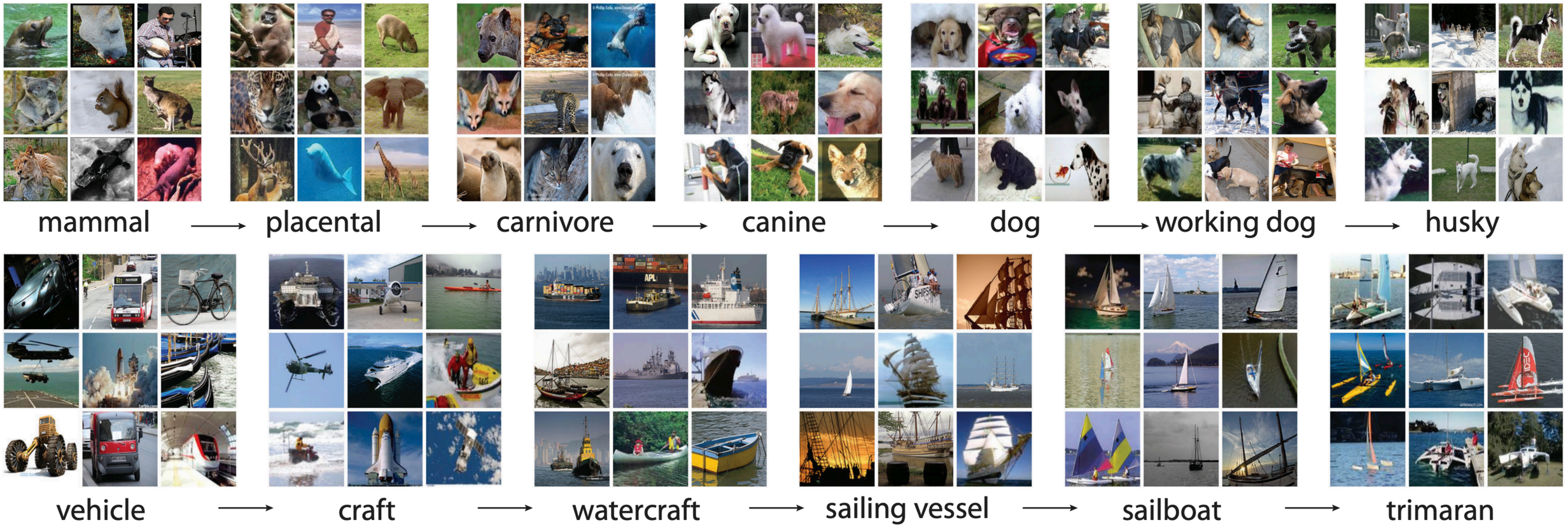
Advanced Computer Vision: Self-Supervised Learning

MLMI17

Ayush Tewari

UNIVERSITY OF CAMBRIDGE

Why is this dataset necessary?



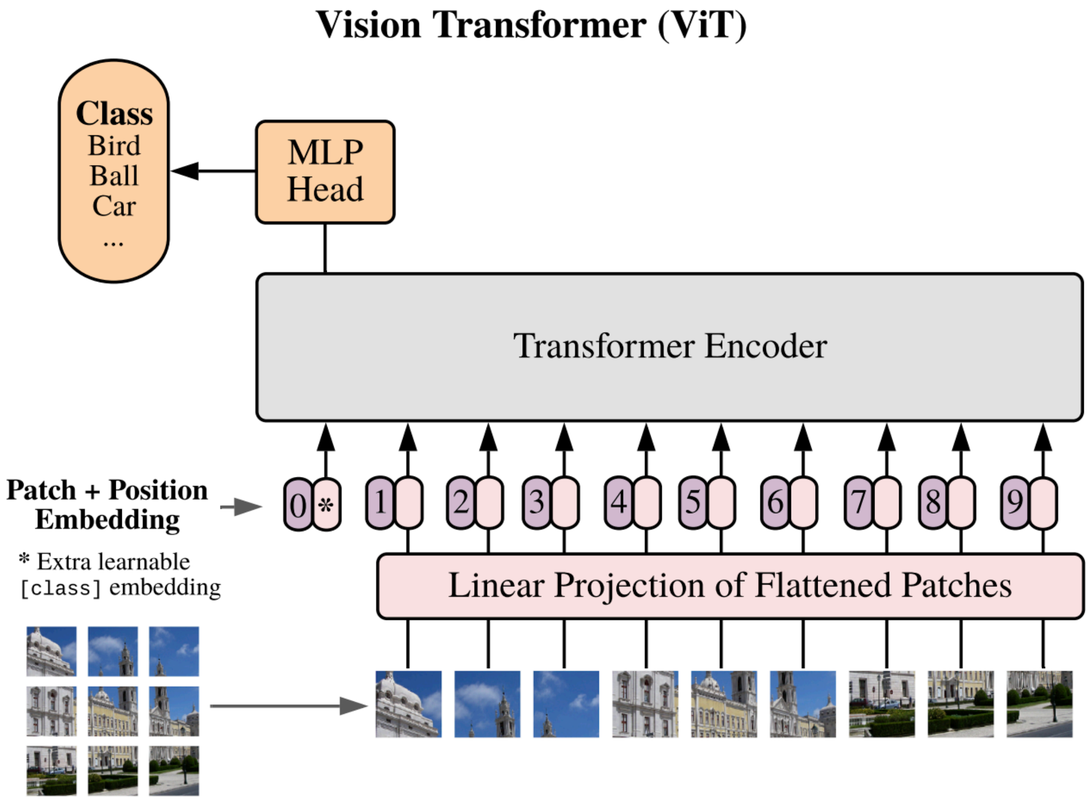
ImageNet [Deng et al., CVPR 2009]

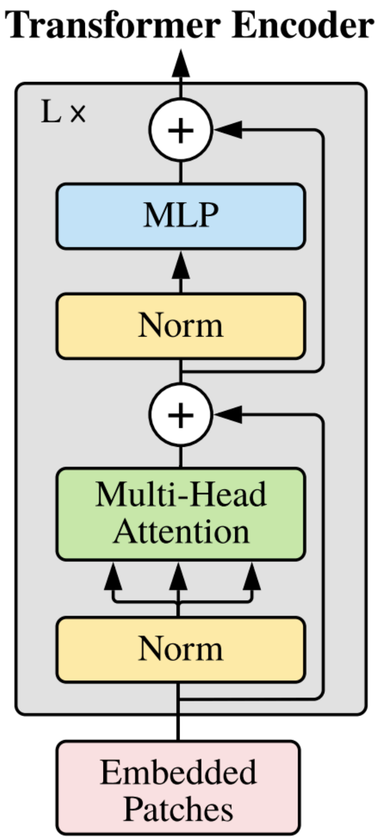
>14M images with human-verified annotations!

