IMAG/e

TU/e Technische Universiteit Eindhoven University of Technology

Discussion of transformer models

Mitko Veta
M.Veta@tue.nl

Image from xkcd.com

Outline

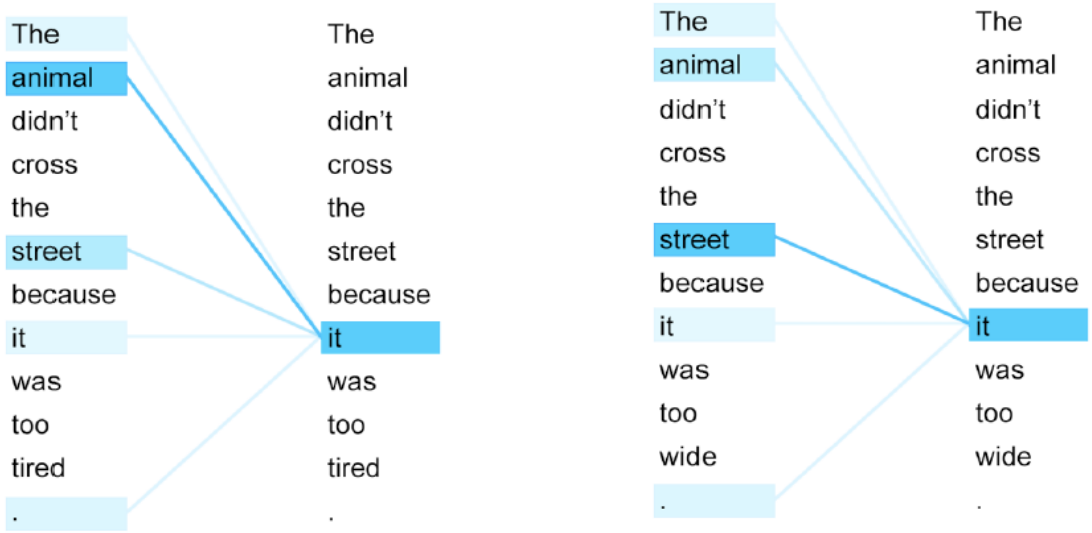
- Introduction
- Topic 1: Dataset size and inductive bias
- Topic 2: Transformers and biomedical data
- Topic 3: Self-supervised learning
- Other questions

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

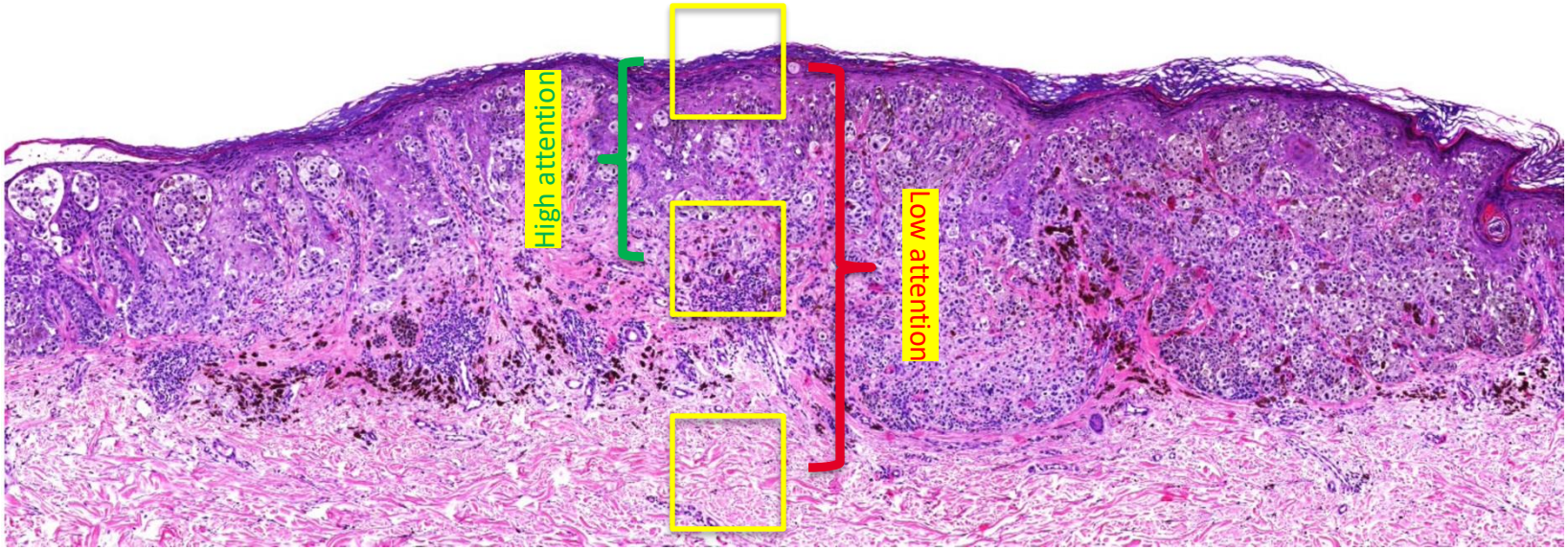
Attention in language:

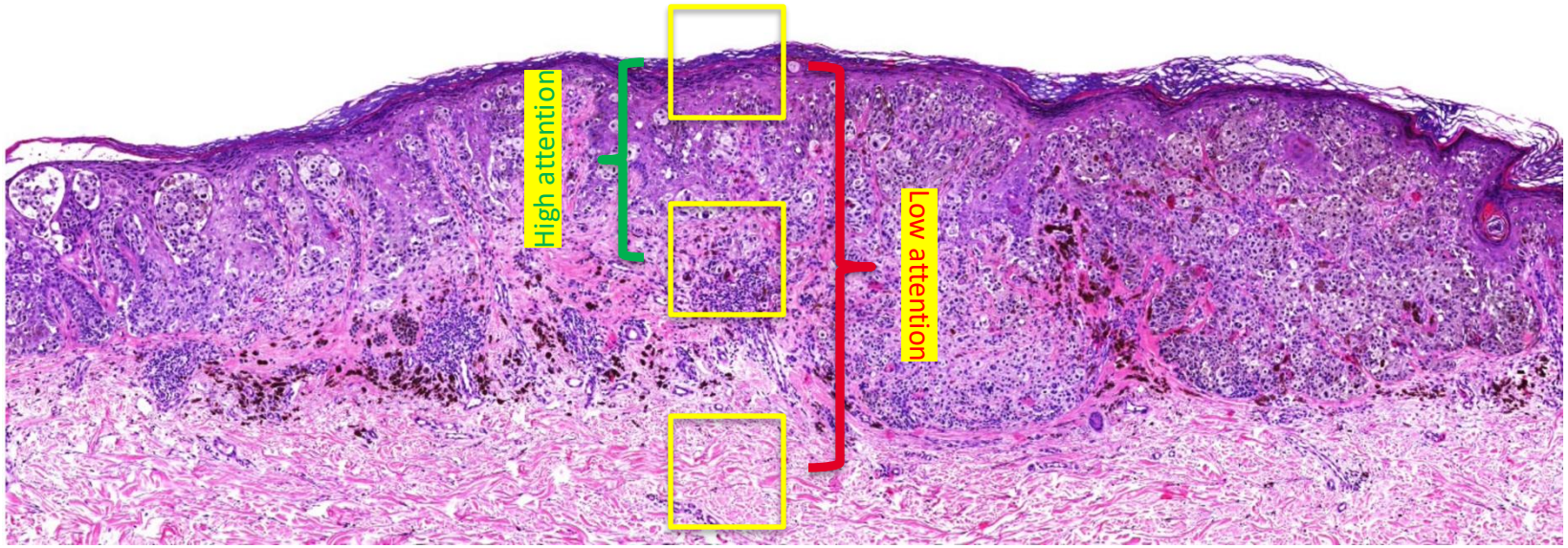
Low attention

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

High attention

Attention in vision (invasive melanoma histology example):





Query: the information I need from other patches.

Key: the information I offer to other patches.

Value: the actual information.

