IMAG/e

TU/e Technische Universiteit Eindhoven University of Technology

Discussion of transformer models

Mitko Veta
M.Veta@tue.nl

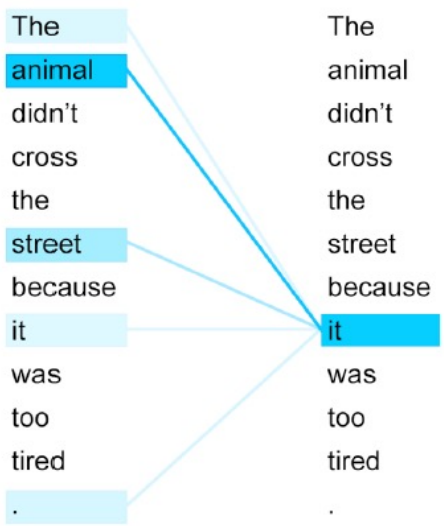
Image from xkcd.com

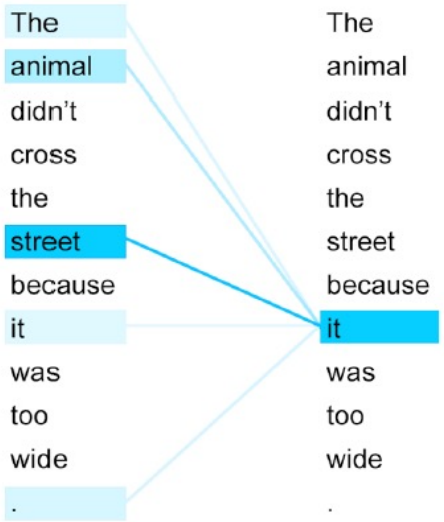
Attention in language:

Low attention

"This GitHub page contains all the general information about the course and the study materials."

High attention





https://probml.github.io/pml-book/book1.html

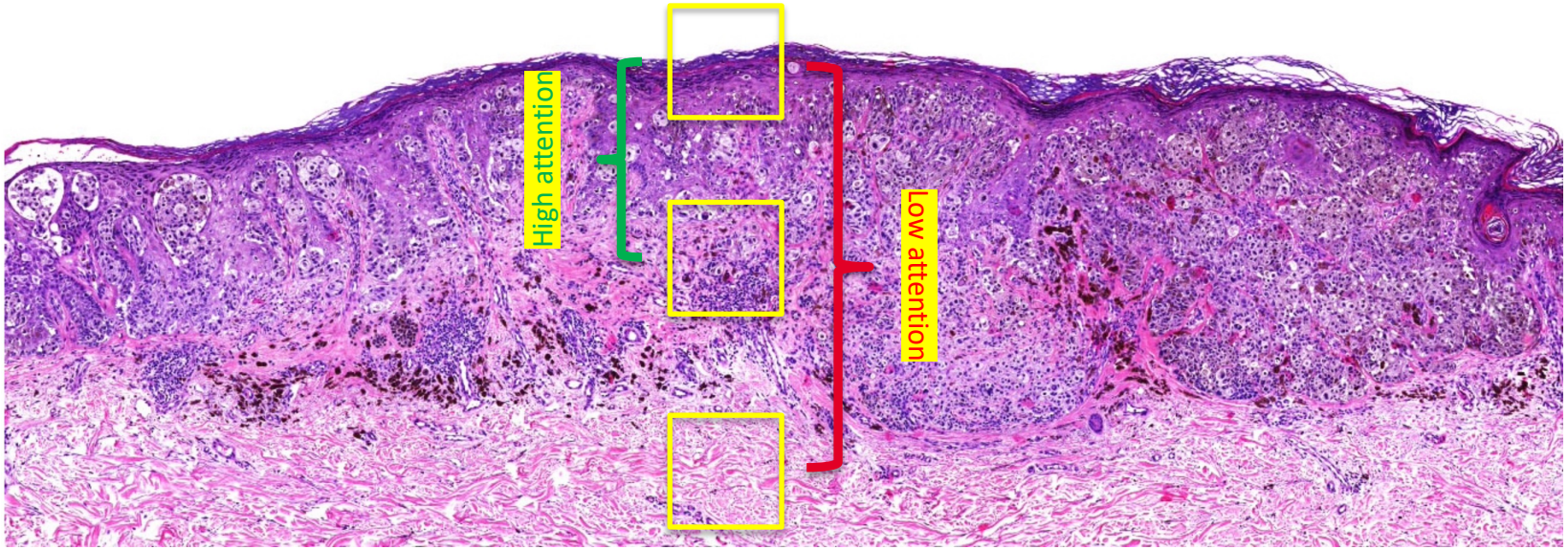
Attention in language:

Low attention

"This GitHub page contains all the general information about the course and the study materials."