Self-Supervised Learning

- Can we learn useful representations from *unlabelled* data?
- What is an ideal representation?
  - Useful for many downstream tasks
- Good representations are
  - Compact (only contains essential information)
  - Predictive (able to take actions that achieve desirable future outcomes)
  - Disentangled (independent factors)
  - Interpretable
  - ….

Detour: Vision Transformers





- Patches
  - Positional Encoding
- Global [CLS] token
- MLP Head

What is self supervision?

- A general recipe:
  - Collect large quantities of unlabelled data
  - Define an auxiliary task
  - Train a model so solve this task
  - **Pray**

Three classes of self-supervision

- Generative
- Contrastive
- Distillation

Three classes of self-supervision

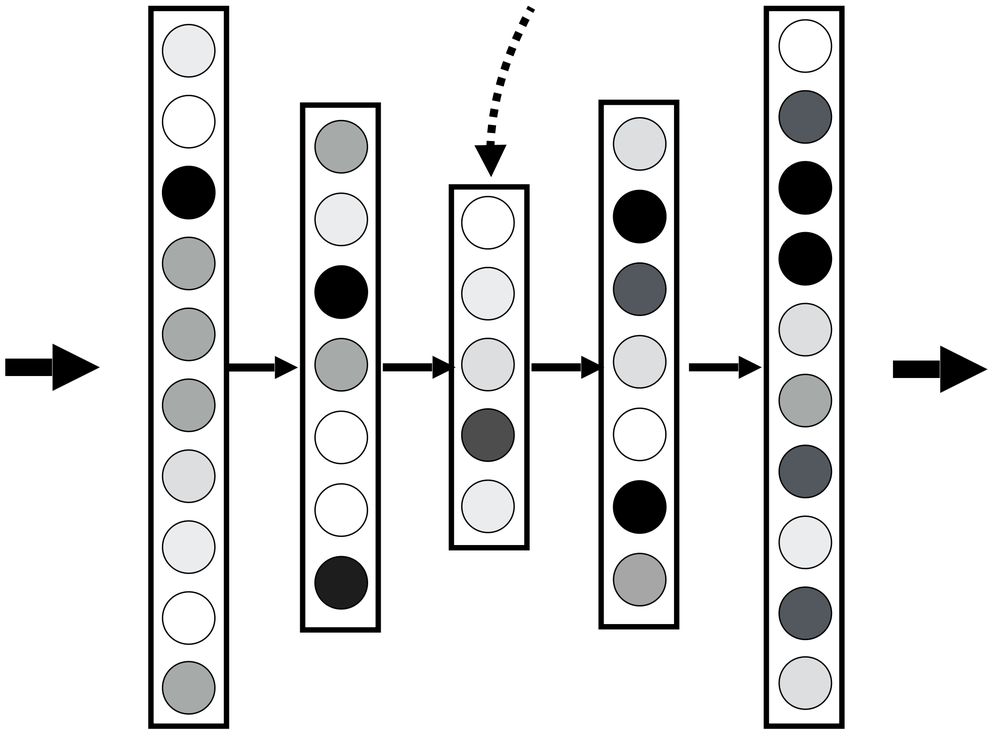
- **Generative**
- Contrastive
- Distillation

Autoencoder

$\mathbf{X}$

compressed image code

(vector **z**)

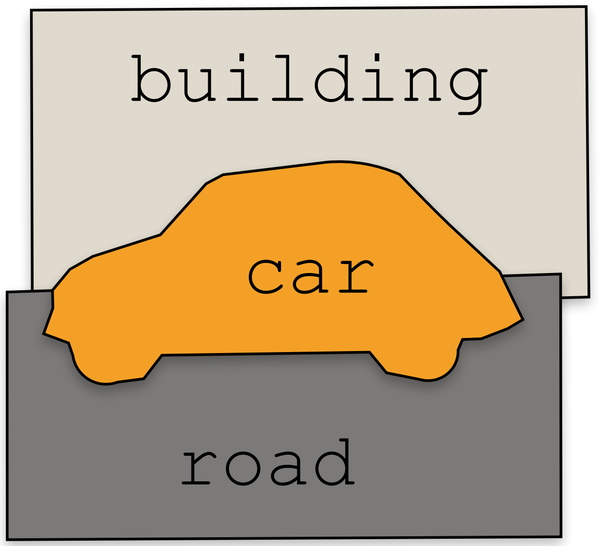


$\hat{\mathbf{X}}$

Reconstructed image

Algorithmic Information Theory

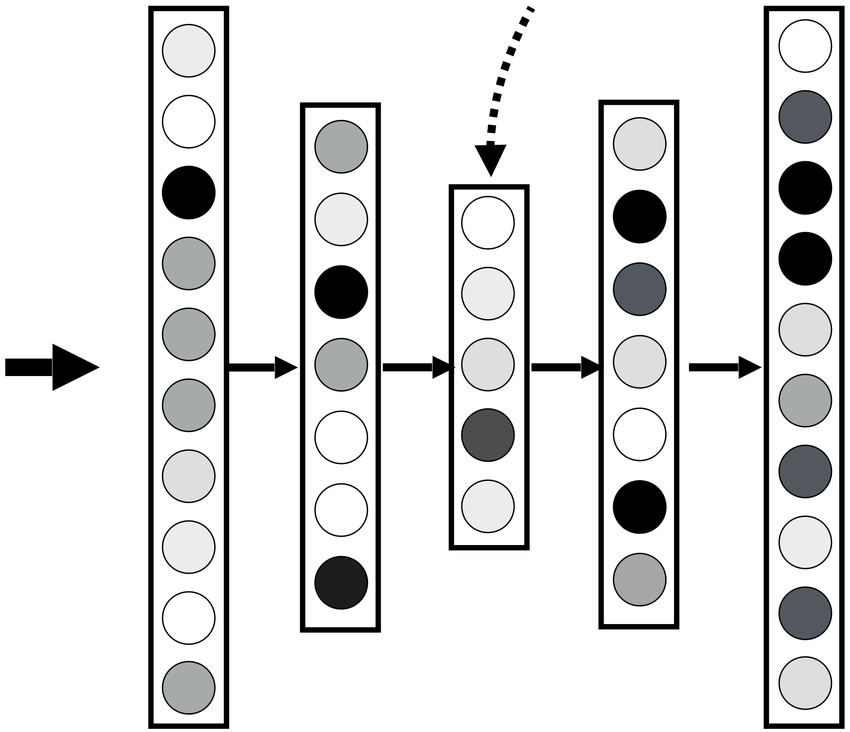
- “Compression is intelligence”
- Occam’s razor - Given multiple hypotheses that are consistent with the data, the simplest should be preferred
- Solomonoff theory of inductive inference - “the best possible scientific model is the shortest algorithm that generates the empirical data under consideration”



