THE UNIVERSITY OF MELBOURNE

Vision Transformers

Semester 2, 2025

Kris Ehinger

Outline

- Inductive bias of CNNs
- Vision Transformer
- Transformers vs. CNNs

Learning outcomes

- Define inductive bias and identify inductive biases in computer vision algorithms
- Explain the basic vision transformer (ViT) architecture and the attention mechanism
- Explain the main differences between CNN and ViT approaches to image recognition

Inductive bias of CNNs

Inductive bias

- Most computer vision tasks have multiple possible solutions
  - Curse of dimensionality: images are large, often fewer training samples than dimensions
- Most methods have relied on inductive bias
- Inductive bias = assumption / restriction that causes a machine learning algorithm to pick one solution over another
- What are the inductive biases of a CNN?

Inductive biases of CNNs

- Locality: the important features emerge from adjacent or nearby pixels (e.g., edges)
  - Features combine to make more complex features, but each level of the hierarchy has this local property
- Translation invariance (equivariance) – the model response should be the same no matter where in the image the feature/object is detected

Vision Transformer

Image as "words"

- Computer vision has a long history of importing models from NLP (e.g., "bag of words")
- Problem: Sentences break down into meaningful parts (words). What are the meaningful parts of an image?
- How do you split an image into "words"?

Bag of words (NLP)

https://www.wordclouds.com/

Bag of features (vision)

- Based on the method from NLP – represent a document using a histogram of word frequency
- In an image, “words” are local features

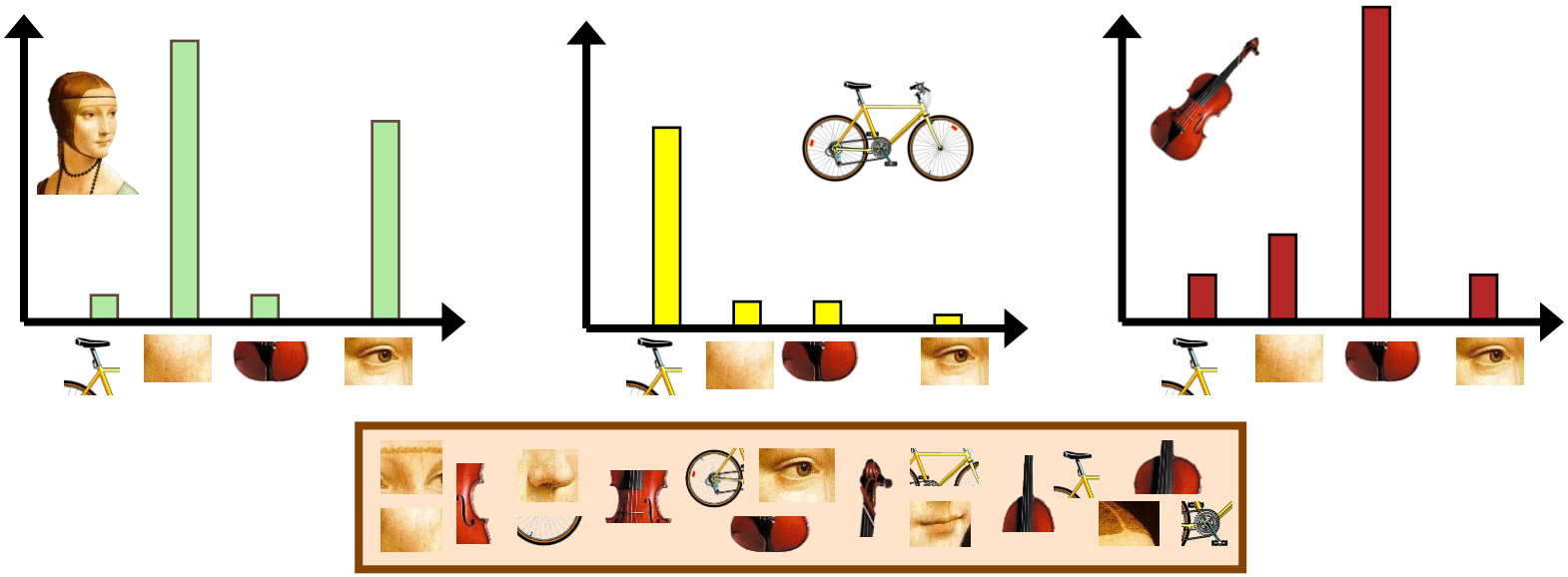
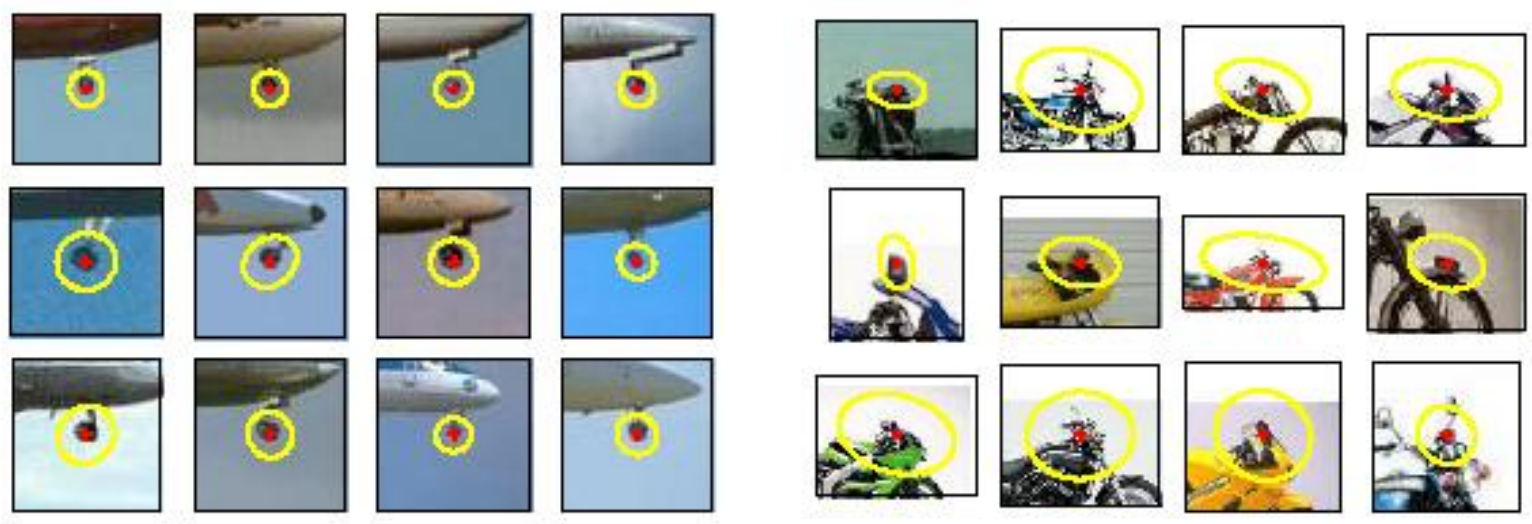


Image: Li Fei Fei

Words from pixels?

