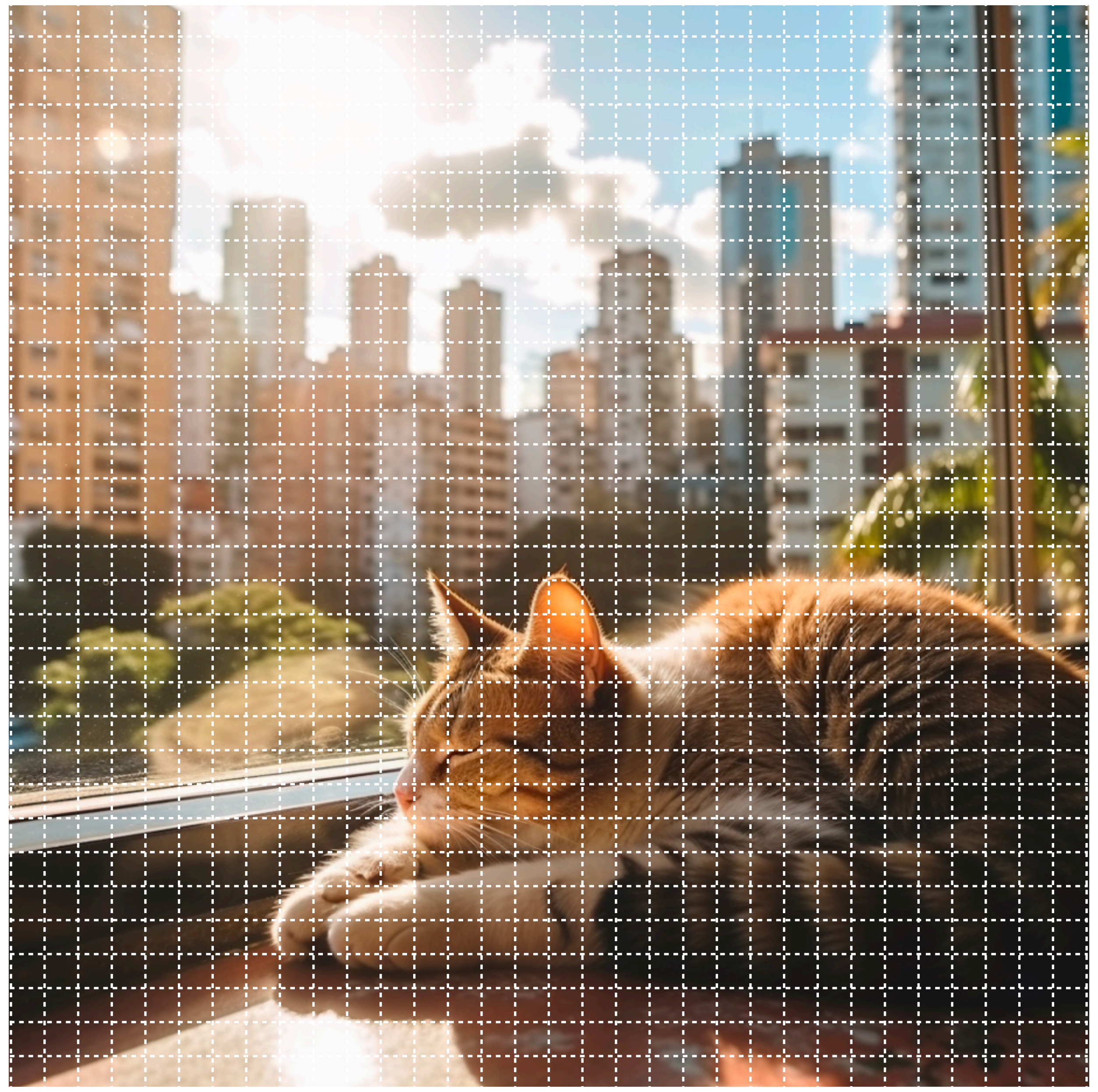


$$x = \{x^{<1>}, x^{<2>}, \ldots, x^{<m>}\}$$

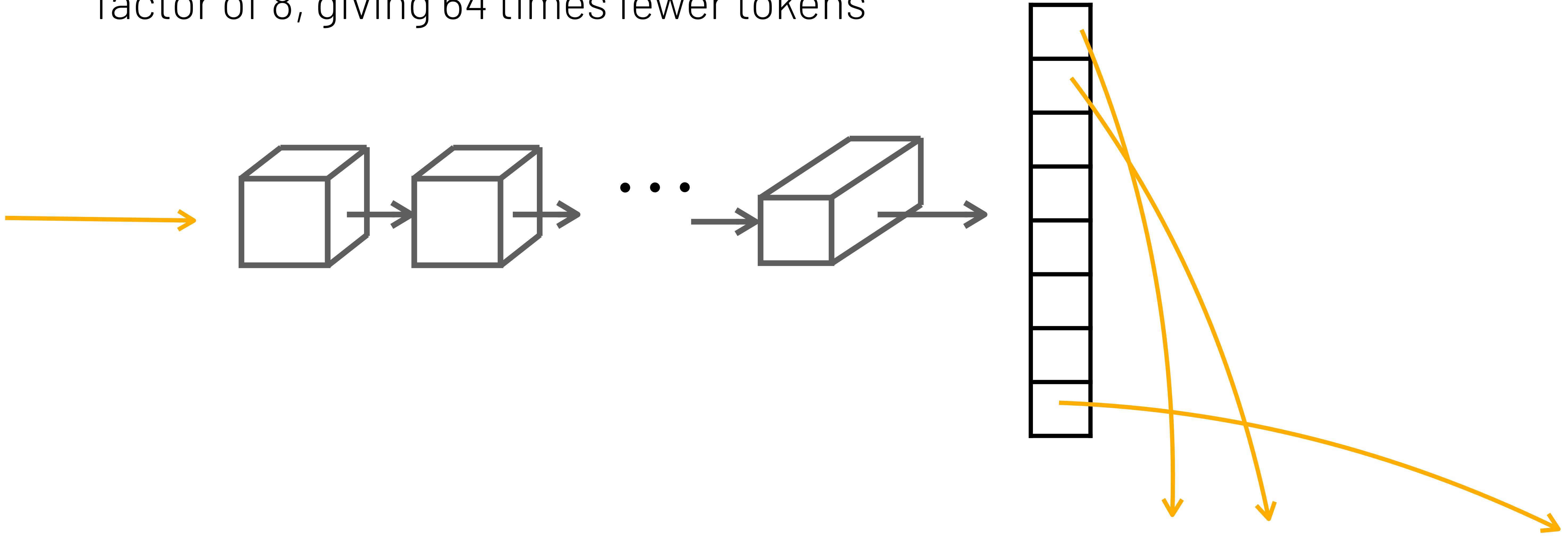
This approach reduces the size of the sequence to $m = \dfrac{hw}{P^2}$

Images as Sequences (Classification): CNN

Another approach to transform imagens in sequences is to use a Convolutional Neural Network (CNN) to extract a feature vector from the image:



ResNet18 would downsample an image by a factor of 8, giving 64 times fewer tokens