Autoencoder

$\mathbf{X}$

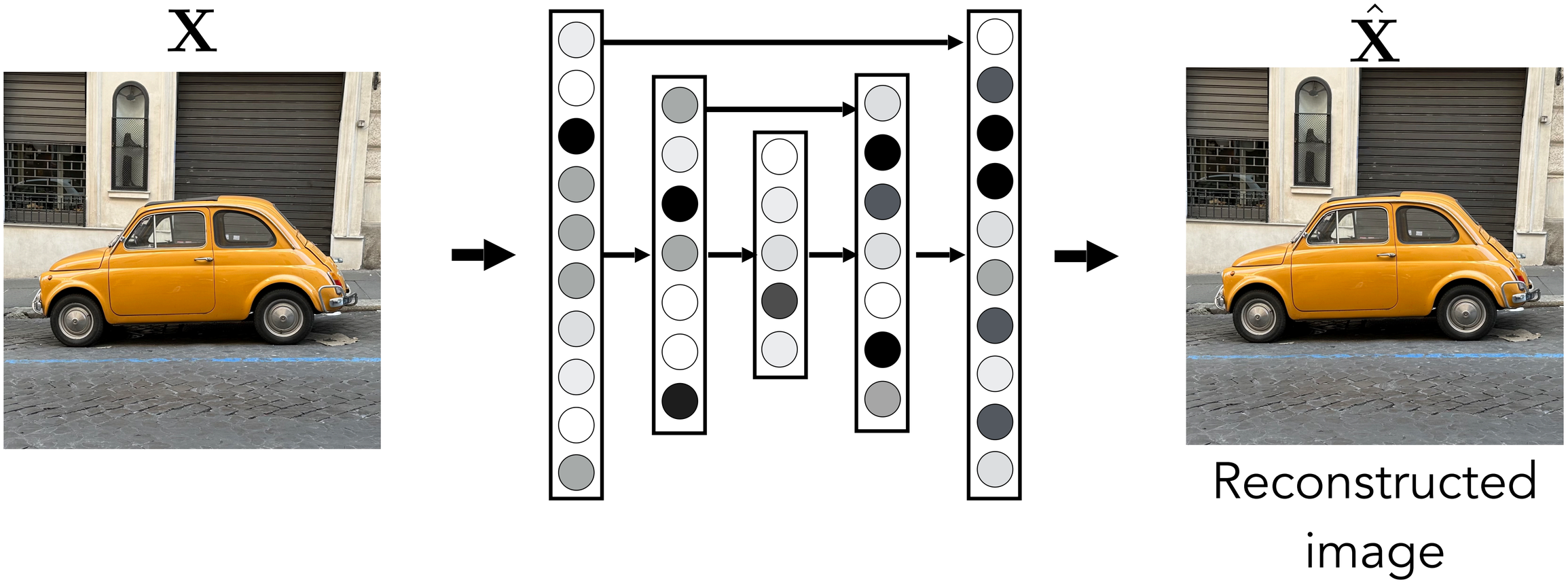
compressed image code
(vector **z**)



$\hat{\mathbf{X}}$

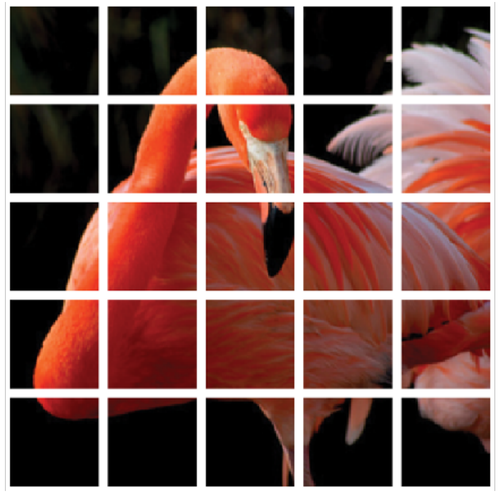
Reconstructed image

Autoencoder with skip connections

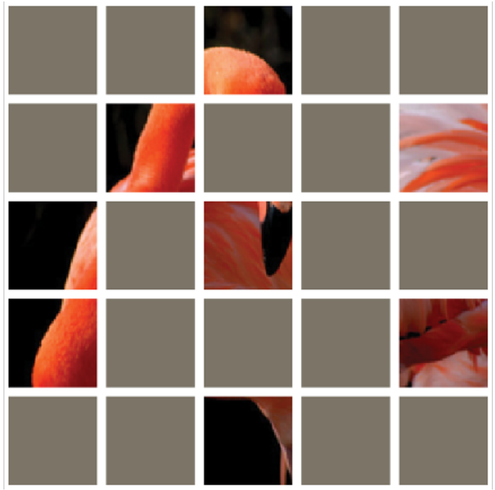


Why does this model not learn good representations?