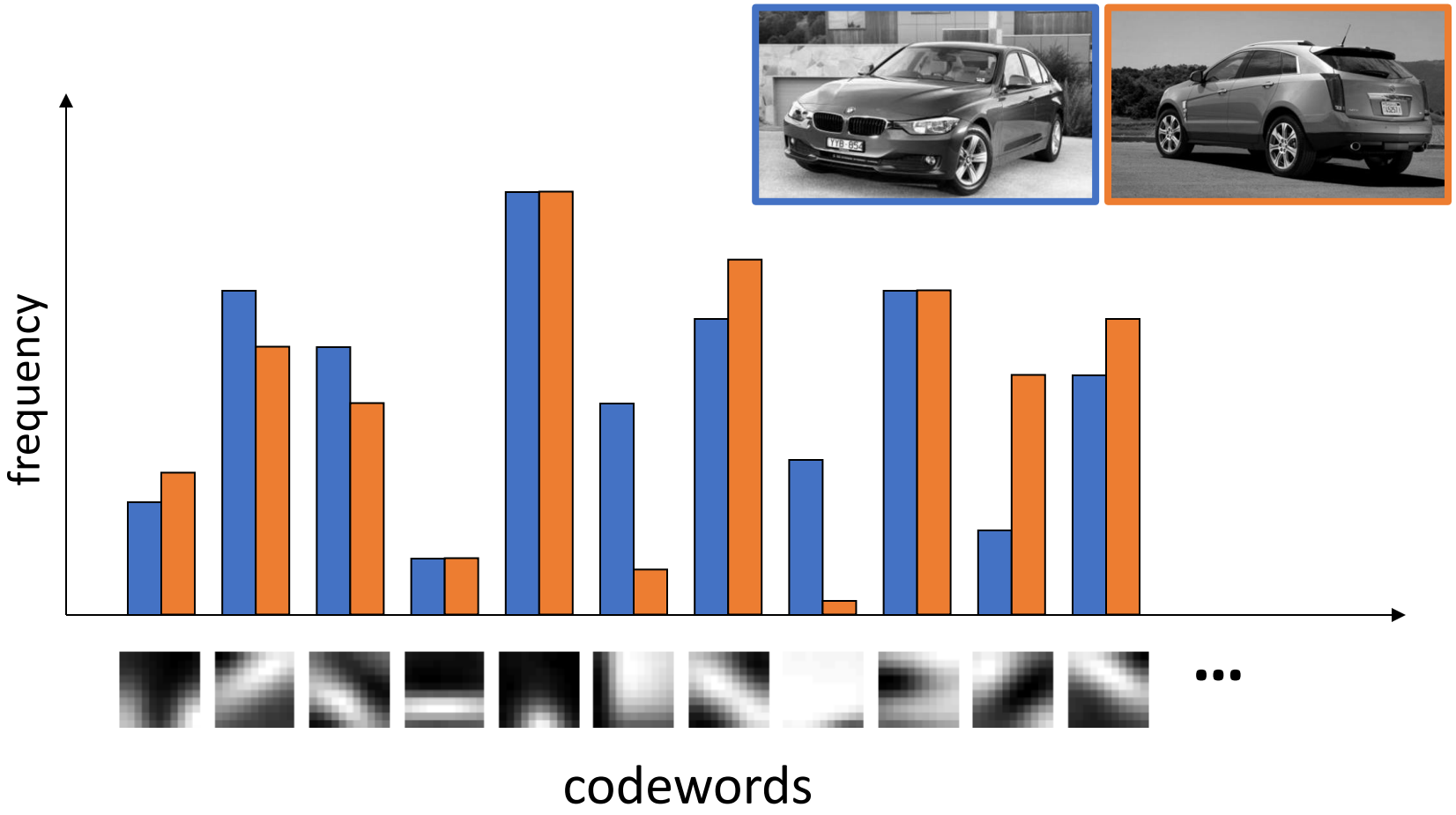
- Problem: in images, the same “word” can have many appearances



- Solution: cluster similar patches, e.g., with k-means clustering

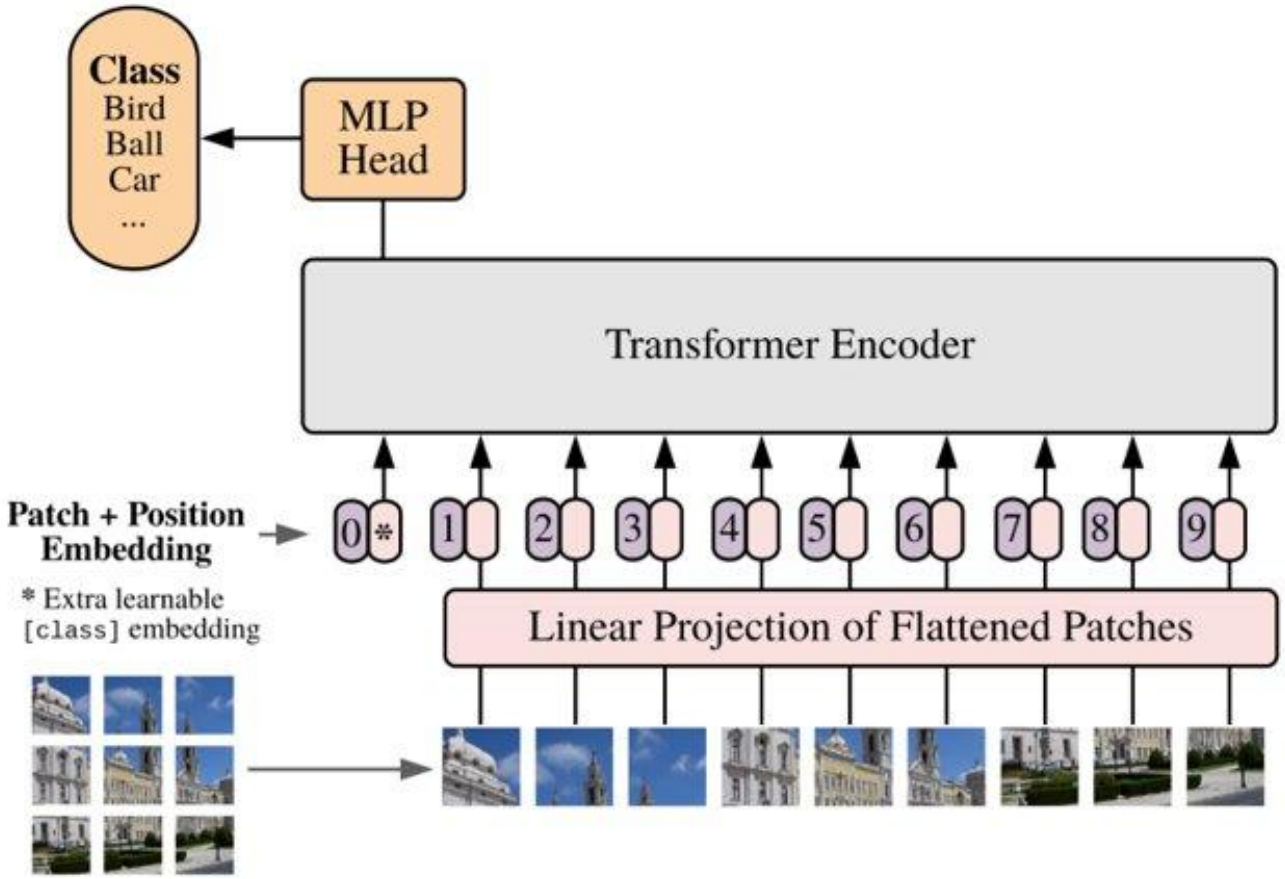
Image: Sivic et al. (2005)