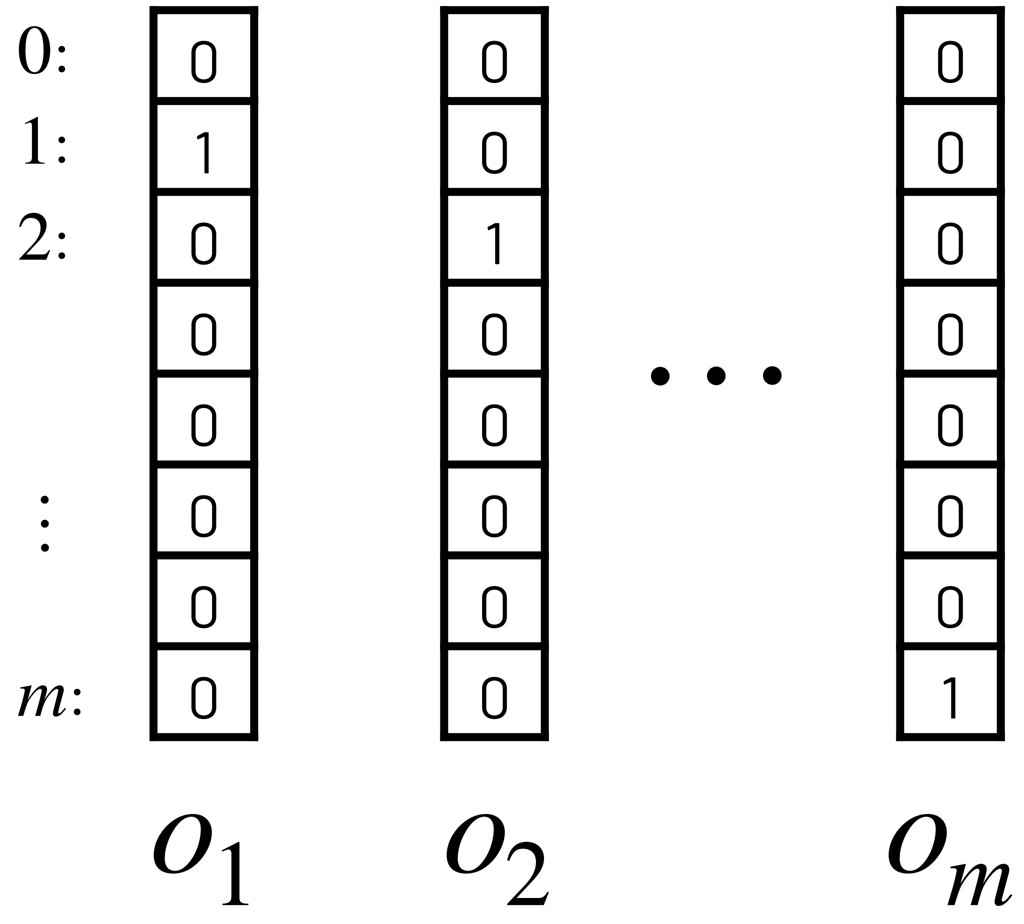
$$x = \{x^{<1>}, x^{<2>}, \ldots, x^{<m>}\}$$

Positional Information (Classification)

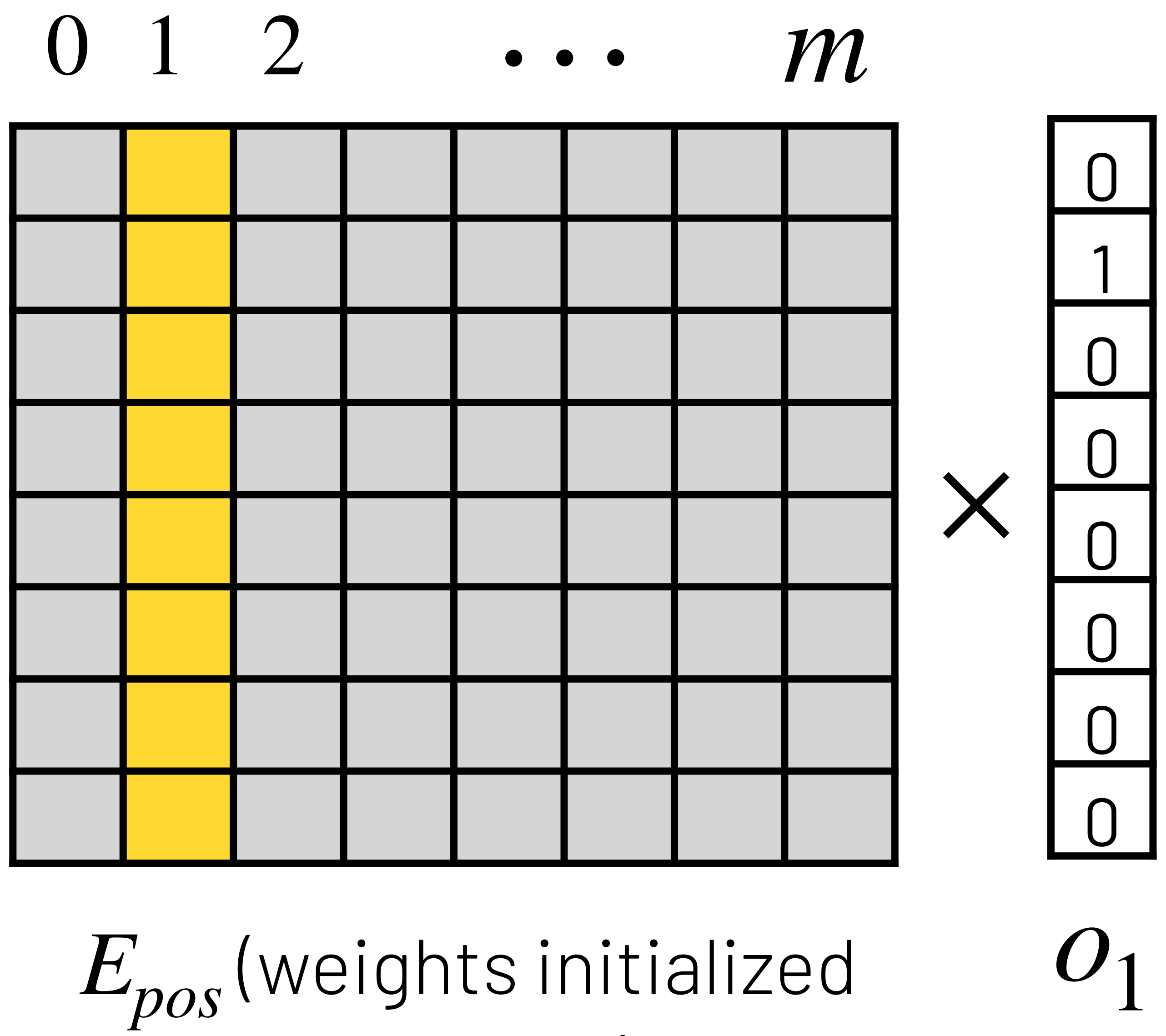
It is possible to define explicit positional encodding to a sequence of patches, but the most common approach is to **learn** these positional embeddings:

1. Create one hot encoding based on position indices:

$$x = \{x^{<1>}, x^{<2>}, \ldots, x^{<m>}\}$$

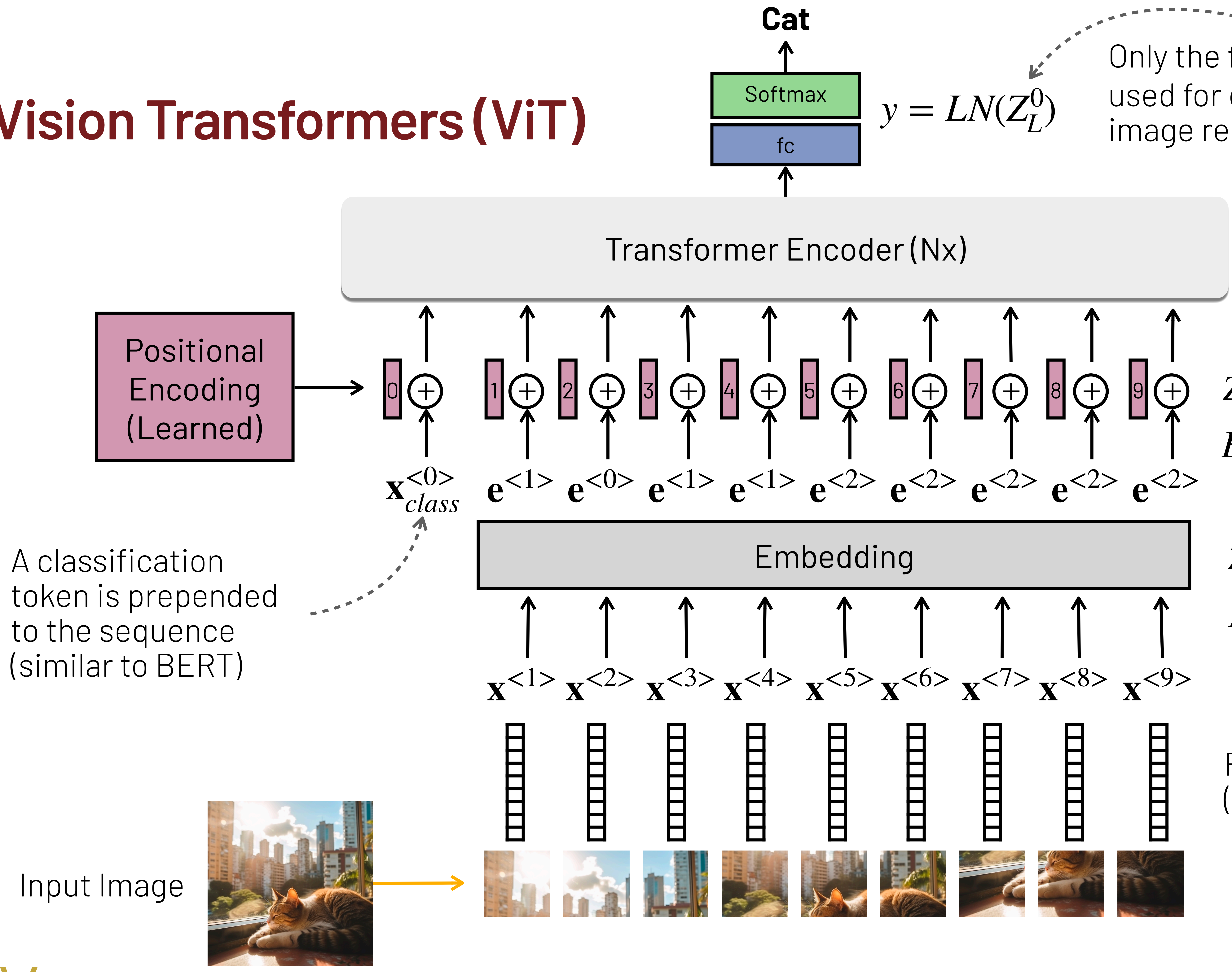


2. Multiply as an embedding matrix:



E_{pos} (weights initialized randomly)

Vision Transformers (ViT)



Only the first element Z_L^0 of the last layer is used for classication. It serves as the entire image representation.

$$Z_l = MLP(LN(Z'_{l-1})) + Z'_{l-1}$$

$$Z'_l = Multihead(LN(Z_{l-1})) + Z_{l-1}$$

$$Z_1 = [\mathbf{x}^{<0>}_{class}, \mathbf{e}^{<1>}, \mathbf{e}^{<2>}, \ldots, \mathbf{e}^{<m>}] + E_{pos}$$

$$E_{pos} \in \mathbb{R}^{(m+1)\times D}$$

$$Z_0 = [E\mathbf{x}^{<1>}, E\mathbf{x}^{<2>}, \ldots, E\mathbf{x}^{<m>}]$$

$$E \in \mathbb{R}^{(P^2C)\times D}$$

Flatten Patches (Input vectors)

Dosovitskiy et al. 2021, An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale

Generative Image Transformers

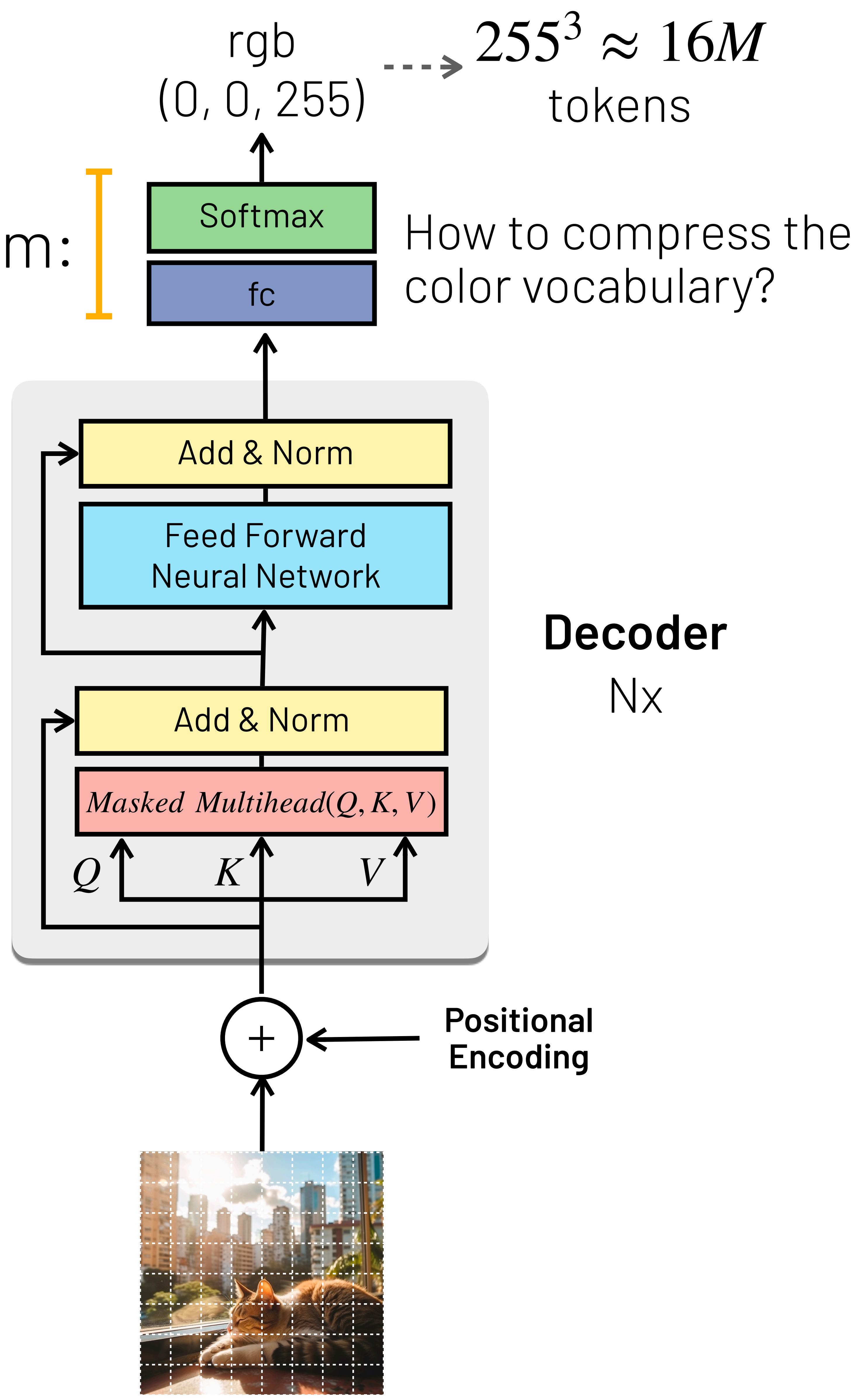
Vision transformers typically use an **transformer decoder** for image generation tasks

Main problem:

The overall architeture remains the same:

Explicit positional embedding (sinusoidal)