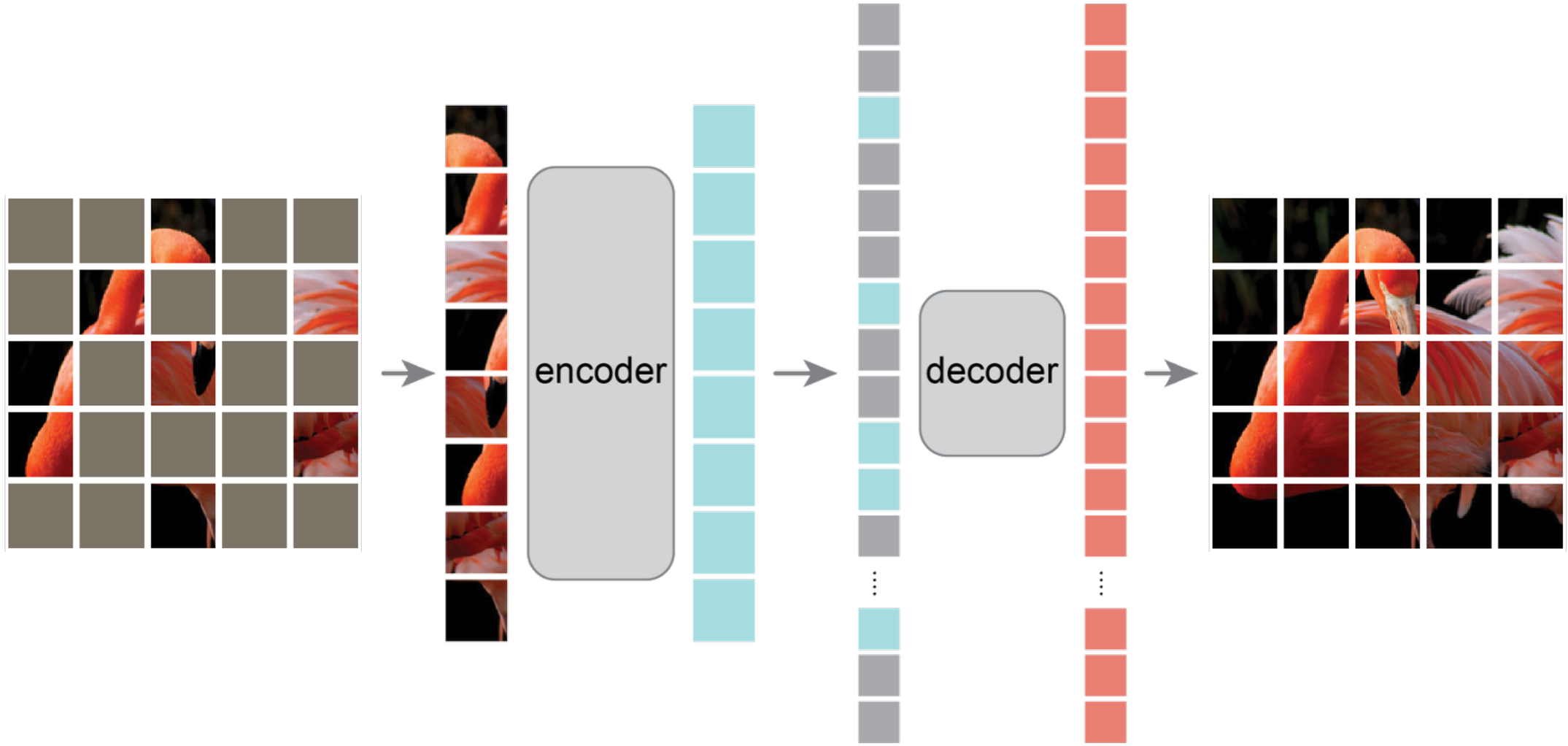
Masked Autoencoders



Masked Autoencoders



Masked Autoencoders



Masked Autoencoders

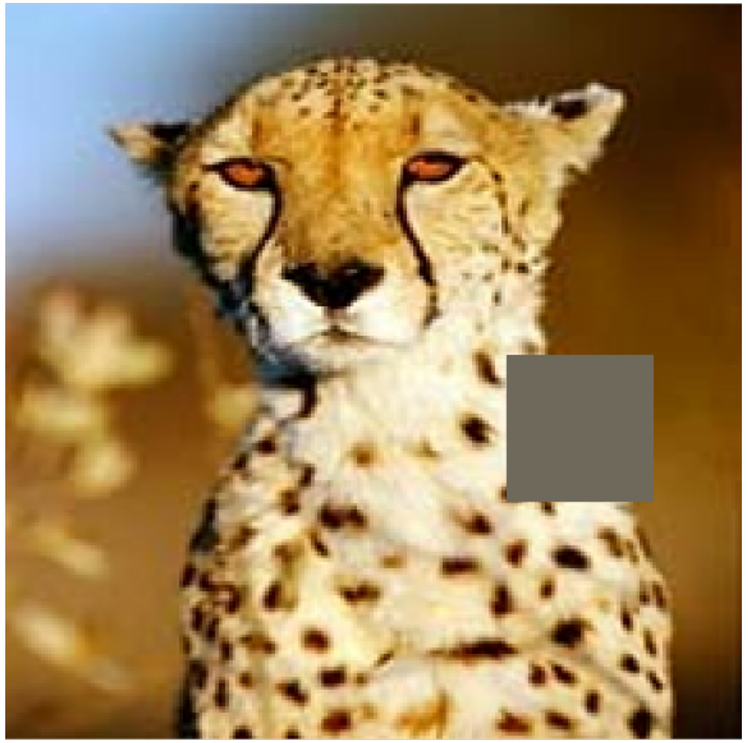
Masked Autoencoders Are Scalable Vision Learners

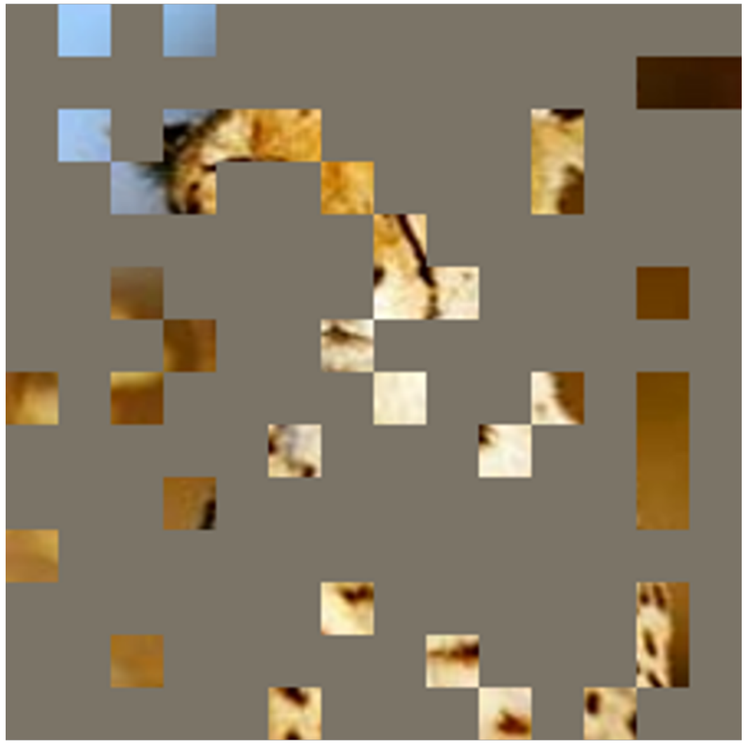
Kaiming He*,† Xinlei Chen* Saining Xie Yanghao Li Piotr Dollár Ross Girshick