Bag of features matching



Adapted from: Li Fei Fei

Vision Transformer (ViT)



Dosovitskiy, et al. (2021)

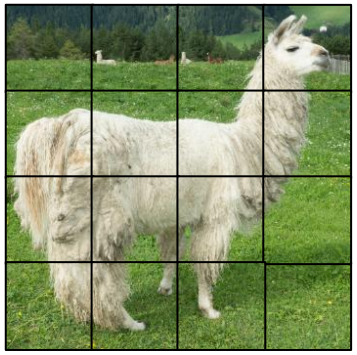
Image as "words"

- Split the image into N patches:
  - E.g., 16 x 16 non-overlapping patches in original ViT (14x14x3 = 588 pixel values for ImageNet)
- Compute an "embedding" of the patch
  - Flatten patch + fully-connected layer
  - E.g., embedding from a trained CNN
- Input to Transformer is a 1D sequence of patch embeddings

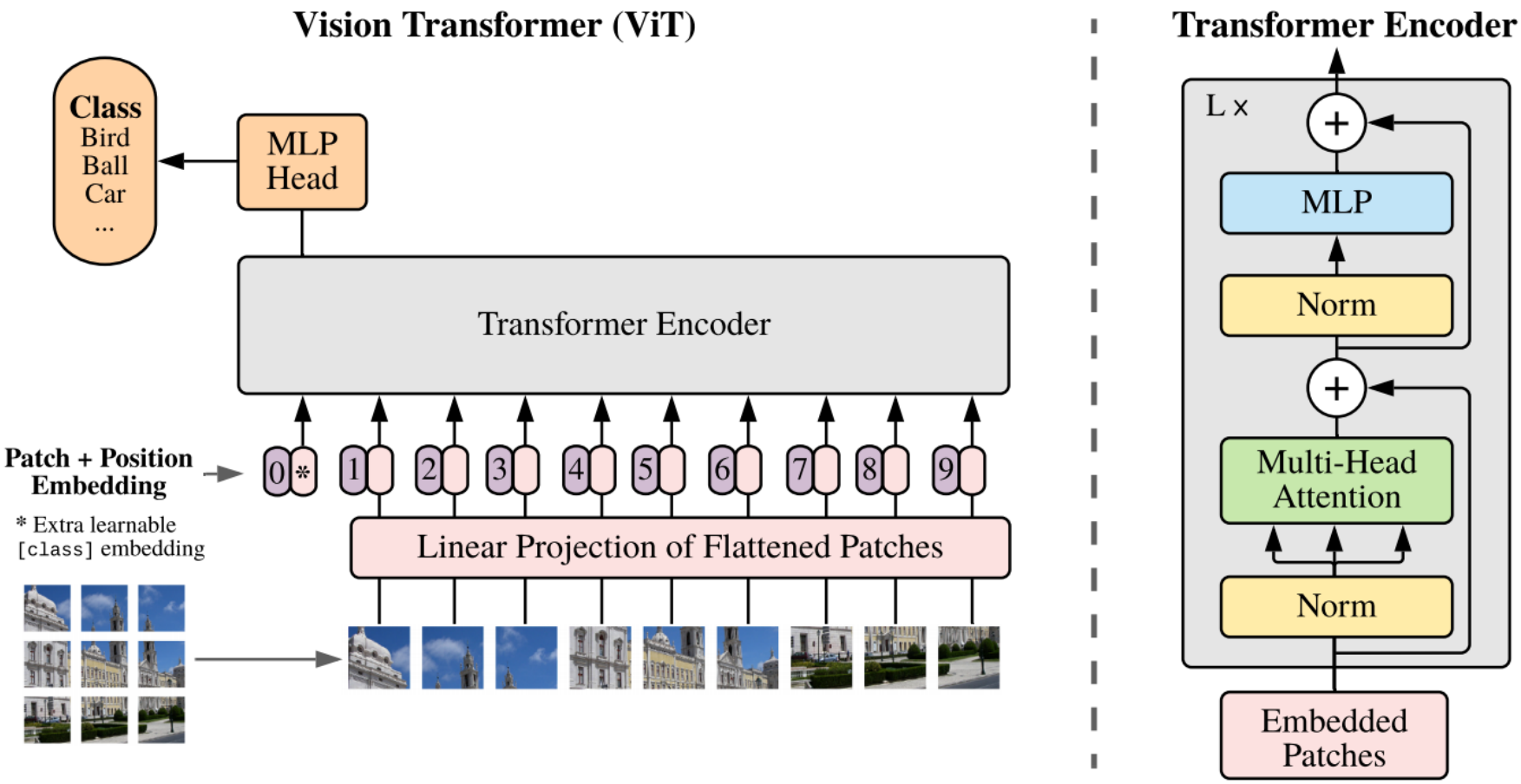
Positional encoding

- None – treat the inputs as a "bag of words"
- 1D position in the patch sequence
- 2D x,y position in the image
- Relative position between patches

Image as "words"



Vision Transformer (ViT)



Dosovitskiy, et al. (2021)