Represent image as a sequence of pixels or patches

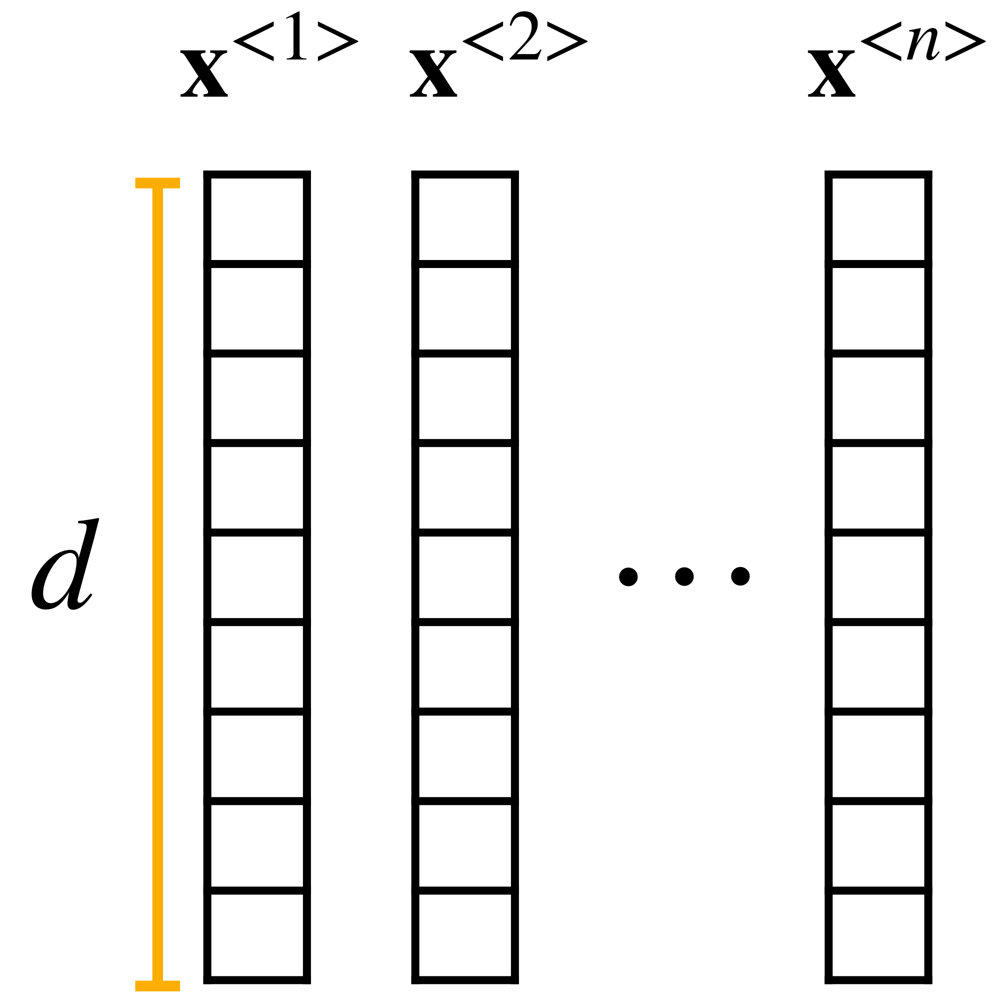


UFV

Vector Quantization

One way to address the problem of the high color vocabulary dimensionality is to use a technique called **Vector Quantization**, which can be viewed as a form of **data compression**.

n data vectors
(e.g., image pixels)

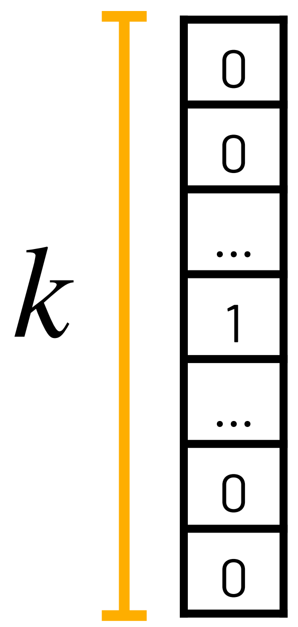


1. Approximate $\mathbf{x}^{<i>}$ by its nearest $\mathbf{c}^{<j>}$ according to some similarity metric

$$\mathbf{x}^{<i>} \rightarrow \text{argmin}_{\mathbf{c}^{<j>}} ||\mathbf{x}^{<i>} - \mathbf{c}^{<j>}||^2$$

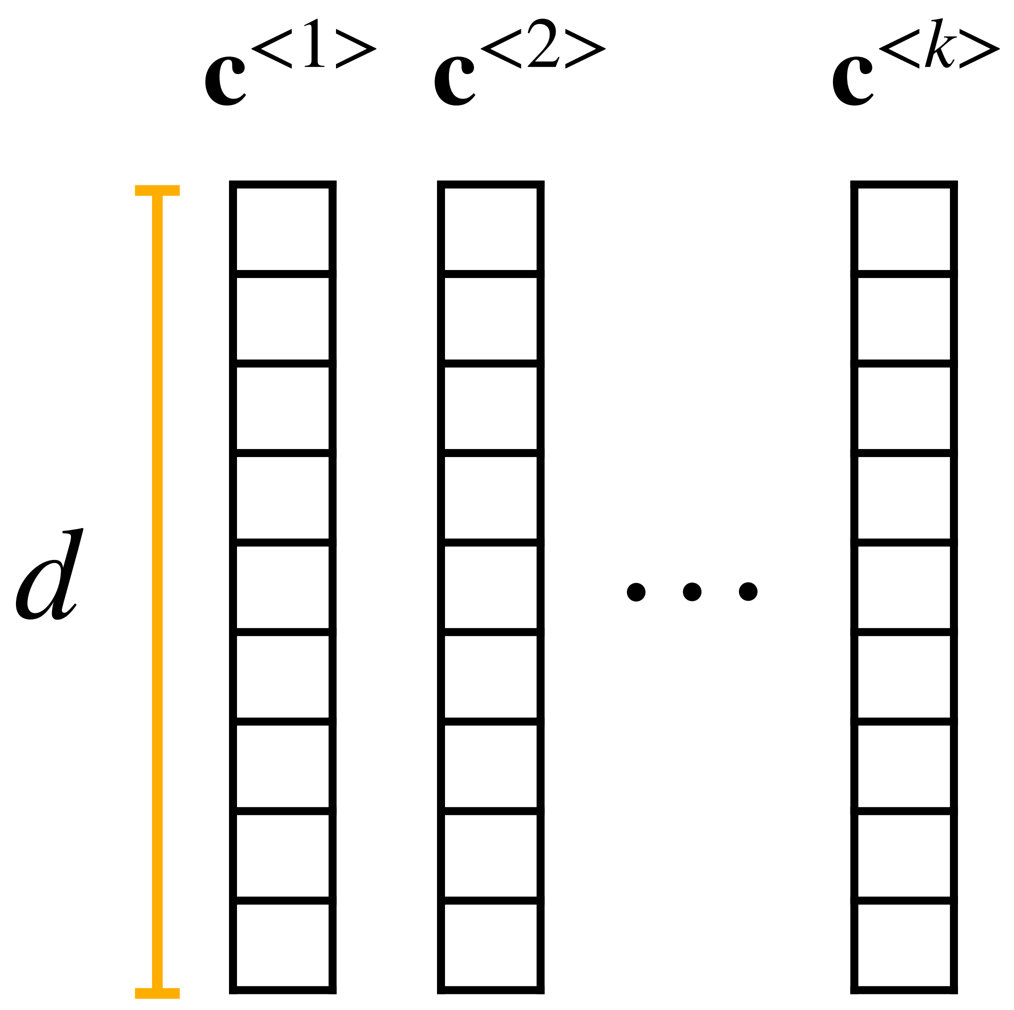
(usually Euclidian distance)