*equal technical contribution †project lead

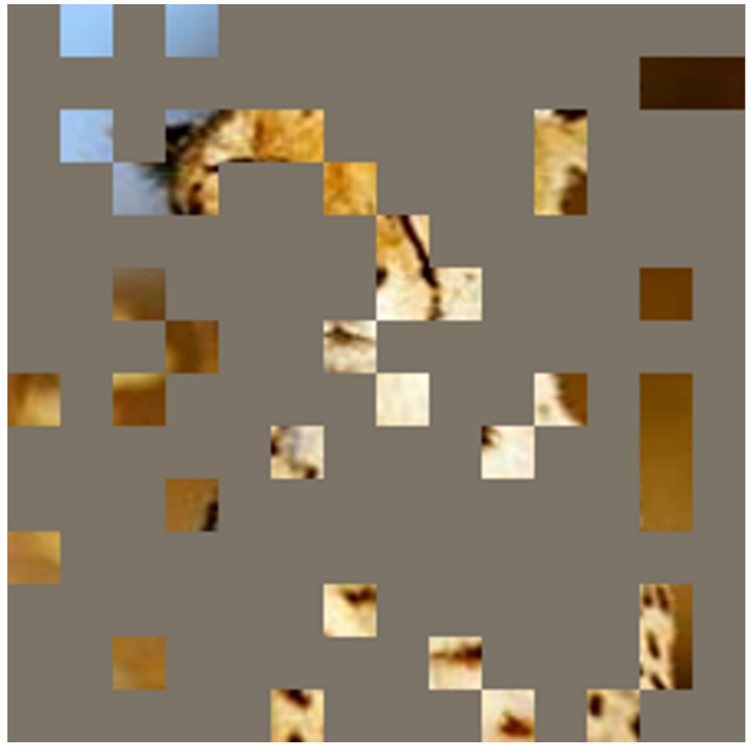
Facebook AI Research (FAIR)