High attention

Attention in vision (invasive melanoma histology example):



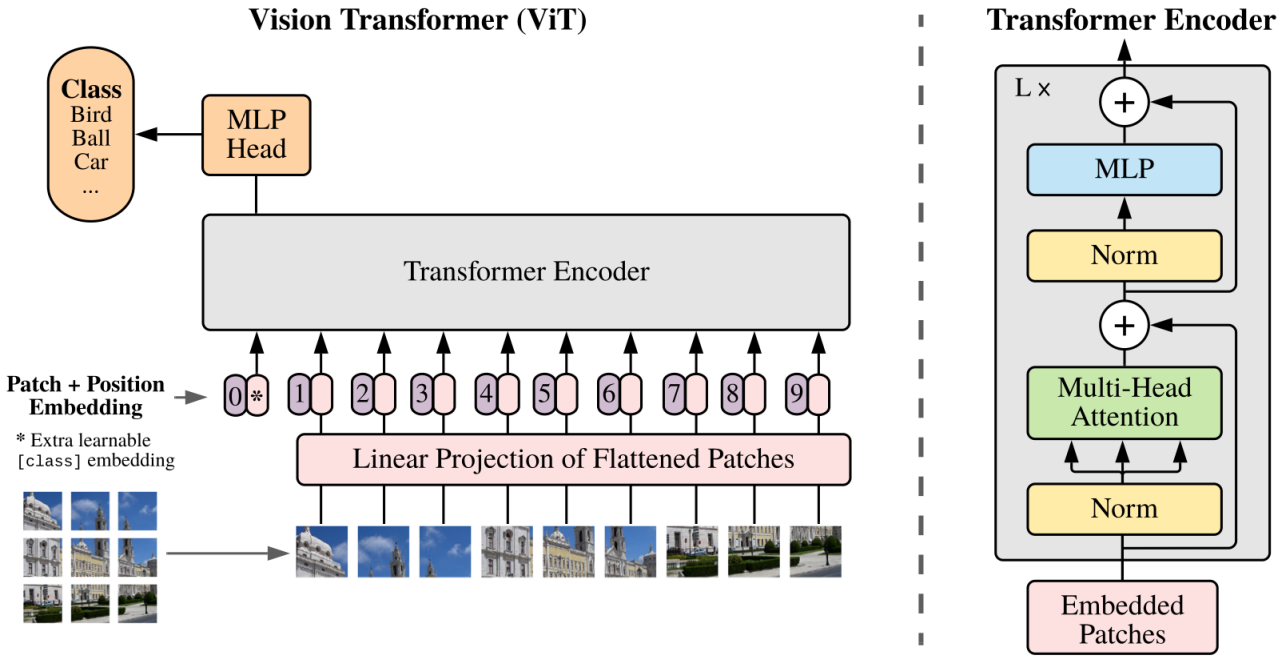
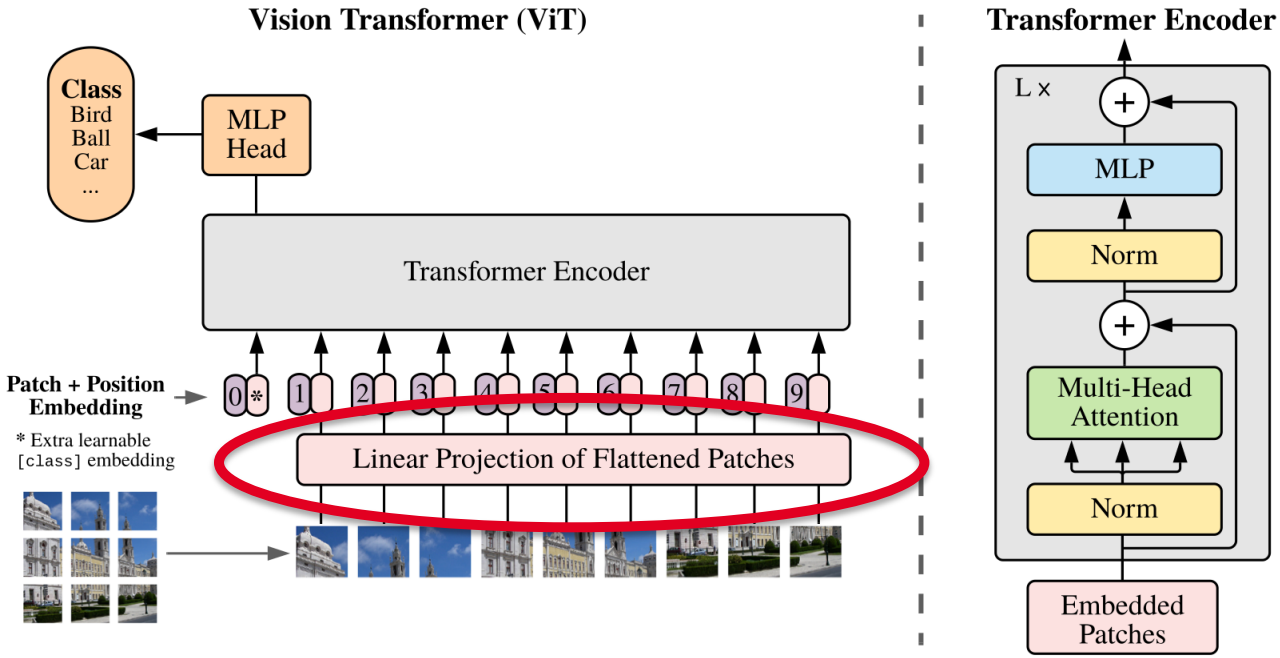