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

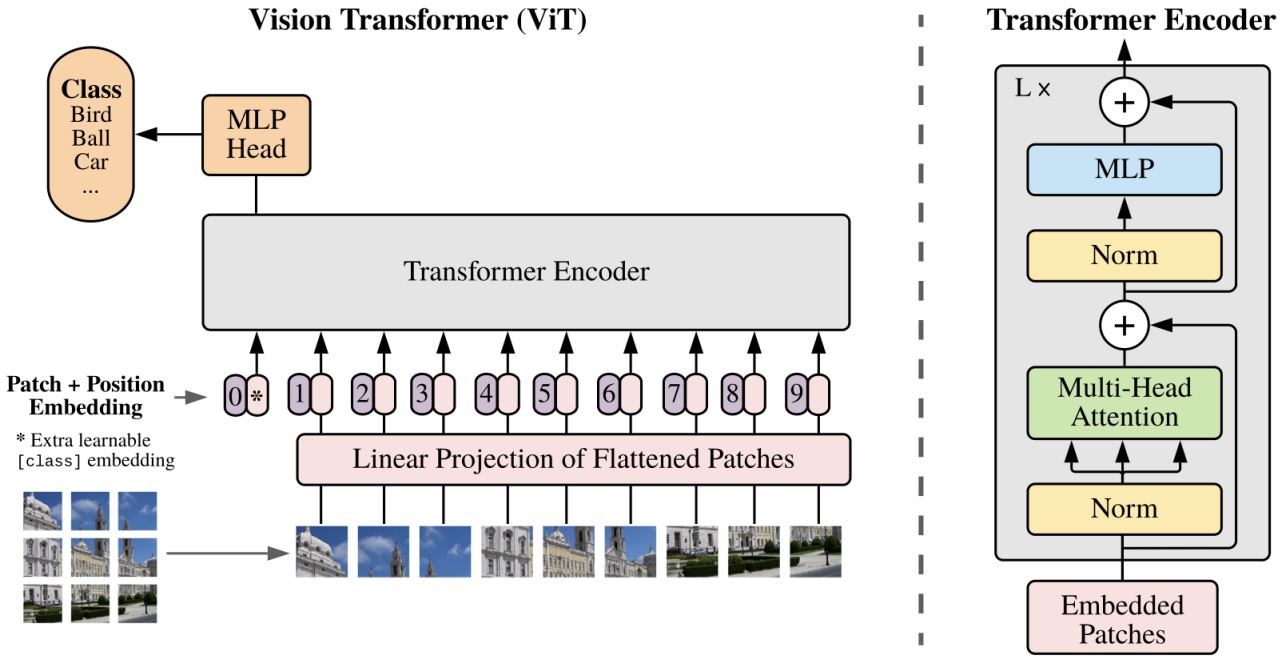
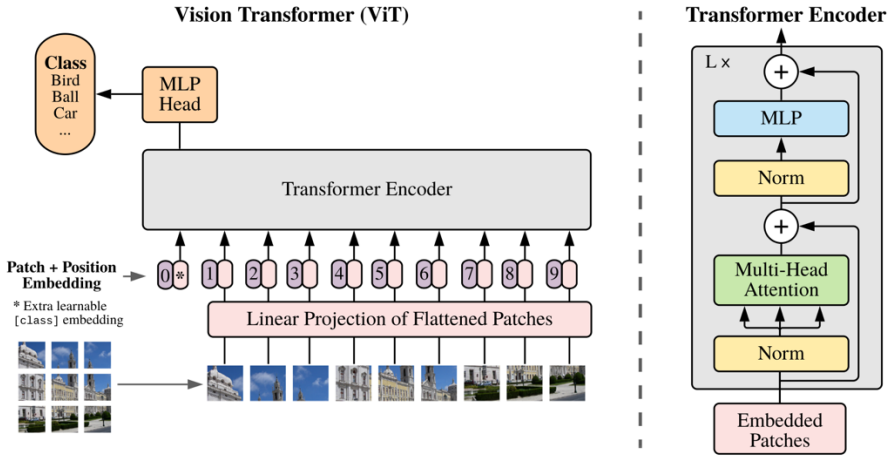


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification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




```
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
```

Topic 1: Dataset size and inductive bias

Q: The authors of the second paper say that, with sufficient large-scale training data, Vision Transformers can outperform convolutional neural networks despite lacking the inductive biases present in CNNs. However, in reality, there is a real scarcity of large-scale training data. Therefore, what will be preferred; techniques with built-in inductive biases, or techniques that rely on large-scale data? And what consequences will this have for future developments?