Masked Autoencoders



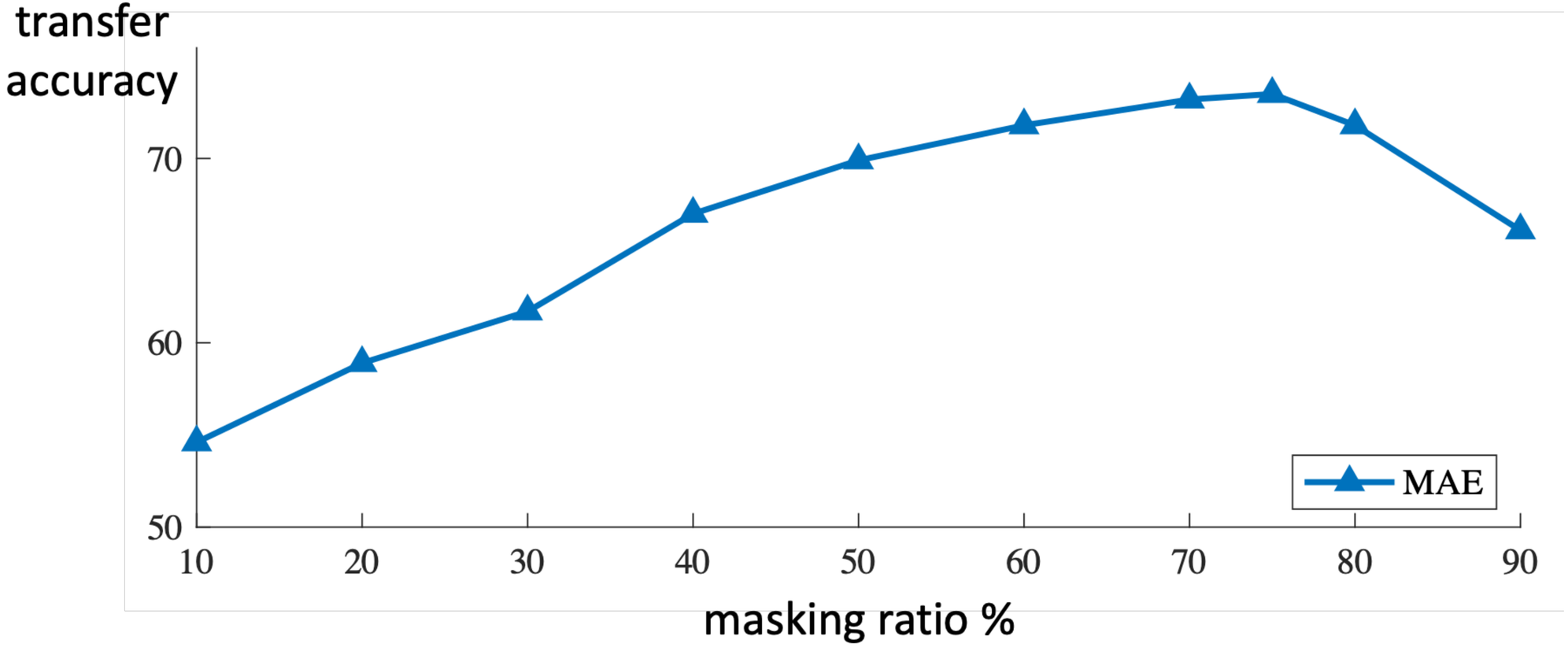


Masked Autoencoders

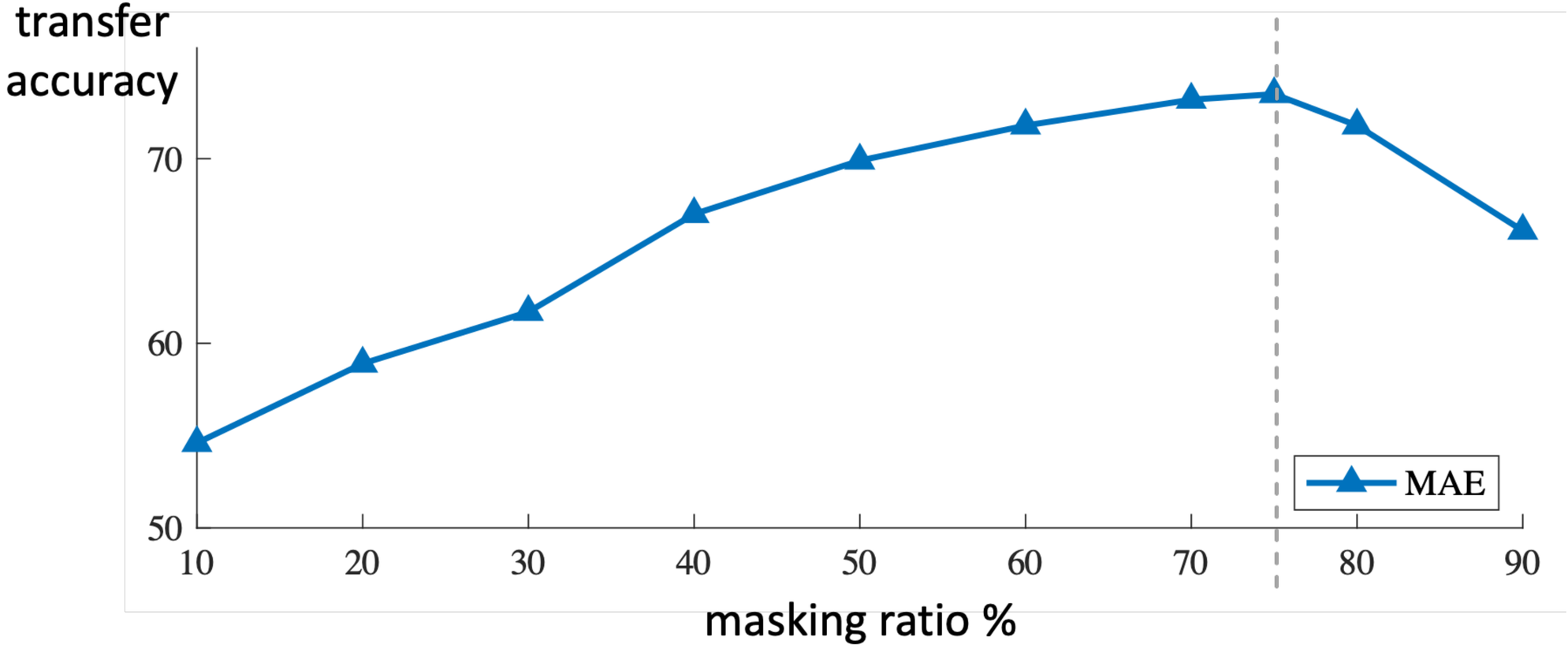


Masked Autoencoders

Masked Autoencoders



Masked Autoencoders



The __ opened their __ and began to__

students .. books .. read

How do we evaluate the quality?

- Linear probing
  - Linear transform from representation to the labels
  - Train with softmax

It is a common practice to normalize the classifier input when training a classical linear classifier (*e.g.*, SVM [11]). Similarly, it is beneficial to normalize the pre-trained features when training the linear probing classifier. Following [15], we adopt an extra BatchNorm layer [31] without affine transformation (`affine=False`). This layer is applied on the pre-trained features produced by the encoder, and is before the linear classifier. We note that the layer does *not* break the linear property, and it can be absorbed into the linear classifier after training: it is essentially a re-parameterized linear classifier.[3] Introducing this layer helps calibrate the feature magnitudes across different variants in our ablations, so that they can use the same setting without further *lr* search.

How do we evaluate the quality?

- Linear probing
  - Linear transform from representation to the labels
- Fine tuning

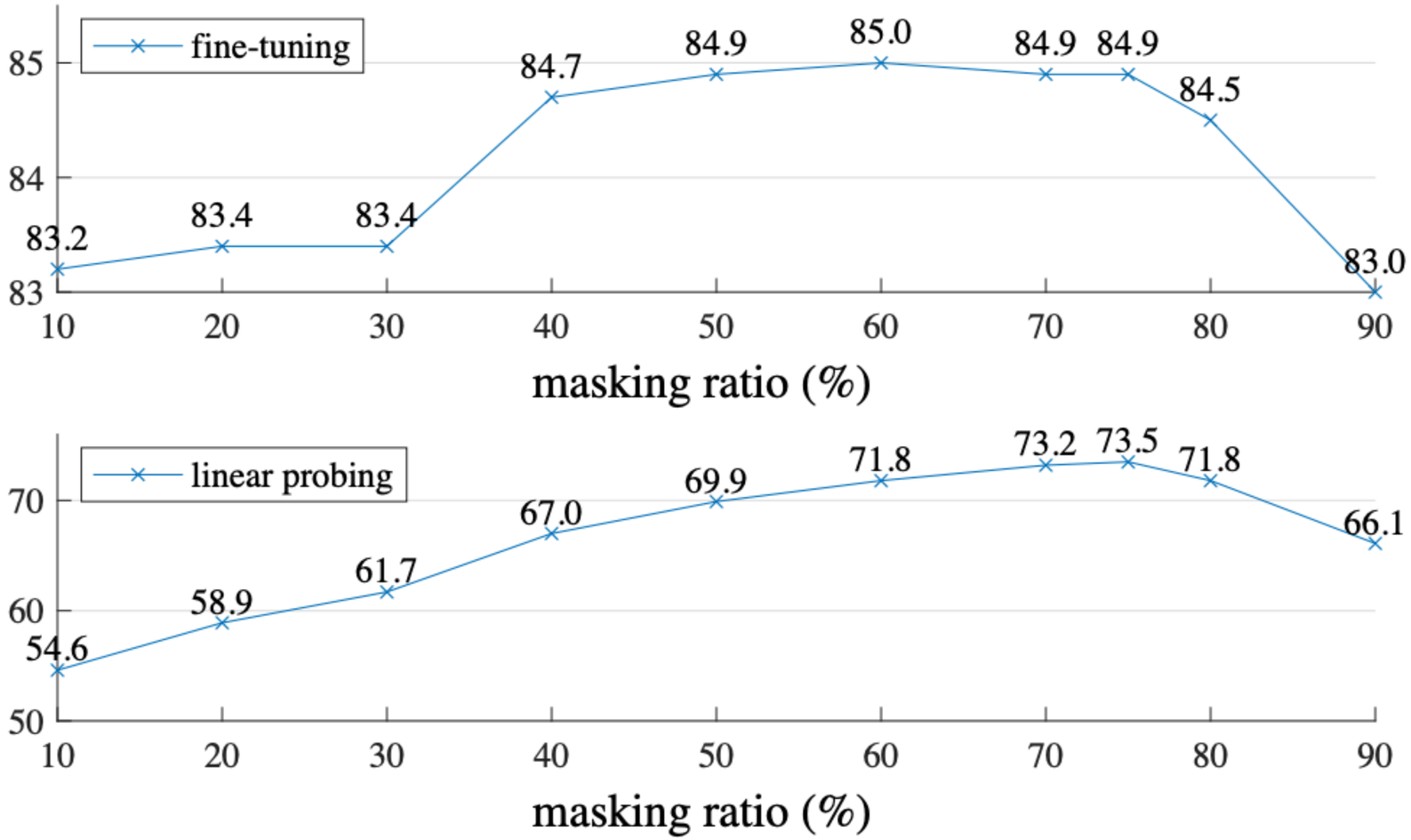
Table 1 shows that linear probing and fine-tuning results are largely *uncorrelated*. Linear probing has been a popular protocol in the past few years; however, it misses the opportunity of pursuing *strong but non-linear* features—which is indeed a strength of deep learning. As a middle ground, we