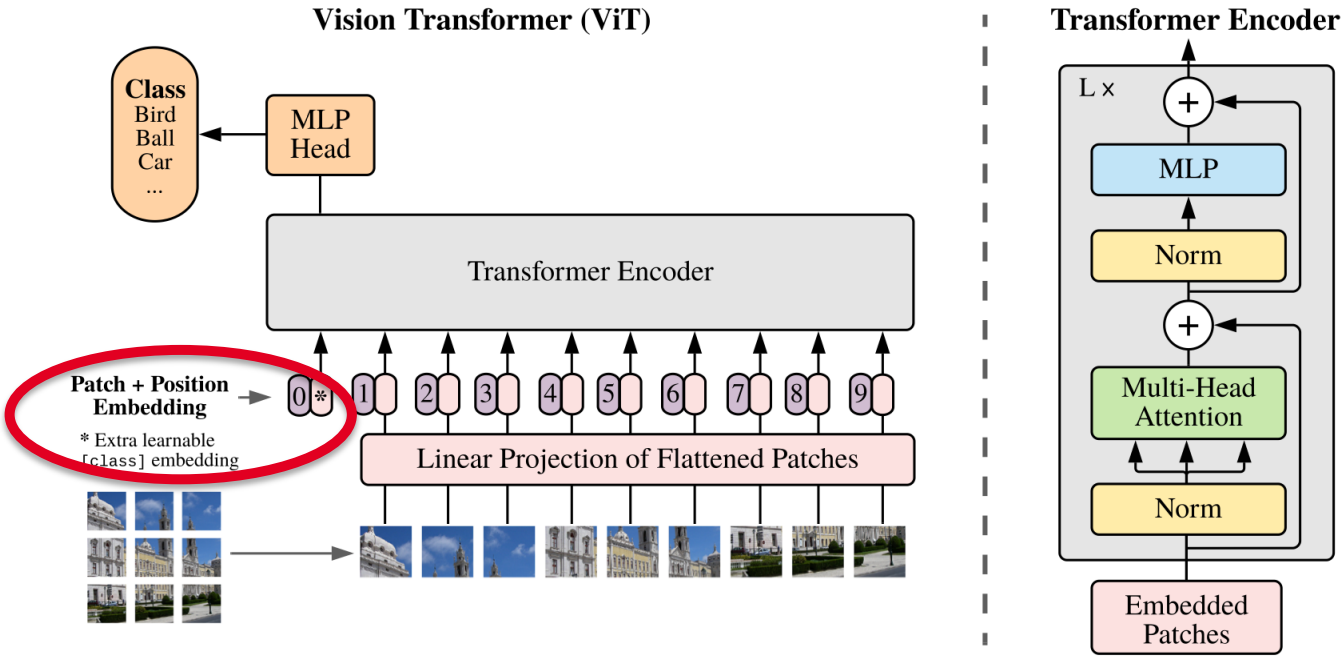
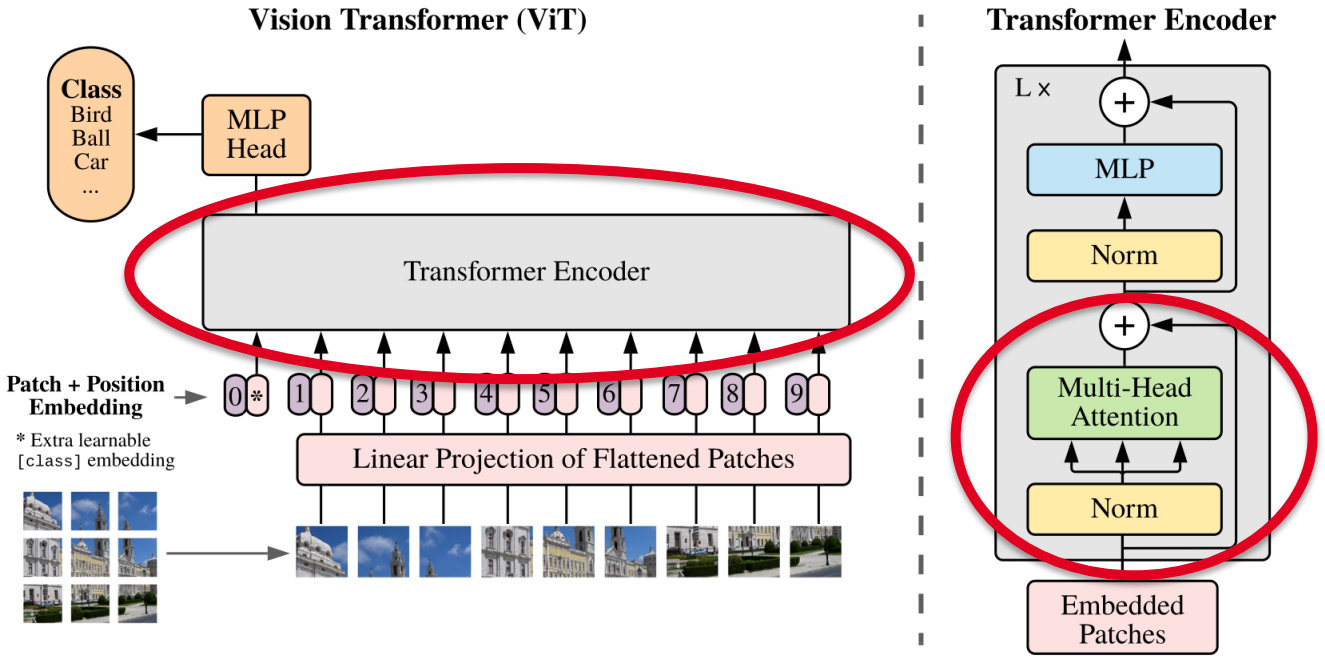
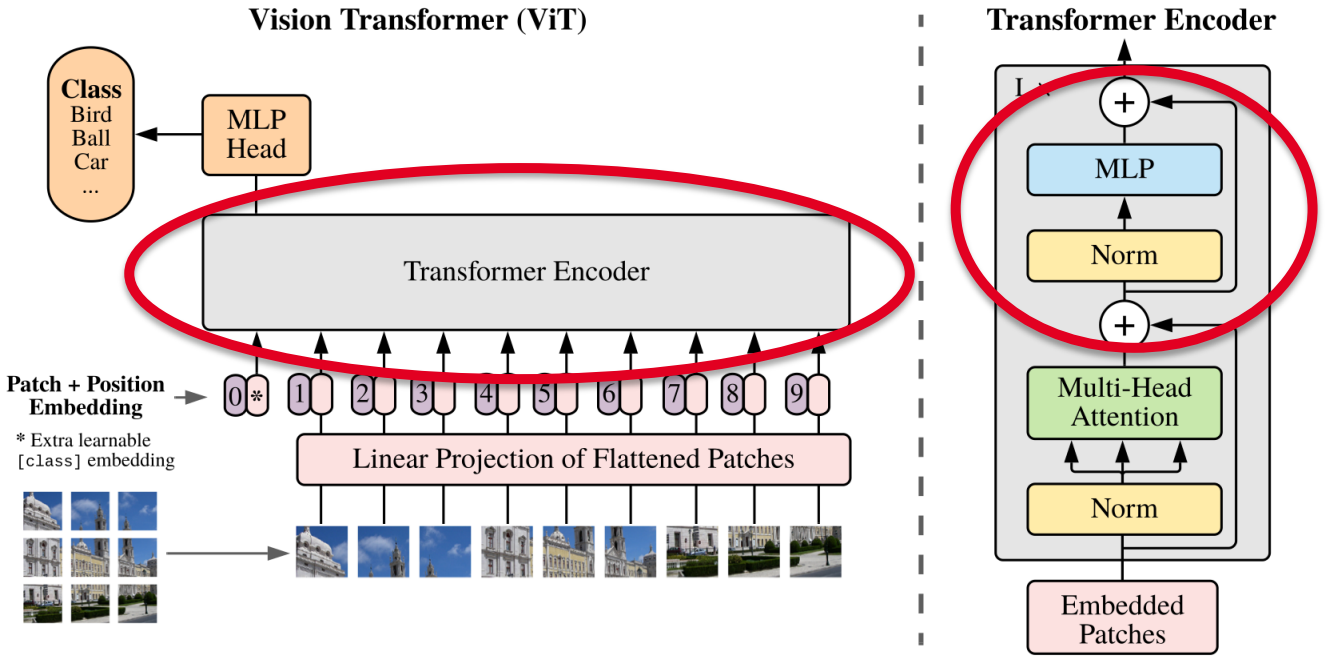