2. Represent each $\mathbf{x}^{<i>}$ as one-hot vector with k dimensions



Use this one-hot vector to retrieve the codebook vector

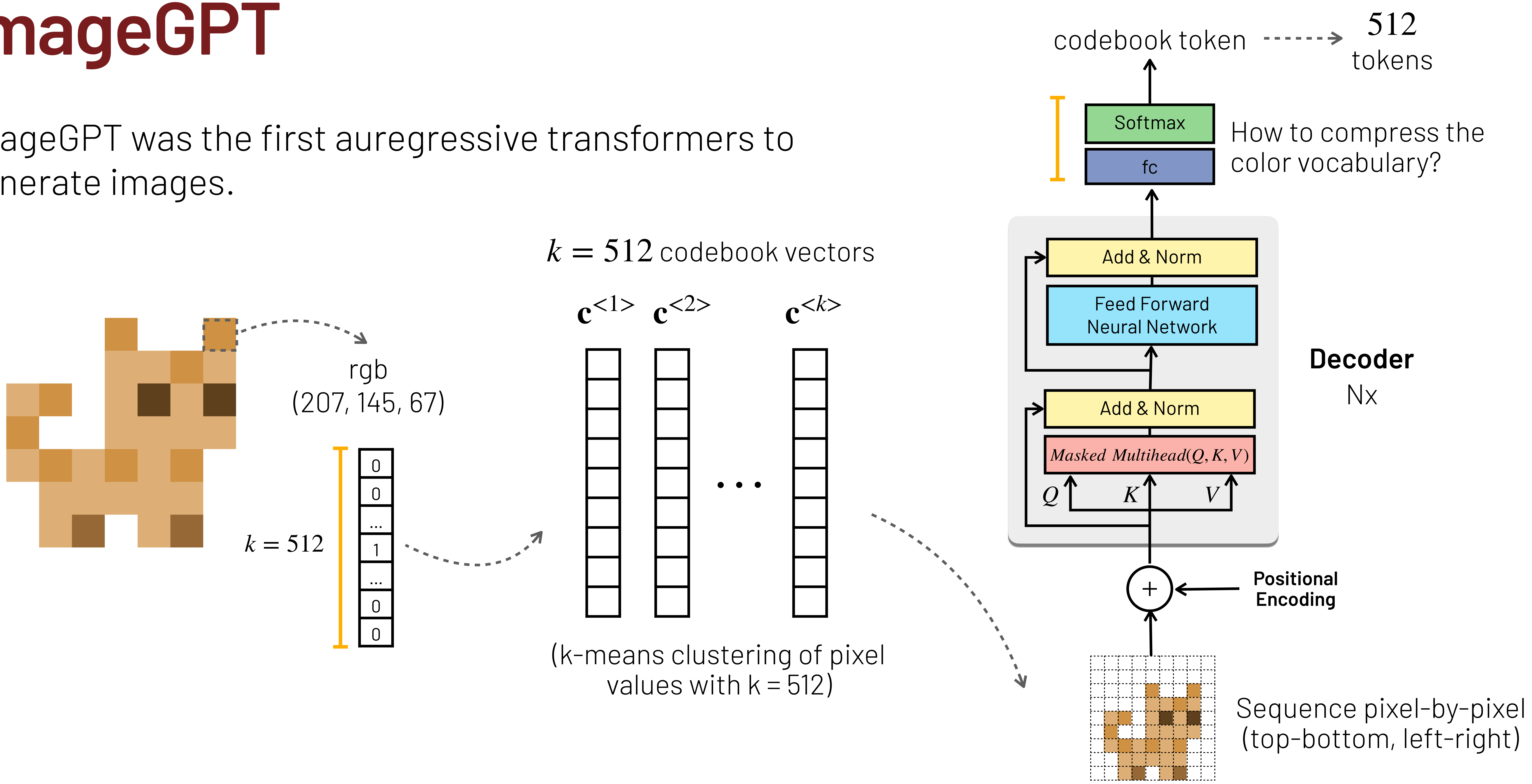
k codebook vectors $k \ll d$



UFV

ImageGPT

ImageGPT was the first auregressive transformers to generate images.

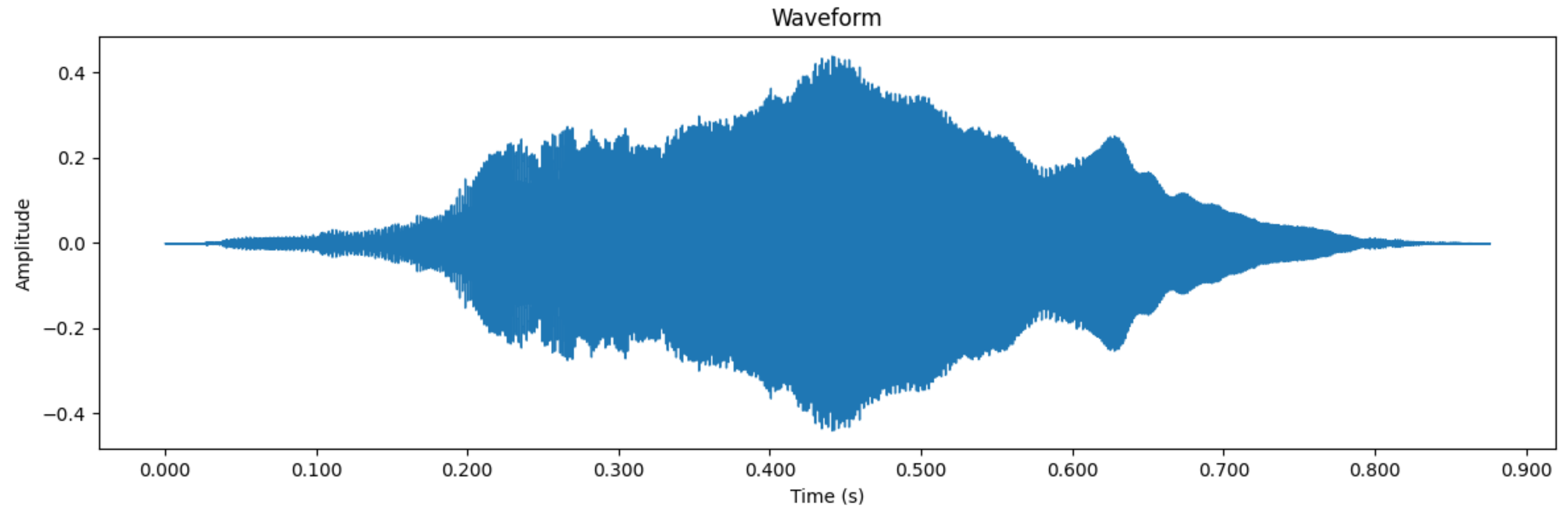


Chen et al. 2020, Generative Pretraining from Pixels

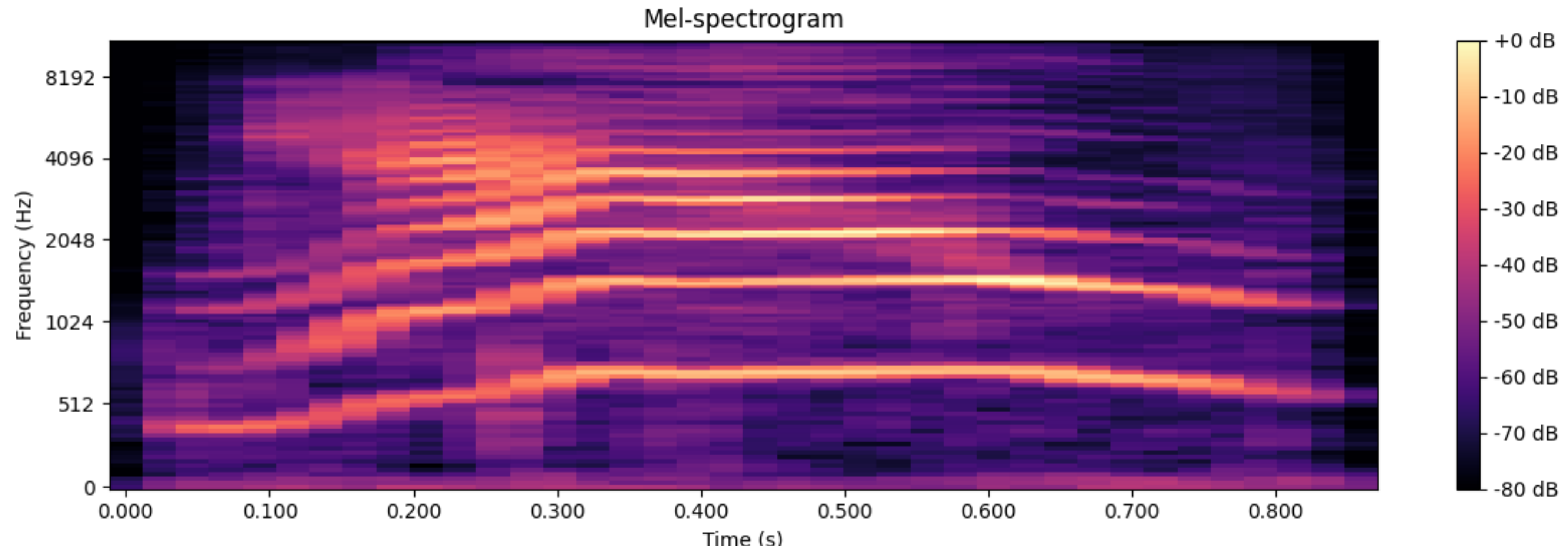
Audio Classification

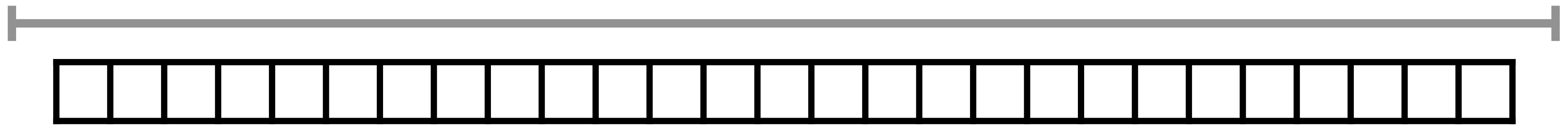
Audio signals are represented as wafeforms, which are measures of amplitude of the air pressure at regular time intervals.

Audio signals are represented as wafeforms, which are measures of amplitude of the air pressure at regular time intervals.