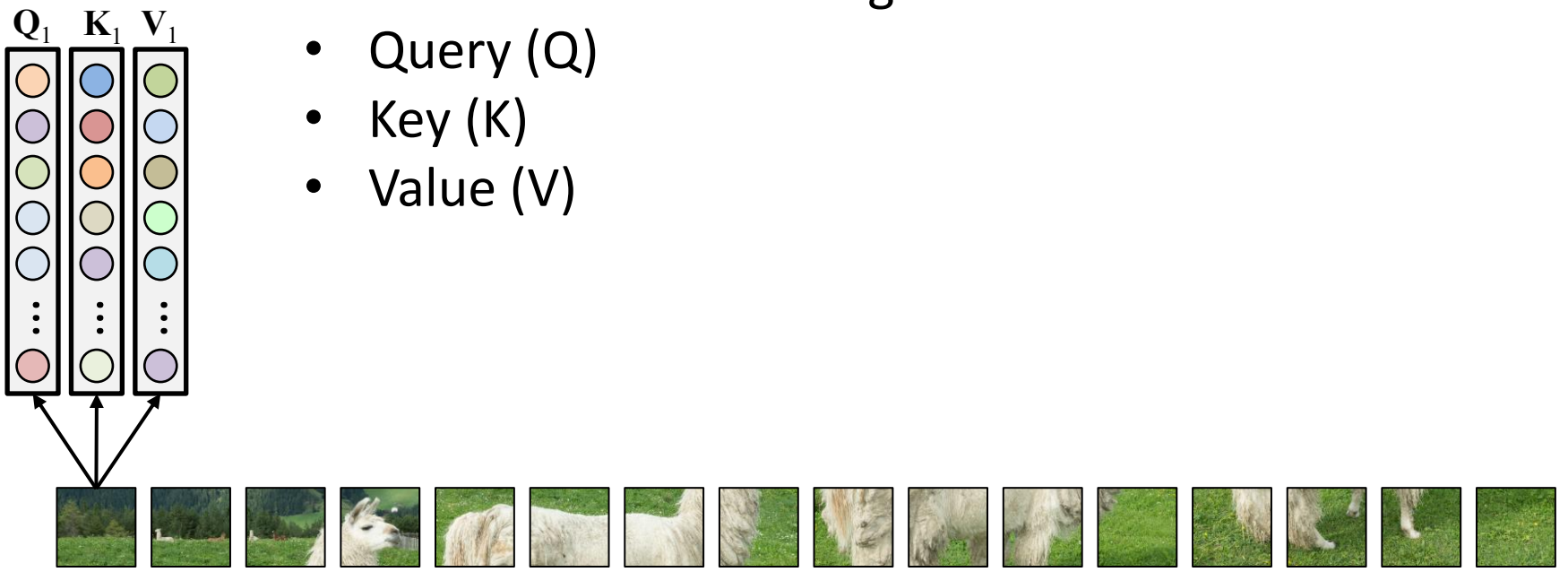
Transformer encoder

- Multi-headed self-attention (MSA)
  - Self-attention = the input is compared to itself
  - Multi-headed = multiple attention "heads" work in parallel to learn different relationships in the data
- Multi-layer perceptron (MLP)
  - Additional non-linear transformation of input

Attention head

Map each patch embedding to three vectors via trainable weights:

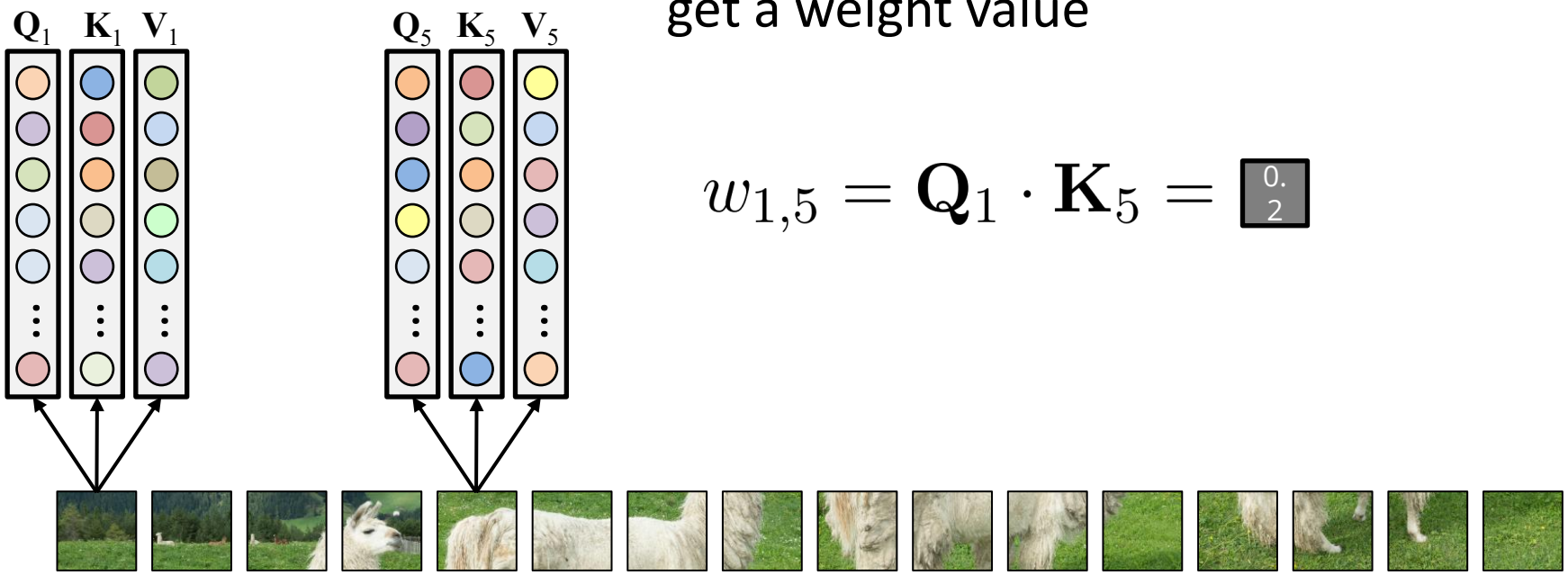
- Query (Q)
- Key (K)
- Value (V)



Attention head

For each patch, compute dot product (similarity) between query and key to get a weight value

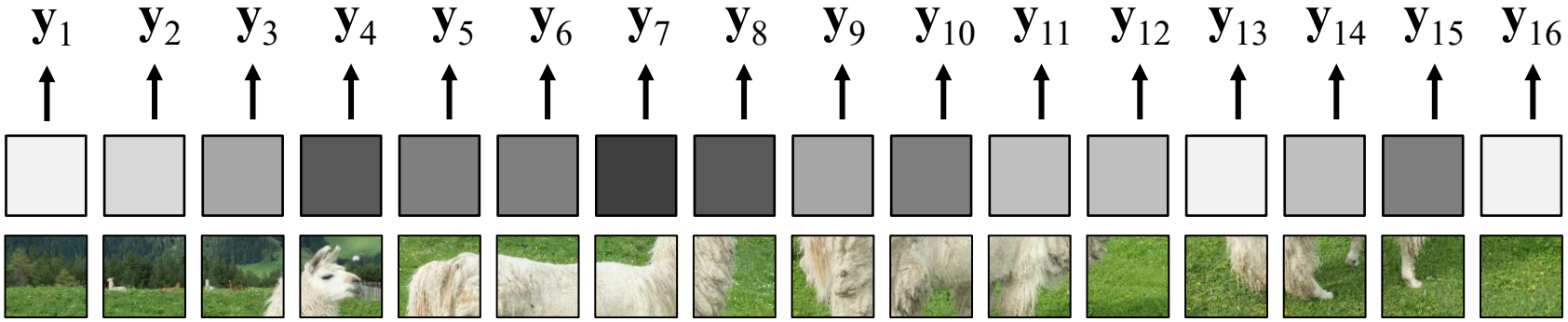
$$w_{1,5} = \mathbf{Q}_1 \cdot \mathbf{K}_5 = \boxed{0.2}$$