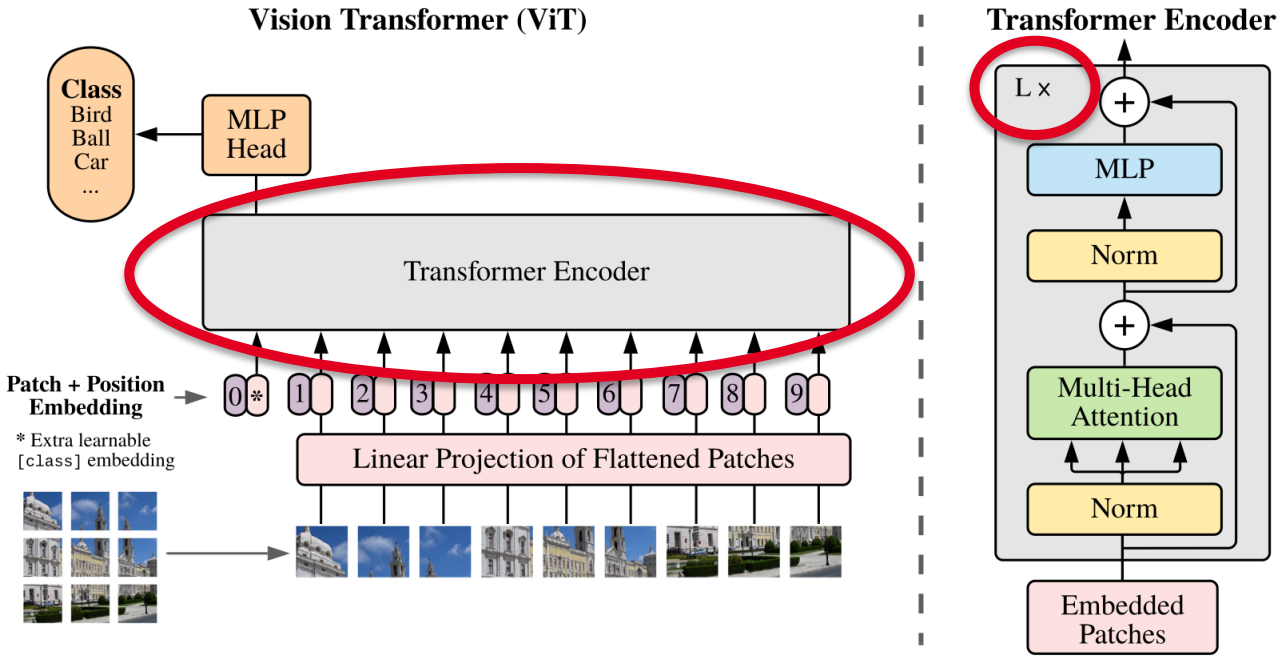
Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