STFT





~44K tokens

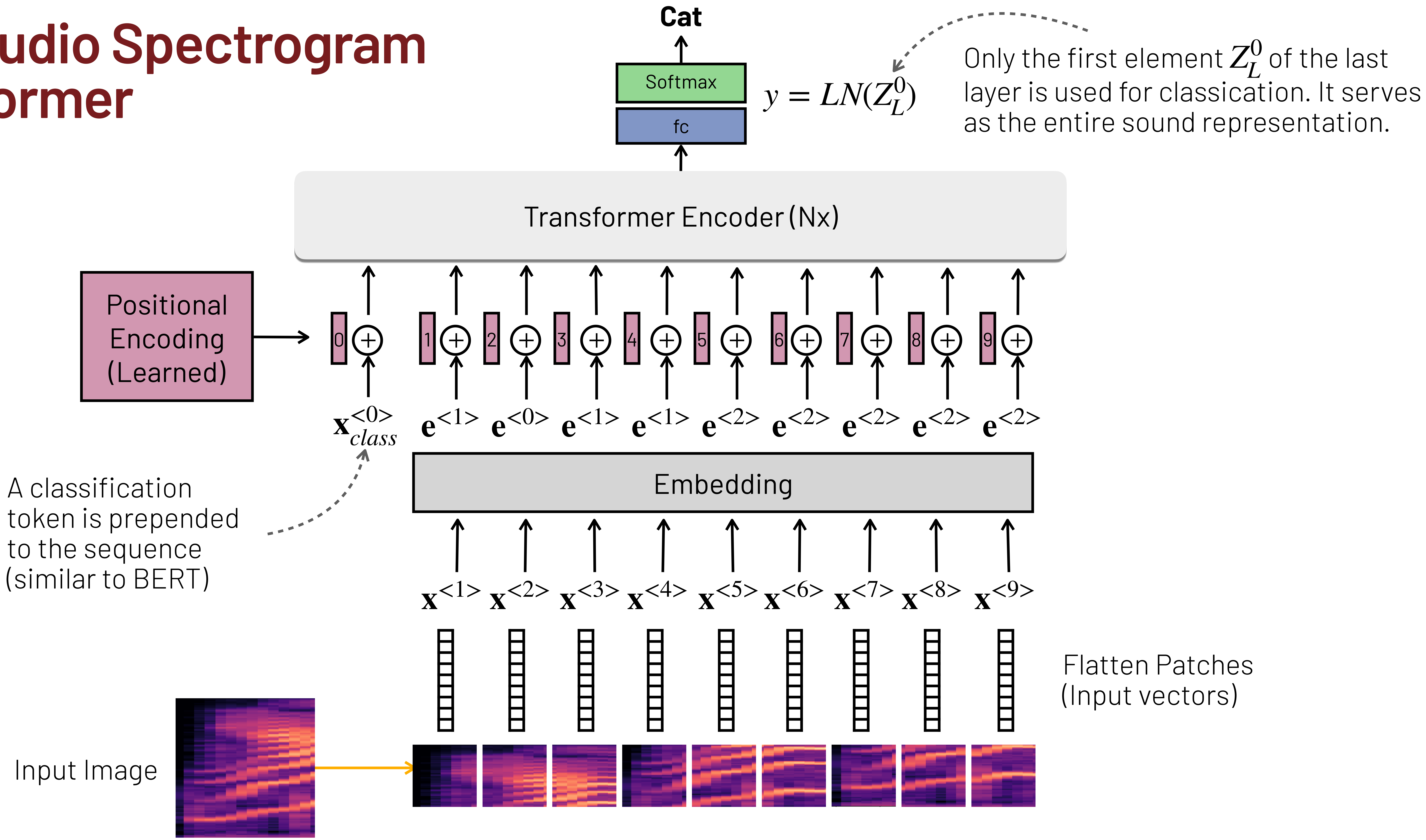
(Short time Fourier Transform)

Waveform

Mel-spectrogram

UFV

AST: Audio Spectrogram Transformer



Multimodal Learning

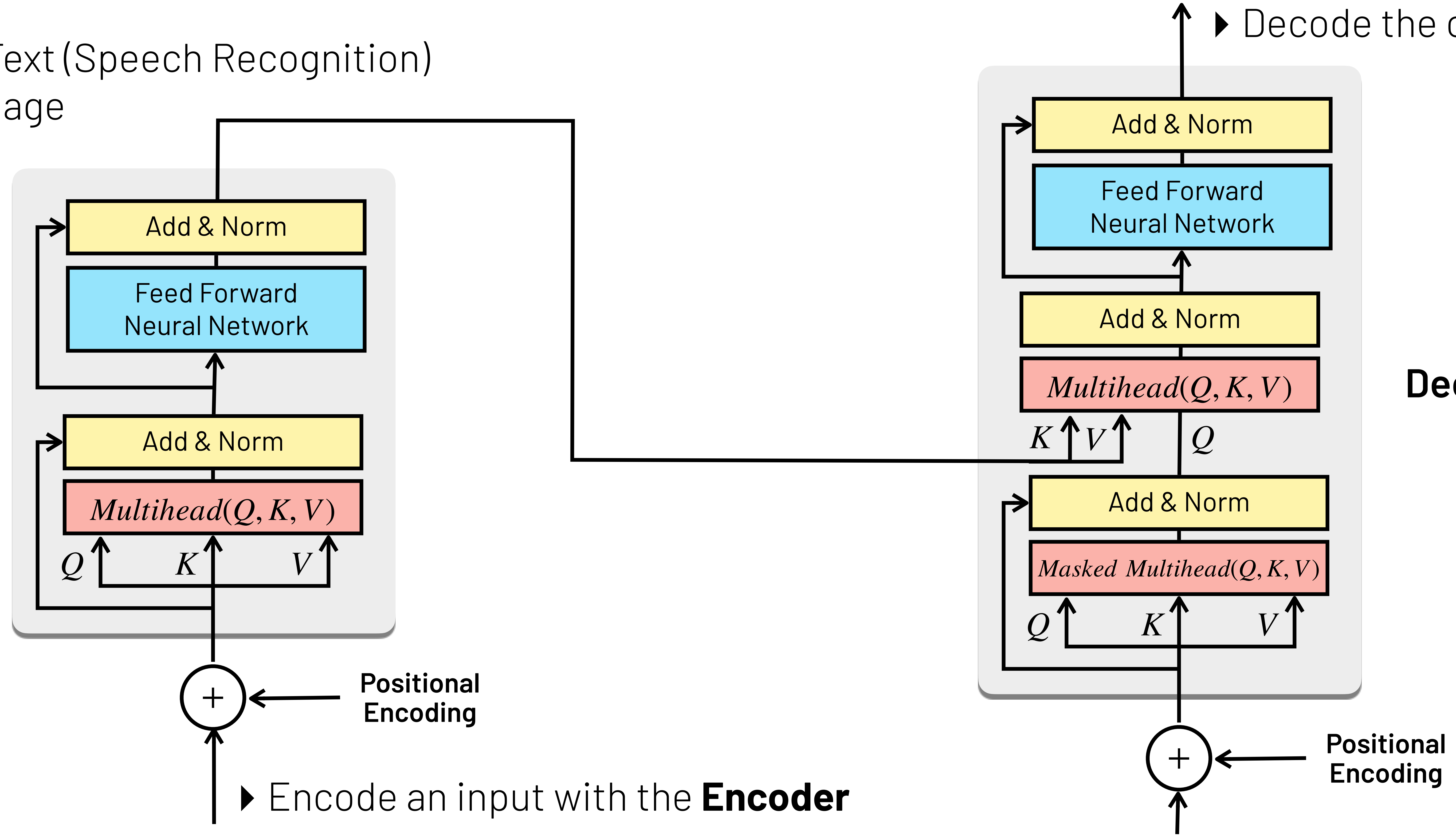
Since transformers can process many different kinds of data, we can create multimodal models relatively straightforward with a "machine translation" framework

Examples:

- Audio-to-Text (Speech Recognition)
- Text-to-Image

- Decode the output with the **Decoder**

Encoder
Nx

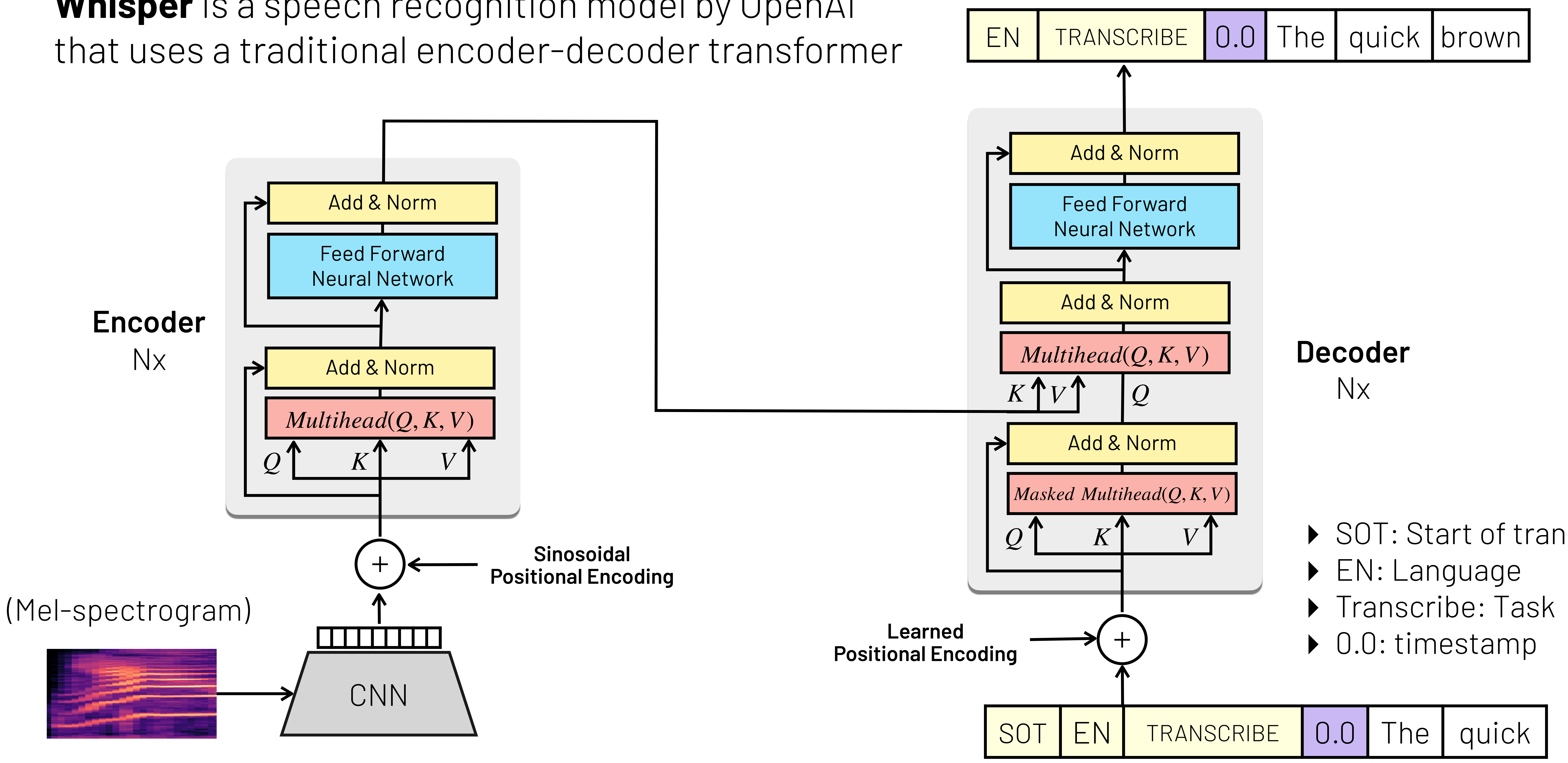


Decoder
Nx

- Encode an input with the **Encoder**

Whisper: Audio-to-Text (Speech Recognition)

Whisper is a speech recognition model by OpenAI that uses a traditional encoder-decoder transformer

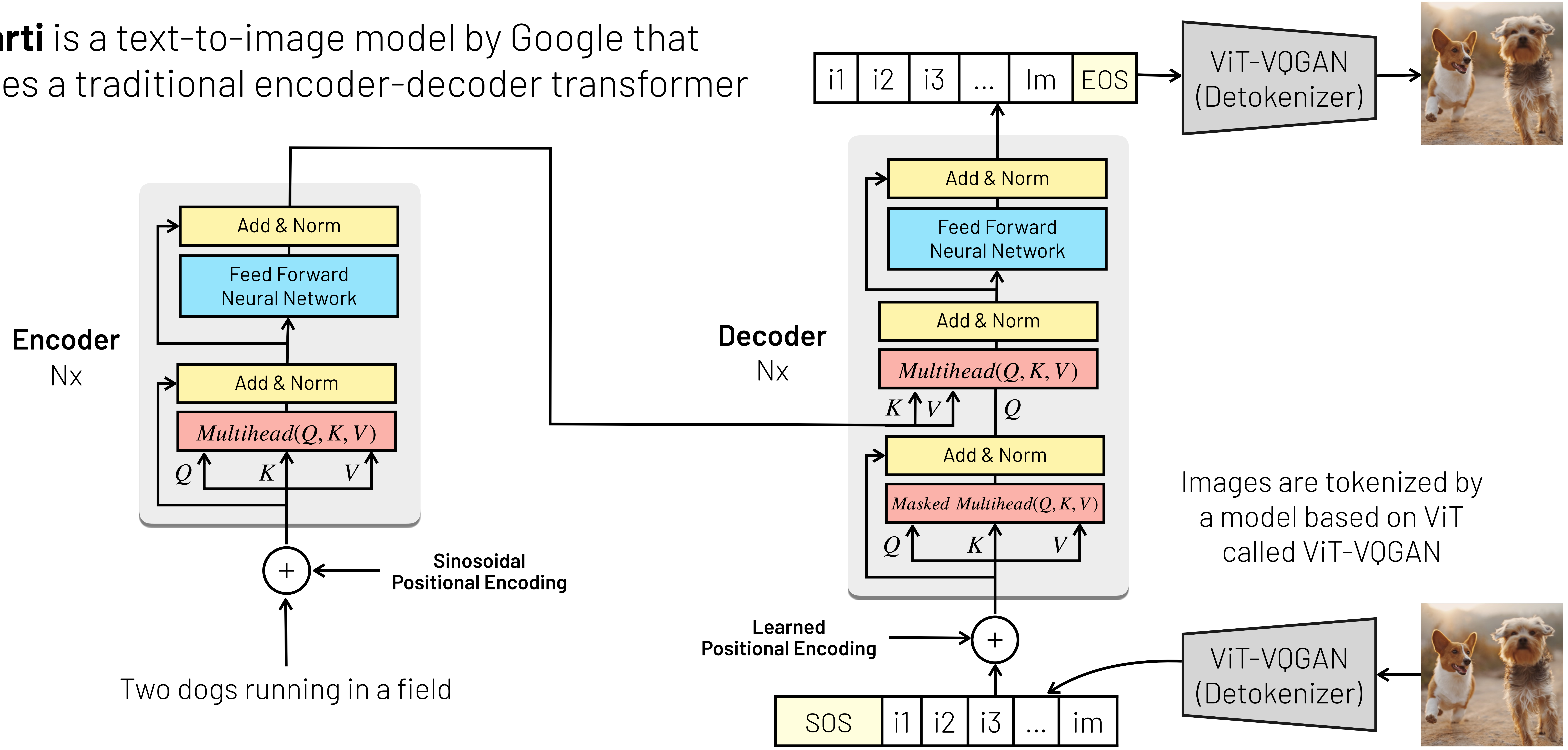


- SOT: Start of transcript
- EN: Language
- Transcribe: Task
- 0.0: timestamp

Radford et al. 2022, Robust Speech Recognition via Large-Scale Weak Supervision

Parti (Pathways Autoregressive Text-to-Image)

Parti is a text-to-image model by Google that uses a traditional encoder-decoder transformer