blocks	ft	lin
1	84.8	65.5
2	**84.9**	70.0
4	**84.9**	71.9
8	**84.9**	**73.5**
12	84.4	73.3

(a) **Decoder depth**. A deep decoder can improve linear probing accuracy.

- Why does linear probing performance increase with depth?

How do we evaluate the quality?

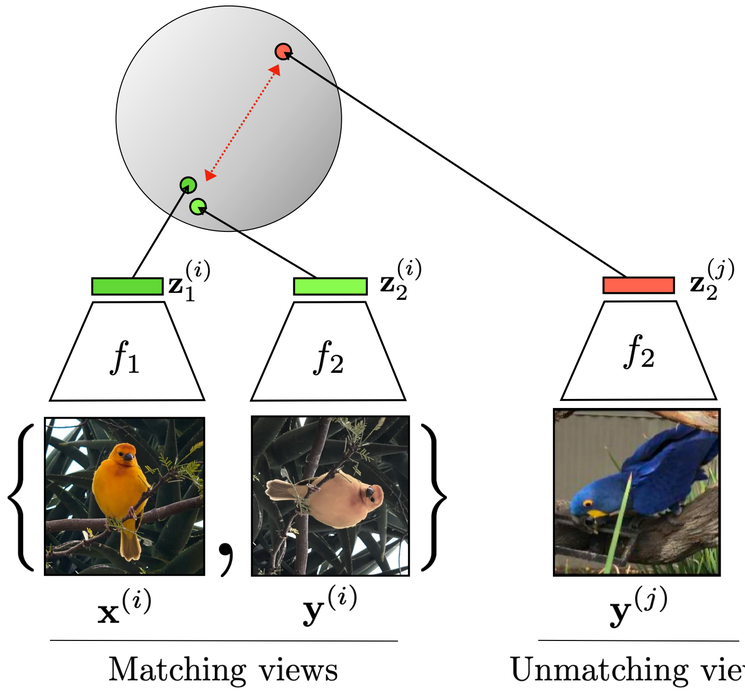


Three classes of self-supervision

- Generative
- **Contrastive**
- Distillation

Contrastive

- Do we need to preserve all information in the image?
  - What if I do not care about low-level details?
- Contrastive learning!
  - Augmentations of an image should have similar representations



Data

$\{\mathbf{x}^{(i)}\}_{i=1}^{N}, T \rightarrow$

Constrastive learning (transformations)