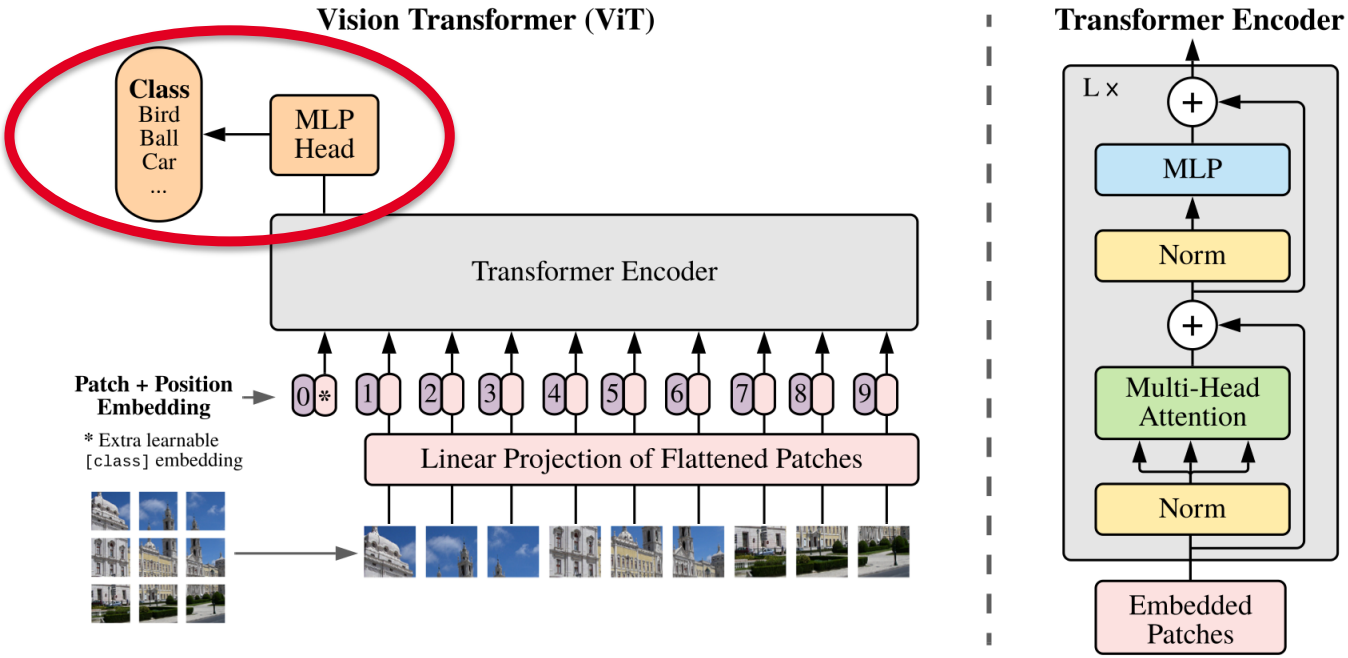

Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches
Transformer Encoder
L x
MLP
Norm
Multi-Head Attention
Norm
Embedded Patches

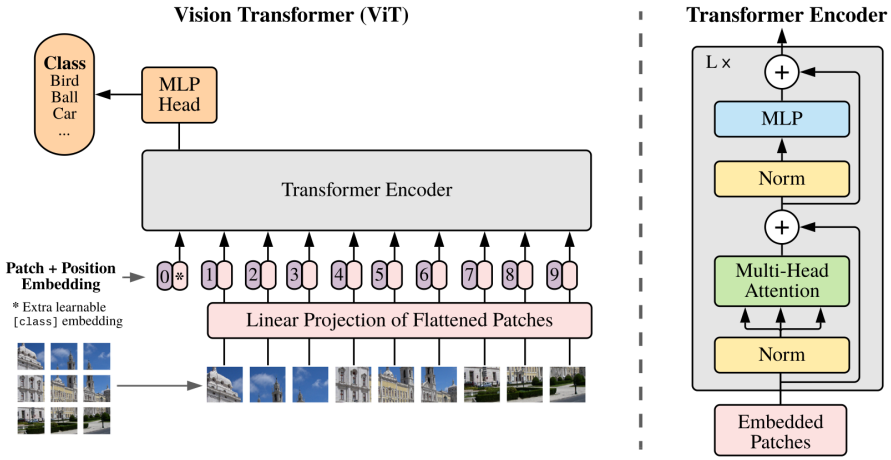

Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches
Transformer Encoder
L x
MLP
Norm
Multi-Head Attention
Norm
Embedded Patches


Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches
Transformer Encoder
L x
MLP
Norm
Multi-Head Attention
Norm
Embedded Patches


Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches
Transformer Encoder
MLP
Norm
Multi-Head Attention
Norm
Embedded Patches


Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches
Transformer Encoder
L x
MLP
Norm
Multi-Head Attention
Norm
Embedded Patches


Vision Transformer (ViT)
Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
* Extra learnable [class] embedding
0 * 1 2 3 4 5 6 7 8 9
Linear Projection of Flattened Patches
Transformer Encoder
L x
MLP
Norm
Multi-Head Attention
Norm
Embedded Patches




```
def EncoderBlock(X):
  Z = LayerNorm(MultiHeadAttn(Q=X, K=X, V=X) + X)
  E = LayerNorm(FeedForward(Z) + Z)
  return E

def Encoder(X, N):
  E = POS(Embed(X))
  for n in range(N):
     E = EncoderBlock(E)
  return E
```

Q1: In transformer models, which of the following vectors have to be of equal length: key, query, value?

Q2: Does the order of the input sequence elements matter?

Q3: What are the advantages and disadvantages of vision transformer models compared to state-of-the-art convolutional neural network models for (medical) image analysis?

Disadvantages:

Need more data to train

Computationally demanding

Advantages:

Less assumptions about the data

Include global information in the early layers

Q4: Why is the inductive bias lower for transformers compared to convolutional neural networks?

Inductive bias ~ assumptions about the data built into your model.