Q: How do Vision Transformers handle the lack of inductive biases that is present in CNNs?

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

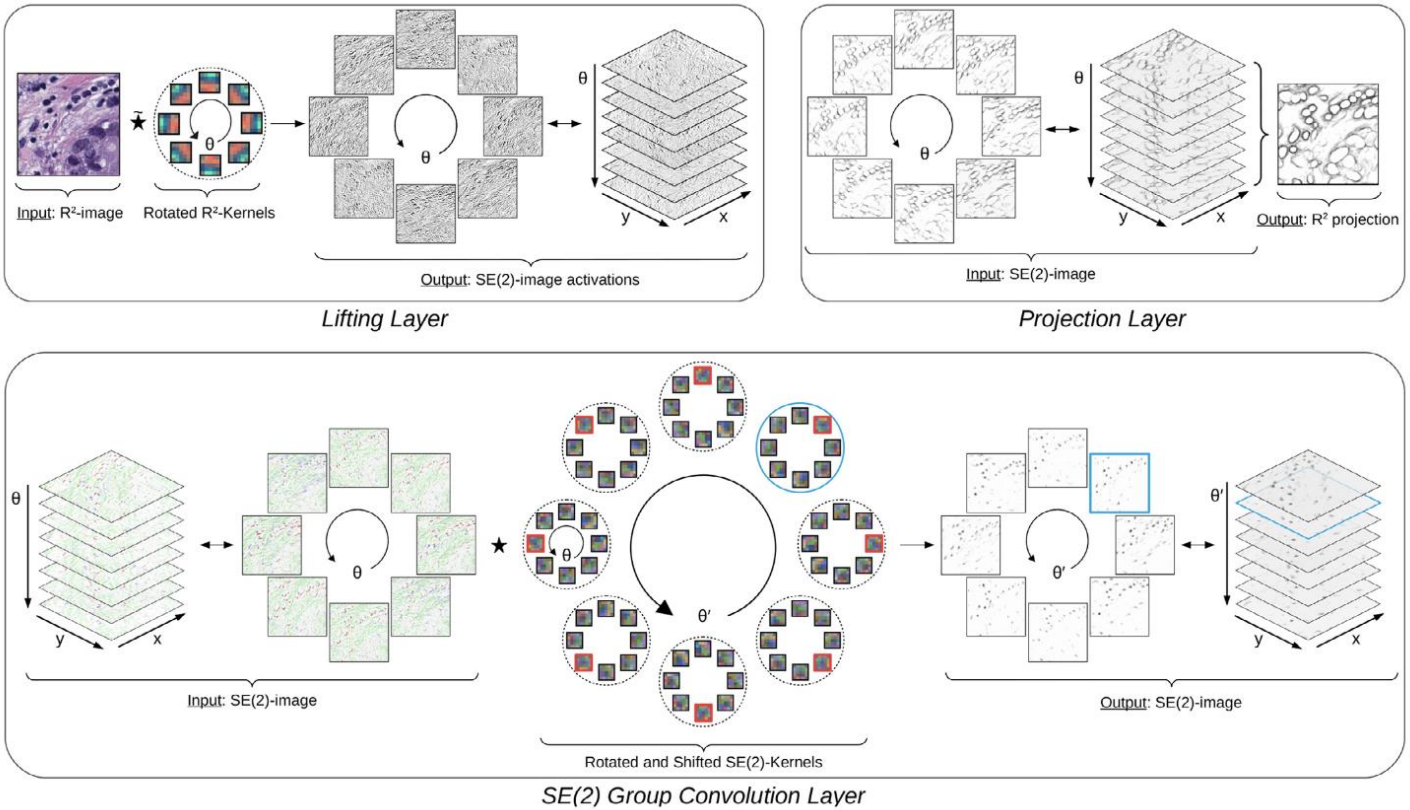
In CNNs:

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

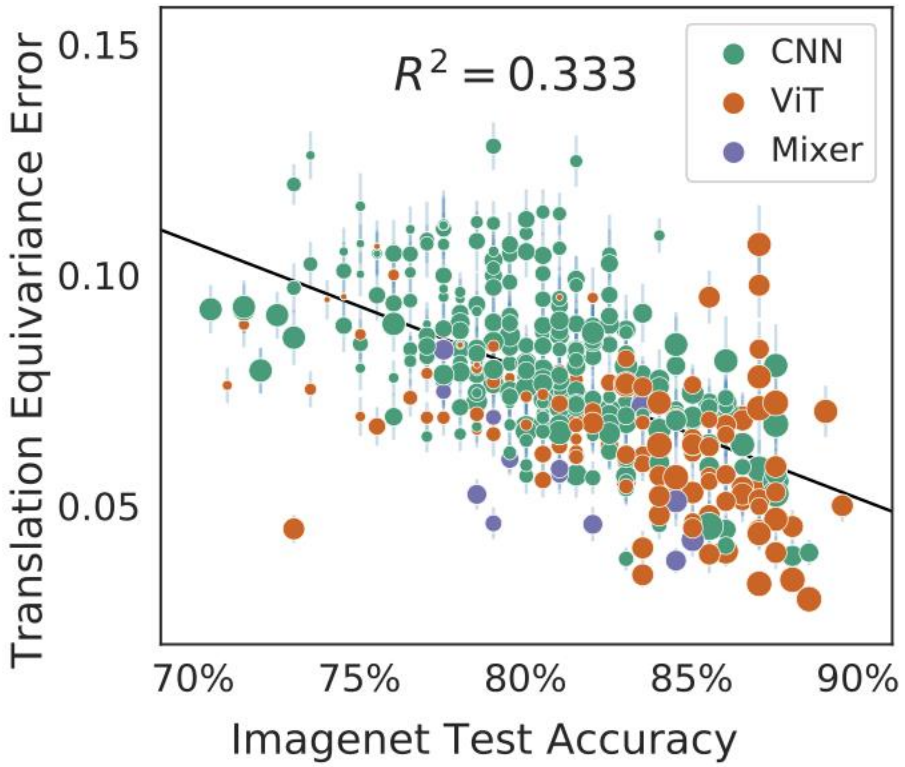
In transformers:

- 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

Topic 2: Transformers and biomedical data

Q: While attention mechanisms in transformers have been successful for 1D (text) and 2D (images) data, how well would these architectures extend to 3D data such as volumetric medical scans ? What architectural changes would be necessary to efficiently model 3D spatial relationships?

Q: What are key challenges when using vision transformers for medical image analysis compared to CNNS, particularly in terms of computational complexity?

Attentions scales quadratically with the number of input tokens. So in 3D we either need more tokens (large memory requirements) or larger patch sizes (poor expresiveness of the models).

Some solutions:

- Hierarchical models (e.g. HIPT)
- Sparse or local attention (i.e. limit the scope of attention)
- Hybrid CNN-transformer models

Topic 3: Self-supervised learning

Q: In scenarios where labeled data is scarce, how does self-supervised learning improve the effectiveness of Transformer models, and what are some common self-supervised tasks used in this context?

Q: The Vision Transformer (ViT) performs well when they are pre-trained on large datasets, while the Hierarchical Vision Transformer (HVT) introduces self-supervised learning to cope with limited labeled data. Why are classical neural networks still used so much when there are these options that use (self) attention which perform better compared to neural networks? Yes they often need more data, but couldn't data augmentation fix that issue?

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.