Attention head

Compute weighted sum of value vectors

$$\text{New vector } \mathbf{y}_1 = \sum_{i=1}^{n} \text{softmax}\left(\frac{\mathbf{Q}_1 \cdot \mathbf{K}_i}{D}\right) \mathbf{V}_i$$



Attention head

- Result: Transform the input patches into new vectors by comparing vectors derived from all pairs of patches
- Called "attention" because the model can learn to weight certain pairs of patches more than others
- Note that patch order is irrelevant

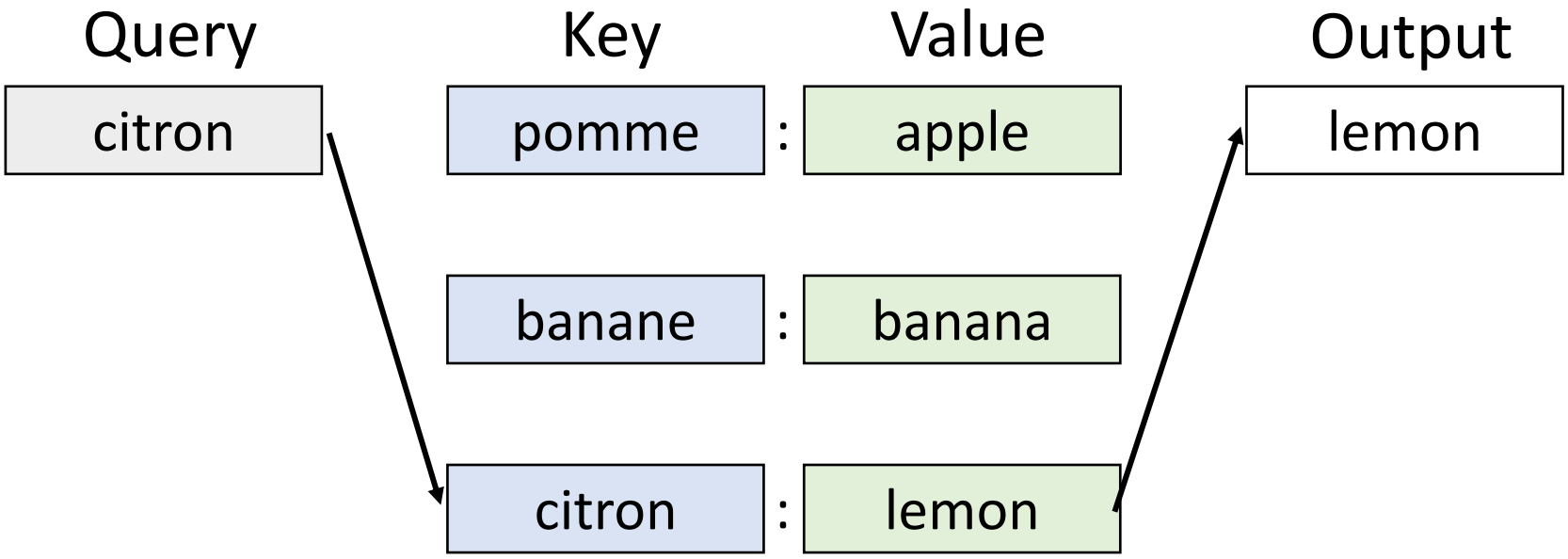
Attention head parameters

- Learned parameters are the weight matrices that map input embeddings to query, key, and value vectors:

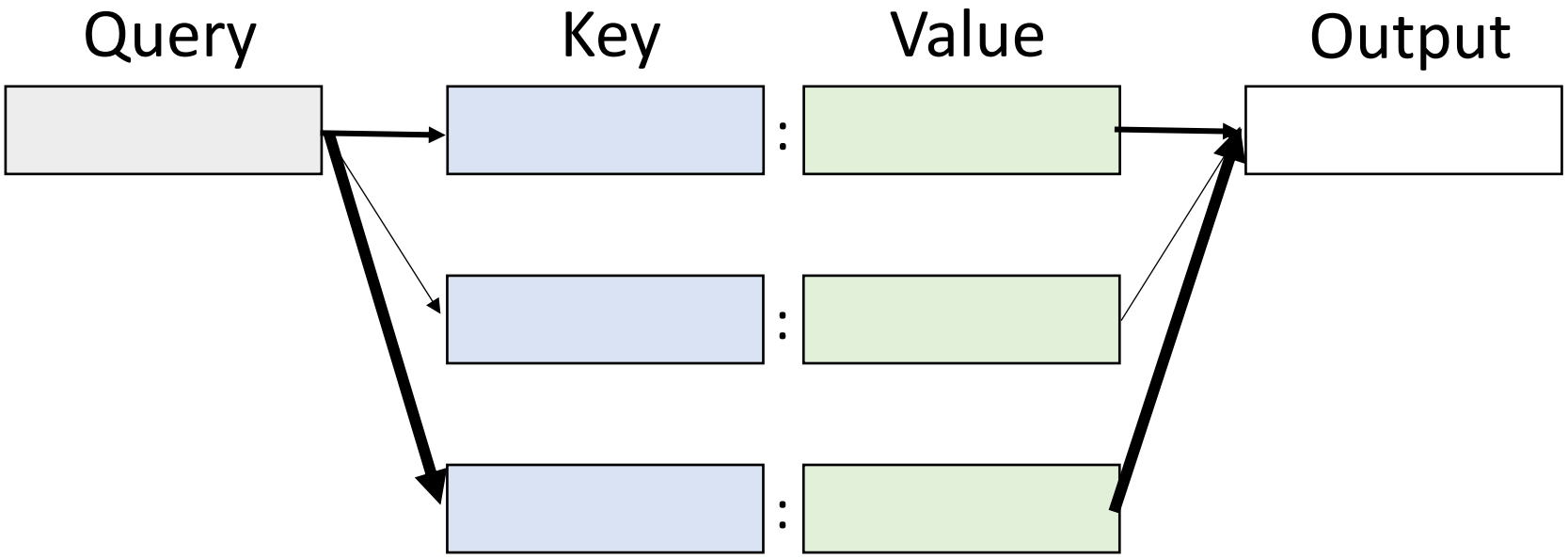
$$\mathbf{Q}_i = \mathbf{W}_\mathbf{q}\mathbf{x}_i, \mathbf{K}_i = \mathbf{W}_\mathbf{k}\mathbf{x}_i, \mathbf{V}_i = \mathbf{W}_\mathbf{v}\mathbf{x}_i$$