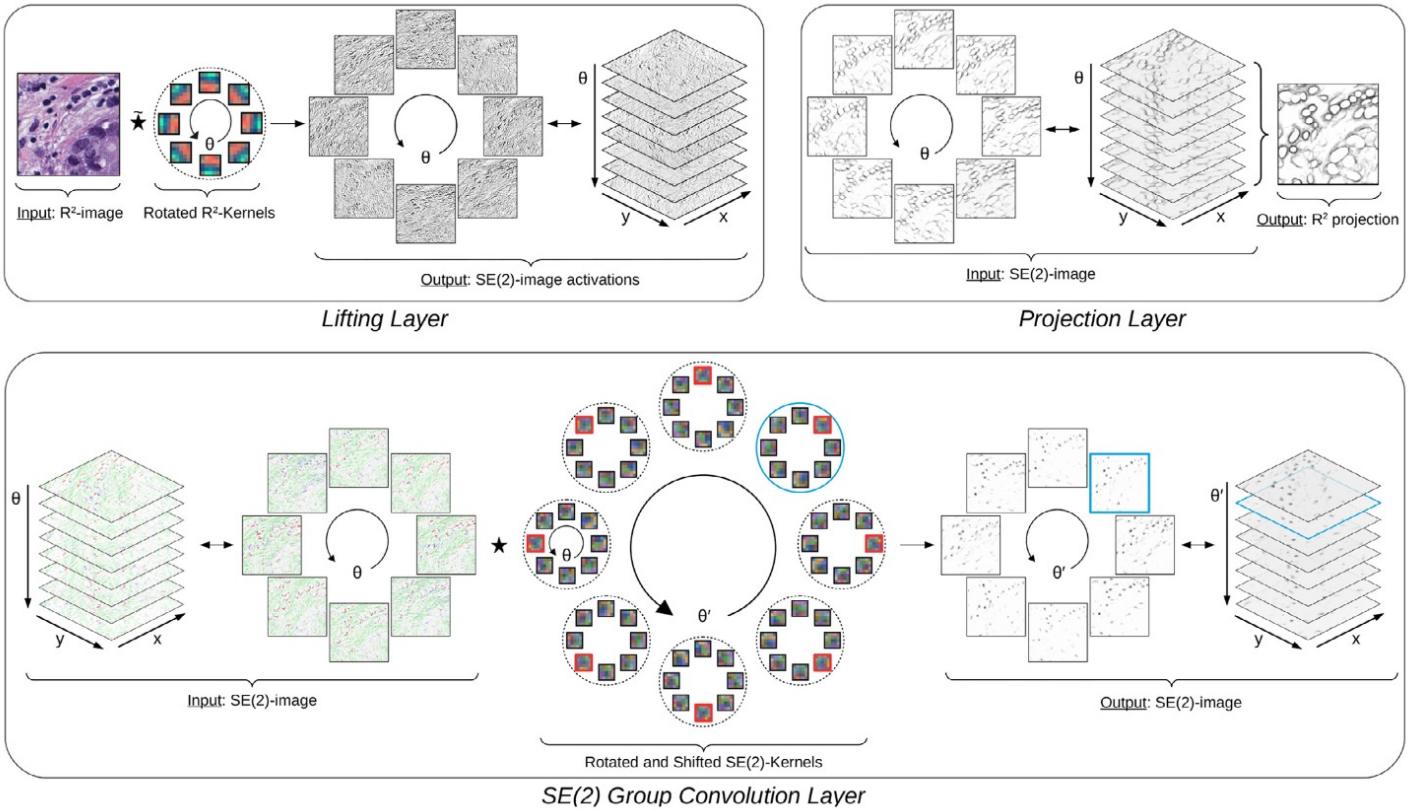
In CNNs:

- Locality (small kernels)
- Equivariance (weight sharing)
- Invariance (pooling)

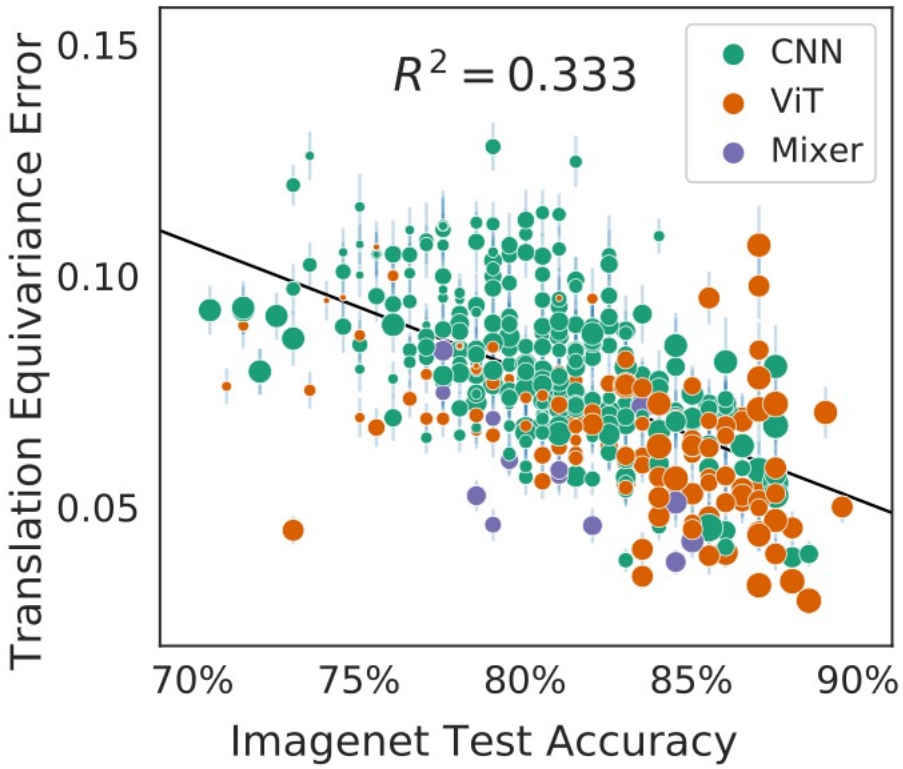
In transformers:

- Locality (cutting into patches)
- Equivariance (linear projection)
- (Both are ”weaker” inductive biases)

Example from our own work: rotational equivariance and invariance (in addition to translational) → higher inductive bias:



Lafarge et al. 2021



https://arxiv.org/pdf/2210.02984.pdf

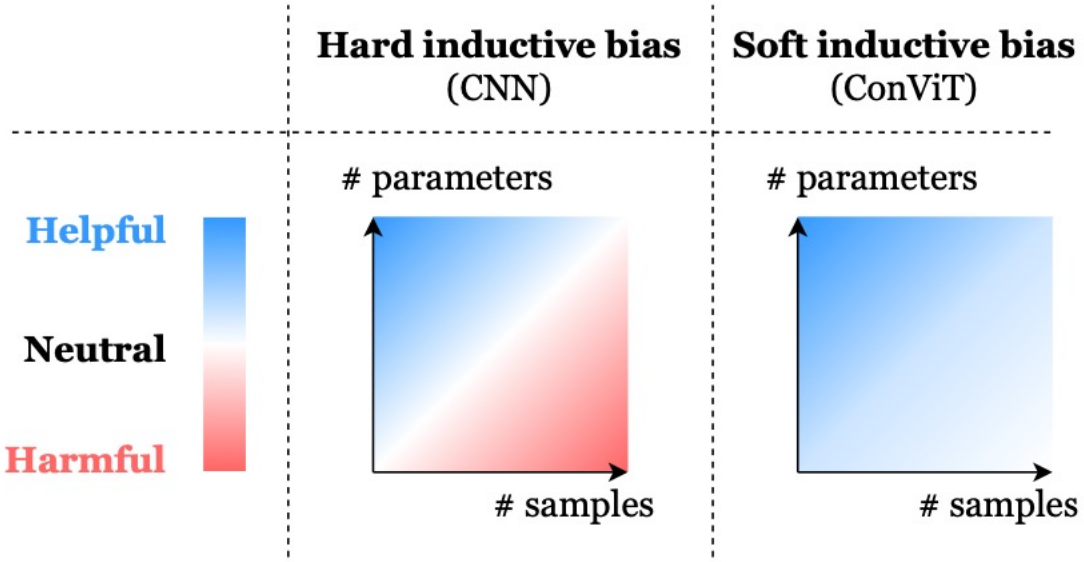


Figure 1. **Soft inductive biases can help models learn without being restrictive.** Hard inductive biases, such as the architectural constraints of CNNs, can greatly improve the sample-efficiency of learning, but can become constraining when the size of the dataset is not an issue. The soft inductive biases introduced by the ConViT avoid this limitation by vanishing away when not required.

https://arxiv.org/pdf/2103.10697.pdf

Q5: How does the self-DIstillation with NO-labels (DINO) method for pretraining vision transformers work?

DINO is a method for self-supervised pre-training.

Self-supervision ~ supervised training with ”intrinsic” labels on a pretext task. Goal is to learn good image representations that can be transferred to a downstream task.

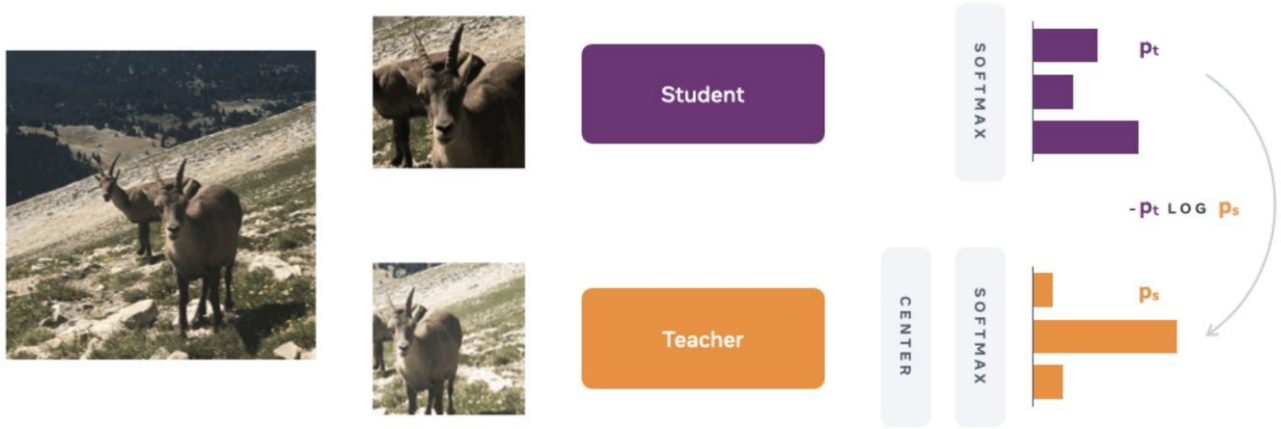
Advantages:

- Can use large, unlabelled datasets for pretraining
- “Richer” training signal compared to narrow supervised tasks

Example pretext tasks:

- Predict one part of the image from another
- Predict if two sub-images originate from the same or from different images
- Force two different traformations of the same image to have the same representation