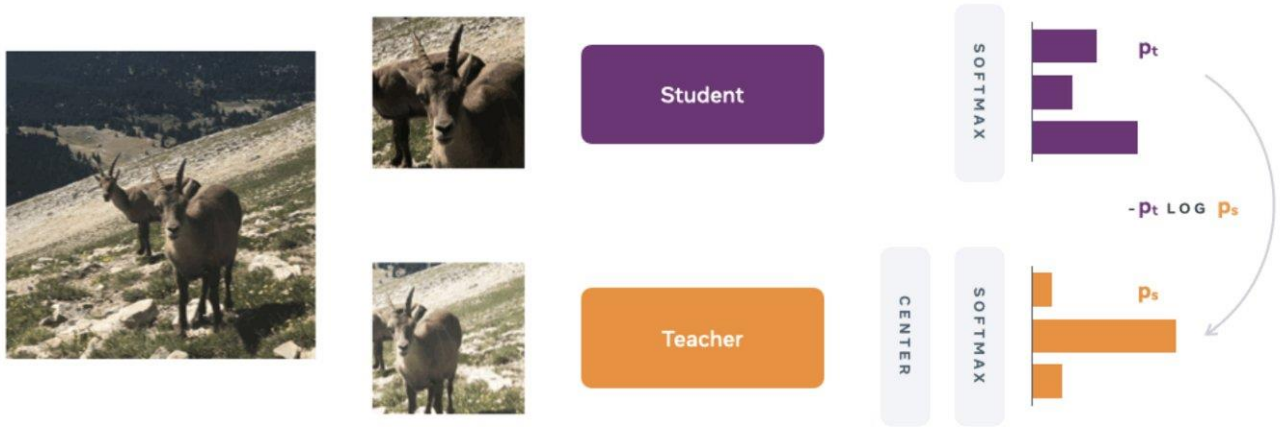
Advantages:

- 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:



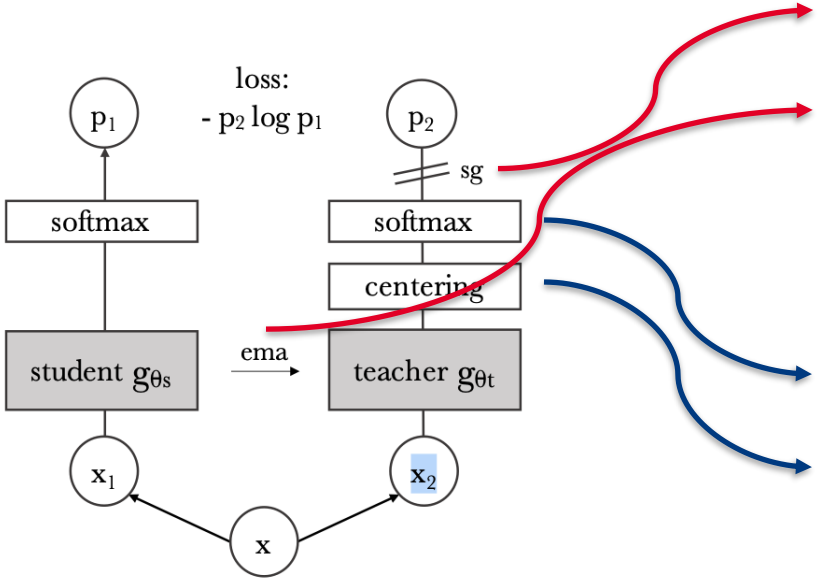


Figure 2: **Self-distillation with no labels.** We illustrate DINO in the case of one single pair of views (x_1, x_2) for simplicity. The model passes two different random transformations of an input image to the student and teacher networks. Both networks have the same architecture but different parameters. 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**

Other questions

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

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

When tokens are embedded in a Transformer model for natural language processing (NLP), a **fixed vocabulary** of tokens is chosen beforehand with a given embedding vector per token. In the multi-headed attention blocks, these embeddings are adjusted based on the context of the preceding tokens. However**, the embedding step is different when you apply a Transformer for computer vision tasks**. Instead of using a "lookup table" to find the embedding for each token, the images are turned into flattened patches and a "trainable linear projection" is used to create the n dimensional embeddings.

Q: Why does the model need to learn how to embed the patches itself, whereas it does not have to do this for tokens in the NLP scenario?

Lookup table for an 16-by-16 RGB image patch, 8 bits per pixel:

$$256^{16 \times 16 \times 3} = 256^{768} = 2^{8 \times 768} = 2^{6144}$$

Number of atoms in the universe:

$$2^{265.75}$$