- Multiheaded: Each head learns its own weights, which allows each head to learn different relationships in the data, or perform different tasks

Attention as a dictionary



Attention as a dictionary



Everything is an embedding

"Soft" look-up based on Query-Key similarity

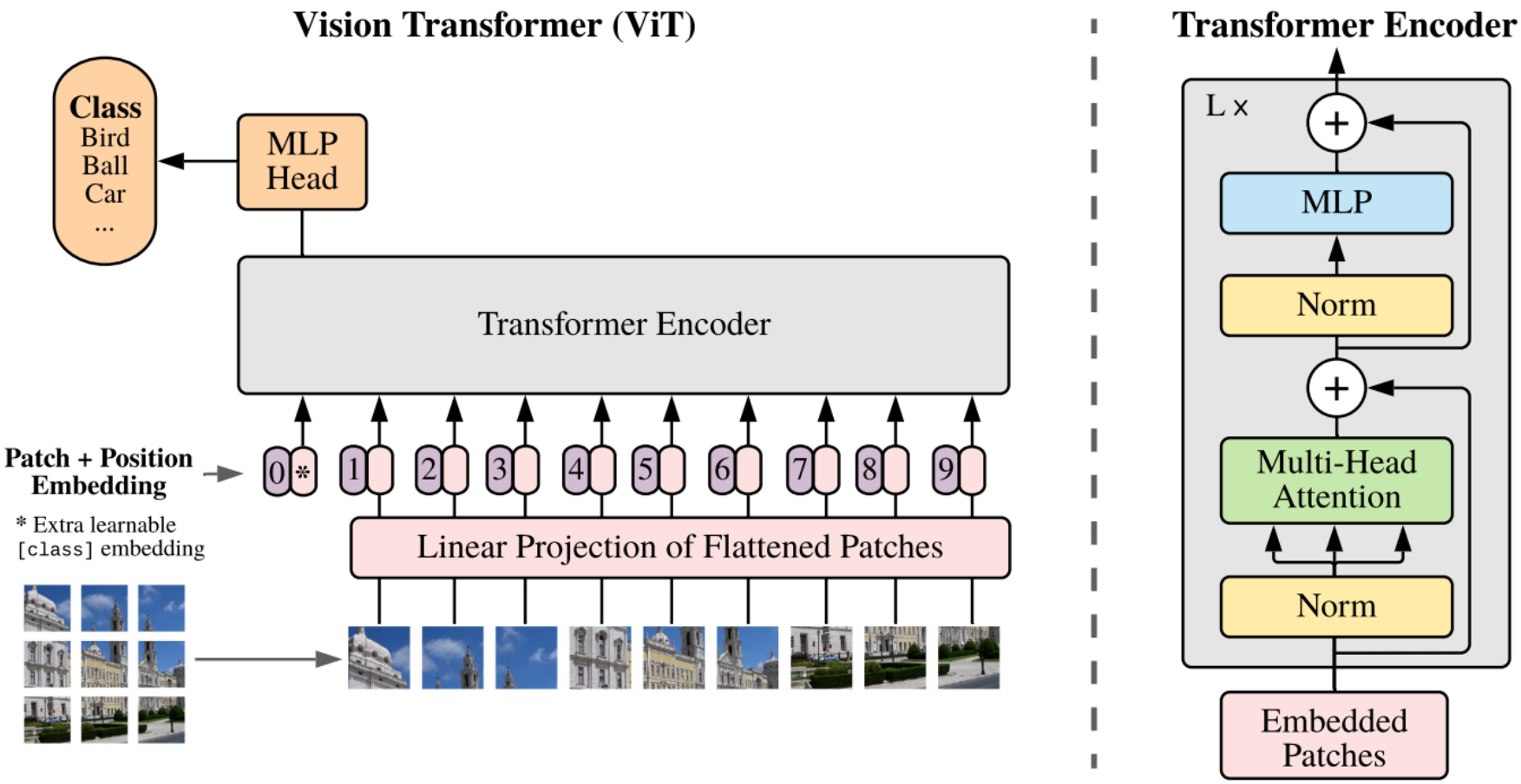
Attention head parameters

- Each patch embedding is used 3 times in self-attention:
  - To determine how to weight every other patch's contribution to this patch's output (Query)
  - To weight this patch's contribution to other patches' outputs (Key)
  - To compute the outputs (Value)
- Dimensions
  - len(Query) = len(Key), len(output) = len(Value)
  - N Querys = N outputs, N Keys = N Values

Multi-layer perceptron

- Attention transforms input vectors into new vectors
- New vectors are further processed by a multilayer perceptron (MLP)
  - E.g., 2 fully-connected layers in original ViT
- Layernorm in between each step
  - Normalise over features instead of images
  - Borrowed from NLP, where you can't easily normalise over sentences of different length

Vision Transformer (ViT)



Dosovitskiy, et al. (2021)

Vision Transformer (ViT)

- Full model is a stack of Transformer layers
- User defined parameters:
  - Number of transformer layers (original ViT uses 12, 24, or 32)
  - Number of attention heads (ViT uses 12 or 16)
  - Size of the embeddings in attention layers, MLP layers

Summary

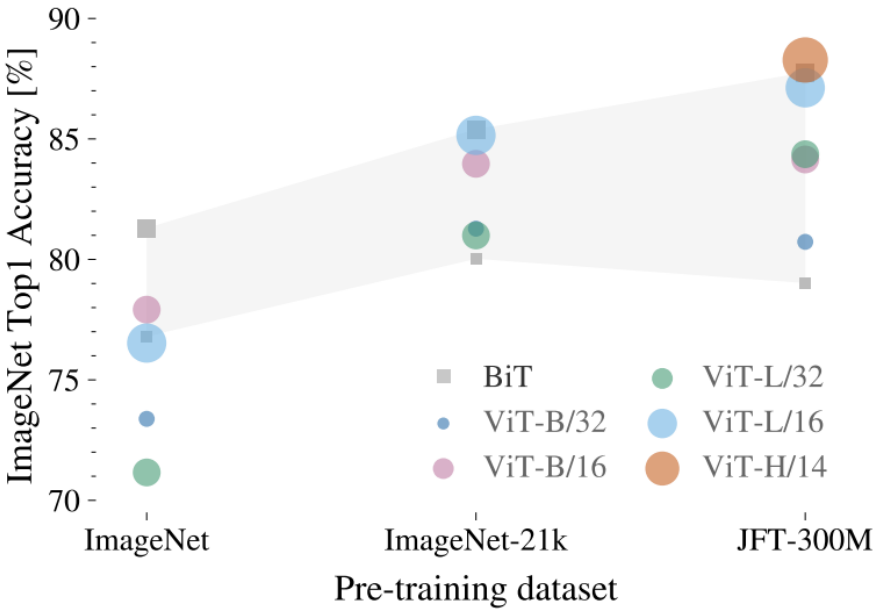
- Model borrowed from NLP:
  - Treat image as a "sentence" = 1D sequence of patches
  - Uses attention to learn relationships between pairs of patches
  - Can learn "features" from disconnected parts of the image
- Removes some of the inductive bias from CNNs