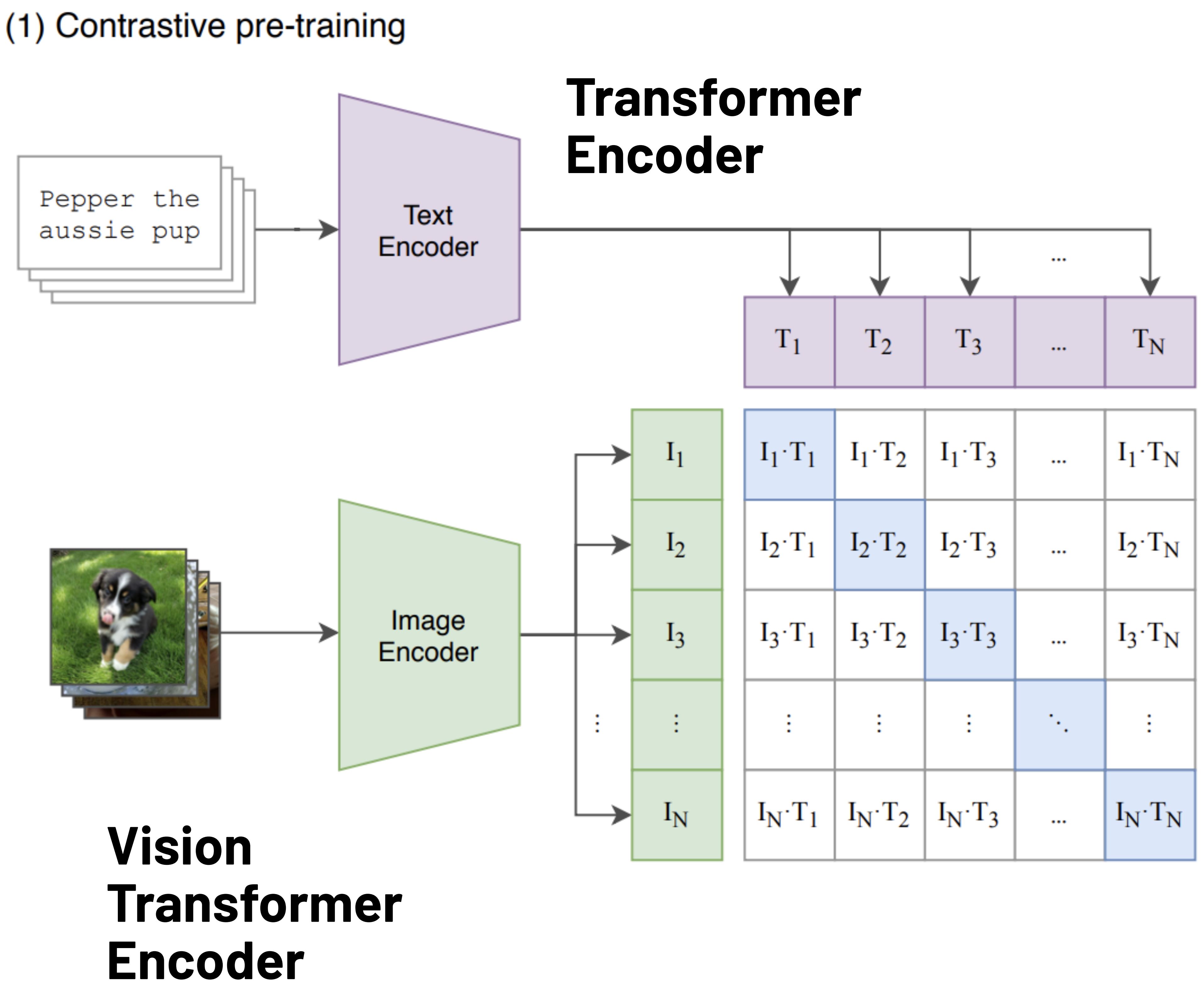


Images are tokenized by a model based on ViT called ViT-VQGAN

Yu et al. 2022, Scaling Autoregressive Models for Content-Rich Text-to-Image Generation

CLIP (Contrastive Language-Image Pre-training)

CLIP is a model developed by OpenAI to learn embeddings for Image and text in the same vector space and can be directly compared



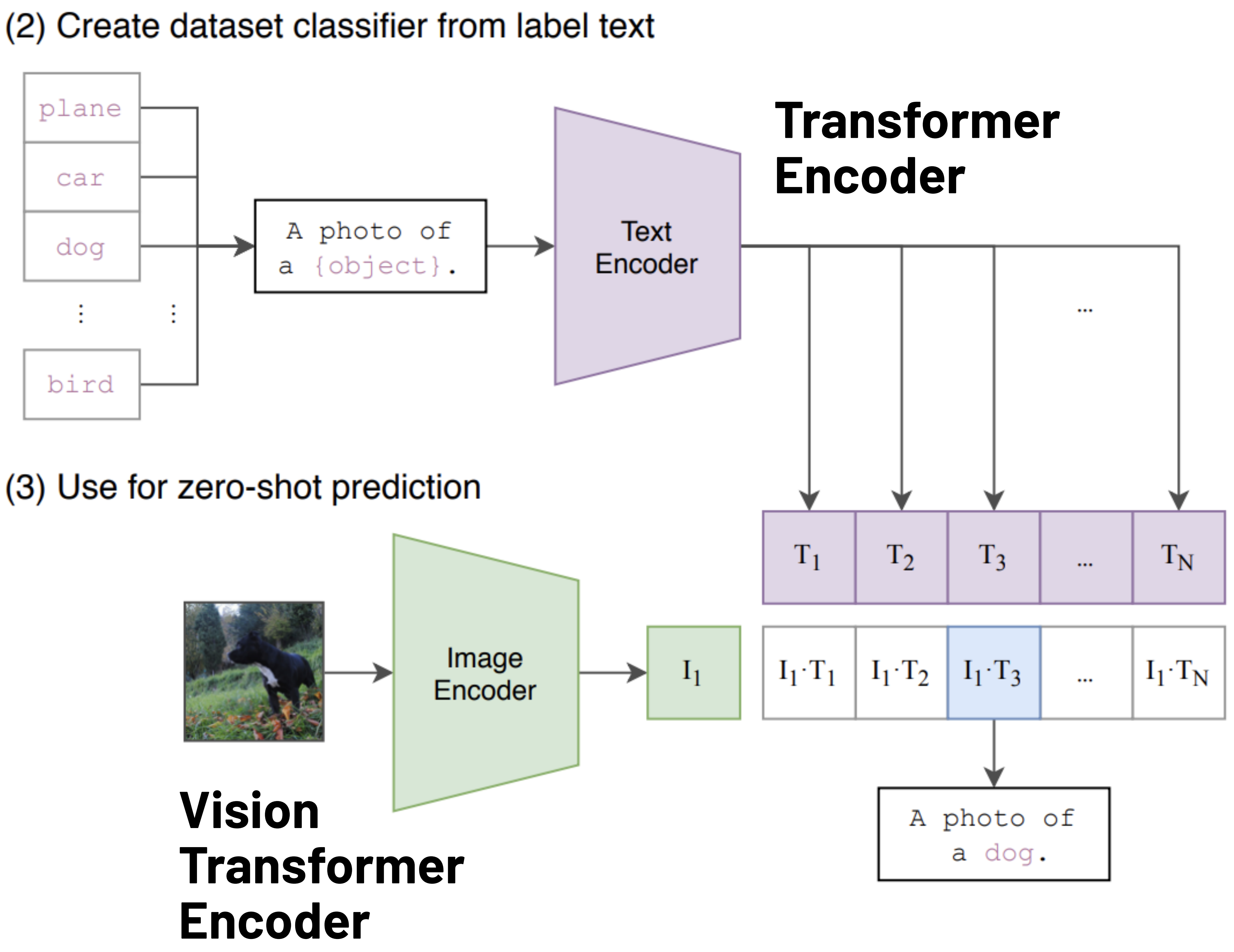
Training

1. Takes batch of (image, text) pairs
2. Passes images through vision encoder
3. Passes captions through text encoder
4. Maximizes similarity between correct image-text pairs
5. Minimizes similarity between incorrect pairs using contrastive loss

Radford et al., J. 2021, Learning Transferable Visual Models From Natural Language Supervision

CLIP (Contrastive Language-Image Pre-training)

CLIP is a model developed by OpenAI to learn embeddings for Image and text in the same vector space and can be directly compared



Inference

1. Can measure similarity between any image and text embedding
2. Enables zero-shot classification by comparing image embeddings to text embeddings of class names
3. Supports open-ended image-text matching tasks

Next Lecture

L20: GANs

Generating images with Generative Adversarial Networks (GANs)