DINO:



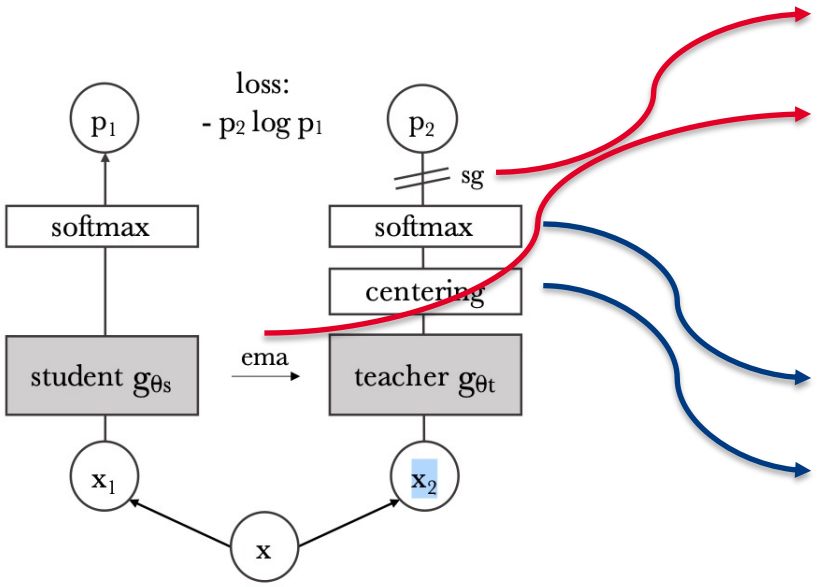


Figure 2: **Self-distillation with no labels.** We illustrate DINO in the case of one single pair of views (x_1, x_2) for simplicity. The model passes two different random transformations of an input image to the student and teacher networks. Both networks have the same architecture but different parameters. The output of the teacher network is centered with a mean computed over the batch. Each networks outputs a K dimensional feature that is normalized with a temperature softmax over the feature dimension. Their similarity is then measured with a cross-entropy loss. We apply a stop-gradient (sg) operator on the teacher to propagate gradients only through the student. The teacher parameters are updated with an exponential moving average (ema) of the student parameters.

Only the student network is updated with gradient descent, the teacher network is updated with exponential moving average of the weights of the student network.

Centering and low-temperature softmax prevent model collapse.

x_1: global and local views of the image

x_2: local views of the image

https://arxiv.org/pdf/2104.14294.pdf

Emerging properties:

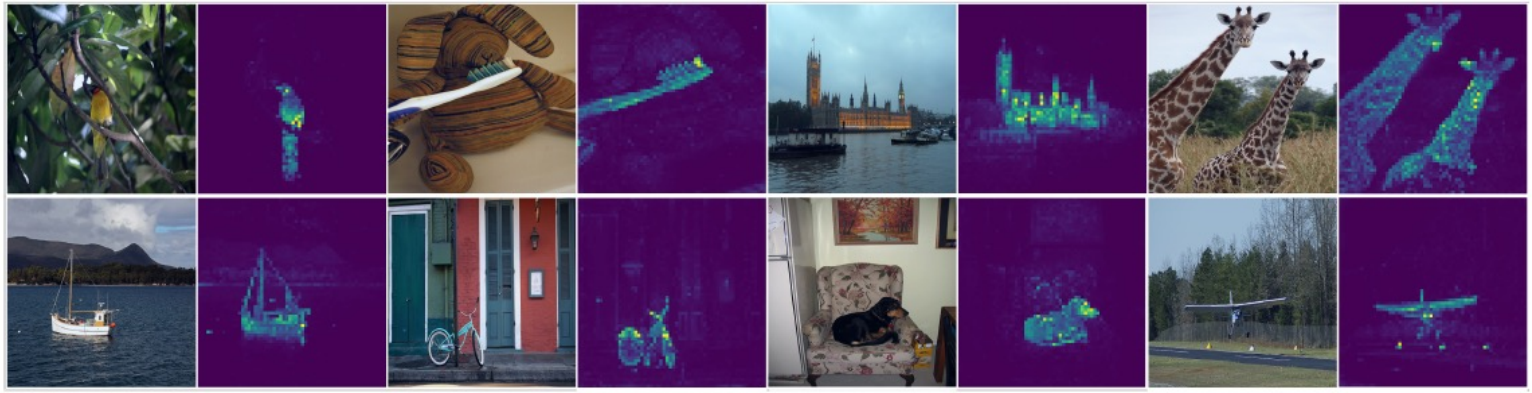


Table 10: *k*-**NN and linear evaluation for ViT-S/16 and ResNet-50 pre-trained with DINO.** We use ImageNet-1k [60] ("Inet"), Places205 [84], PASCAL VOC [24] and Oxford-102 flowers ("FLOWERS") [46]. ViT trained with DINO provides features that are particularly *k*-NN friendly.

	Logistic			*k*-NN		
	RN50	ViT-S	Δ	RN50	ViT-S	Δ
Inet 100%	72.1	75.7	3.6	67.5	74.5	7.0
Inet 10%	67.8	72.2	4.4	59.3	69.1	9.8
Inet 1%	55.1	64.5	9.4	47.2	61.3	14.1
Pl. 10%	53.4	52.1	-1.3	46.9	48.6	1.7
Pl. 1%	46.5	46.3	-0.2	39.2	41.3	2.1
VOC07	88.9	89.2	0.3	84.9	88.0	3.1
FLOWERS	95.6	96.4	0.8	87.9	89.1	1.2
Average Δ			2.4			**5.6**