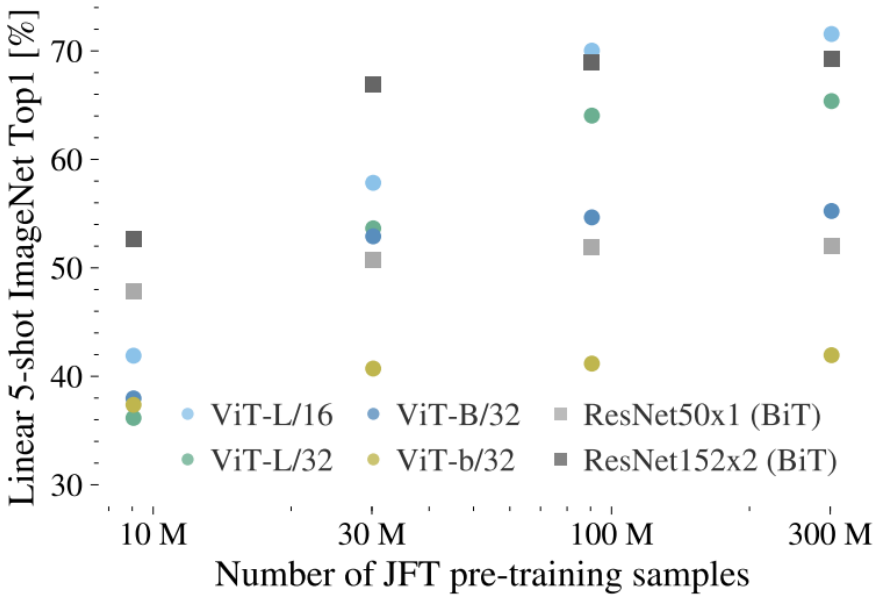
Transformer results

Training Transformers

- Transformers are even more prone to overfitting on small datasets than CNNs, require more data than CNNs to learn generalisable features
- ViT pretrained on JFT-300M (300 million images) and fine-tuned for smaller datasets like ImageNet

Transformer results

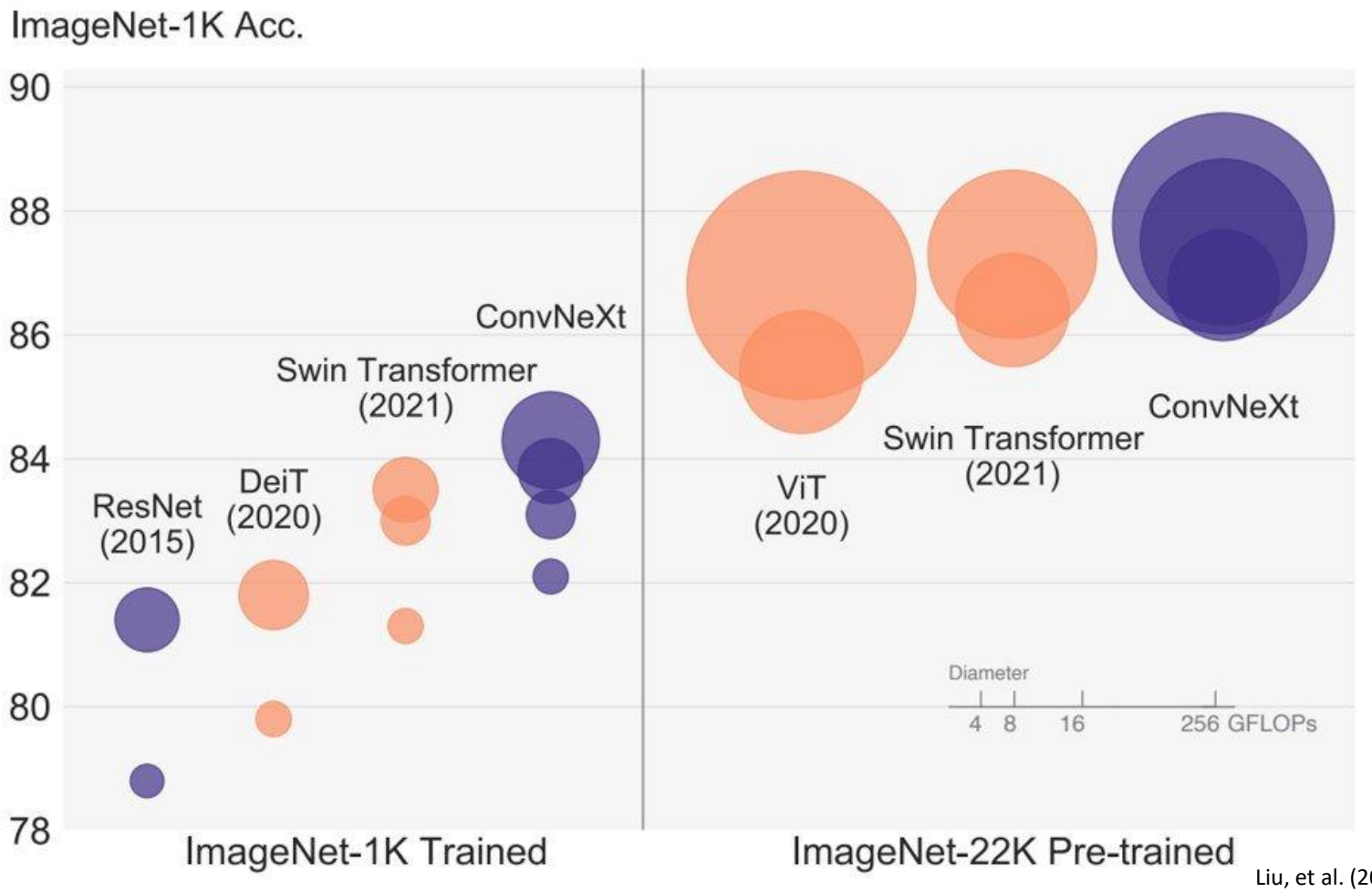




Claim: Transformers outperform CNNs on very large scale datasets

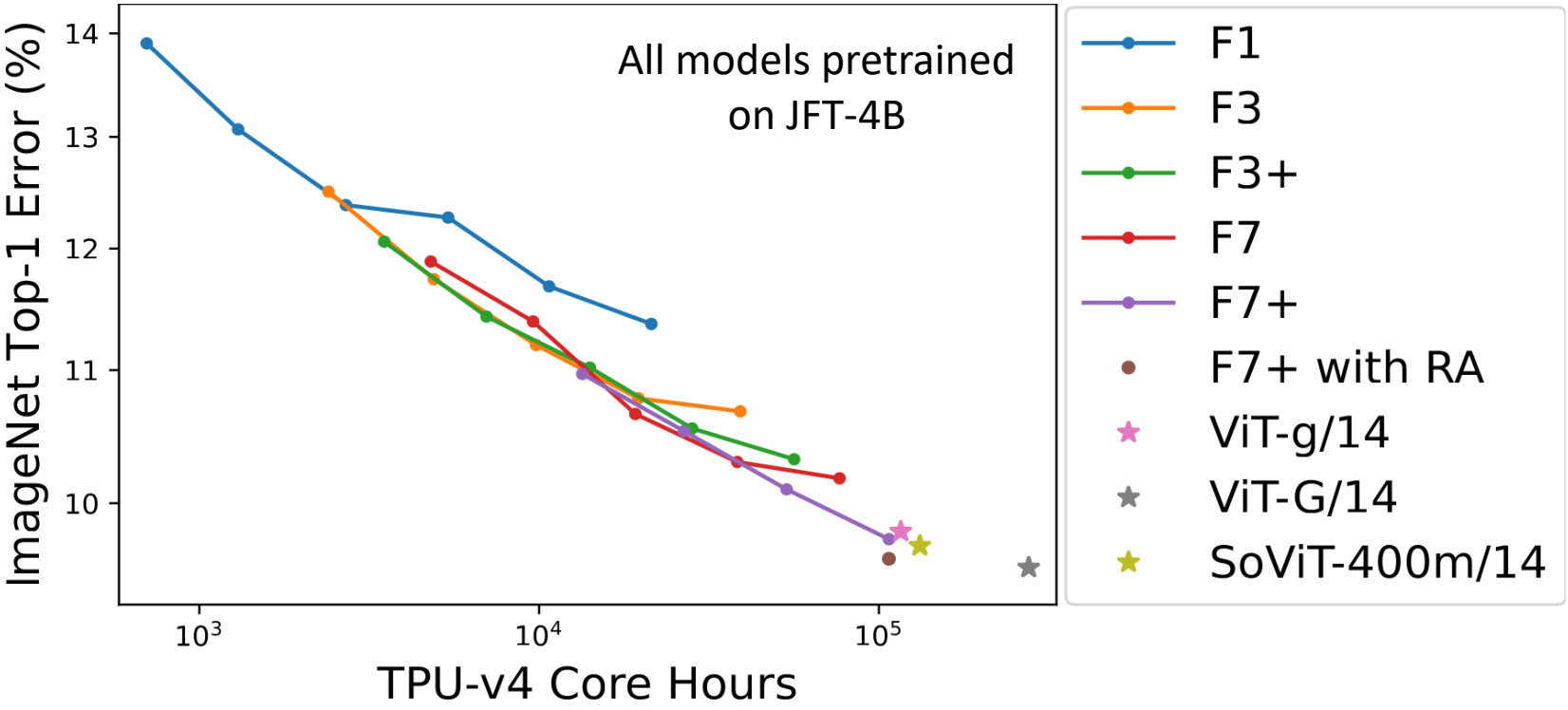
Dosovitskiy, et al. (2021)

Transformer results



Liu, et al. (2022)

Transformers vs. CNNs



Claim: Given equal pretrain compute budget, ViT and CNN performance is the same on ImageNet

Smith, et al. (2023)

Transformers vs. CNNs

- Transformers learn better on large-scale datasets
  - Unclear, may be equivalent when both models have the same computational budget
- Transformers learn features that generalise better to new contexts
  - Common assumption, not yet rigorously tested

Inductive biases of ViTs

- Positional encoding
- Patch encoding
- Some evidence of bias in what functions they learn; e.g., preference for functions that ignore input order (Lavie, Gur-Ari, Ringel (2024))

Smith, et al. (2023)

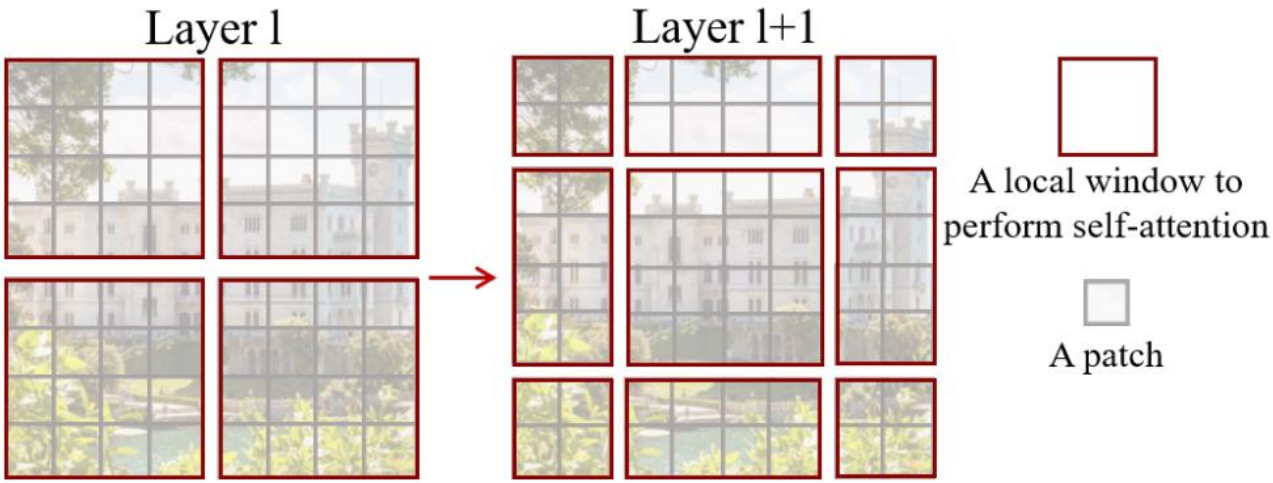
Patches as "words"

- Simple but not ideal: ignores scale, important information may be split across patches
- Overlapping patches? Multiscale patches?
- Adding patches is computationally expensive: attention is O(N^2) when N is number of patches

Smith, et al. (2023)

Problem with patches = "words"

- Improvements on ViT have tried to improve patching without increasing computational complexity
- Example: Swin Transformer



Liu et al. (2021)

Summary

- Advantages of ViT over CNNs
  - Can efficiently learn features based on distant parts of an image
  - Might learn better features for generalisation
- Disadvantages of ViT
  - Lacks the inductive biases of convolution (locality and translation invariance of features)
  - Simple non-overlapping patches can miss or obscure important features
  - Computational cost scales quadratically with number of patches