Objective

$\sum_{i,j} D(f(T(\mathbf{x}^{(i)})), f(\mathbf{x}^{(i)}))$

$-D(f(\mathbf{x}^{(i)}), f(\mathbf{x}^{(j)}))$

Hypothesis space

$f : \mathbb{R}^N \rightarrow \mathbb{R}^M$

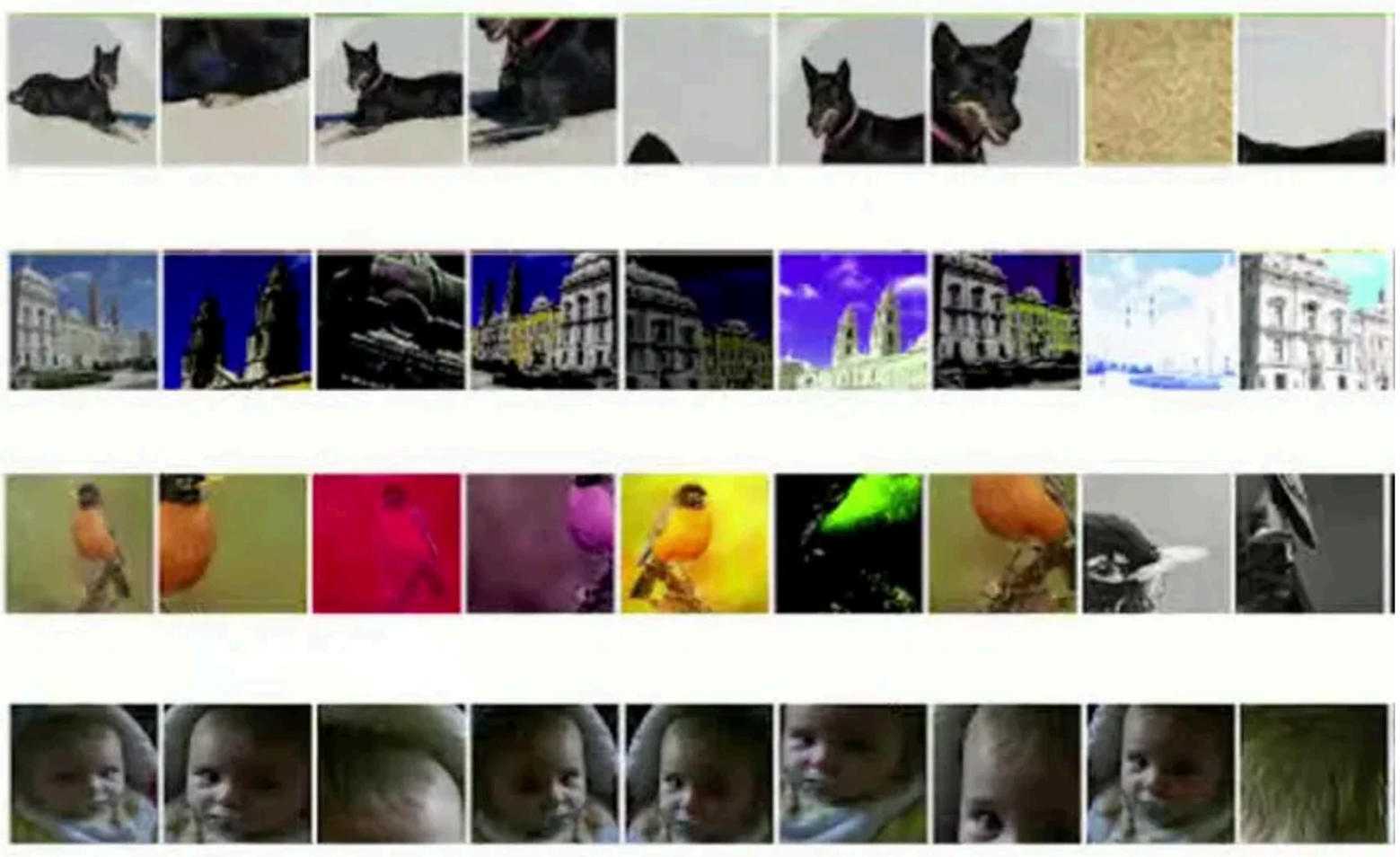
$\rightarrow f$

Contrastive

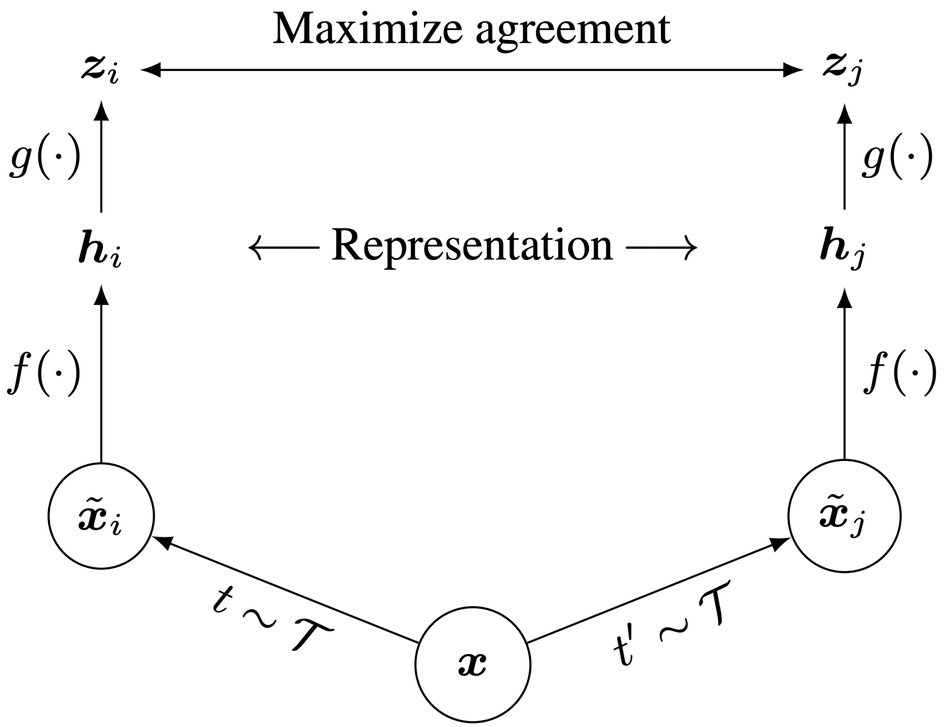
A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen [1] **Simon Kornblith** [1] **Mohammad Norouzi** [1] **Geoffrey Hinton** [1]

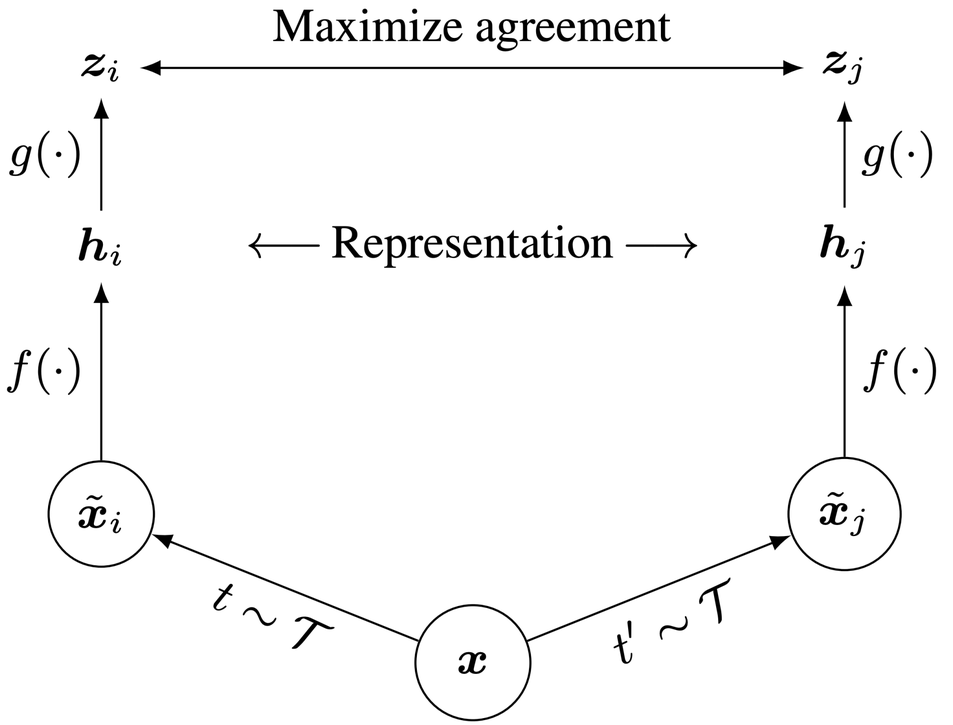
SimCLR - Augmentations

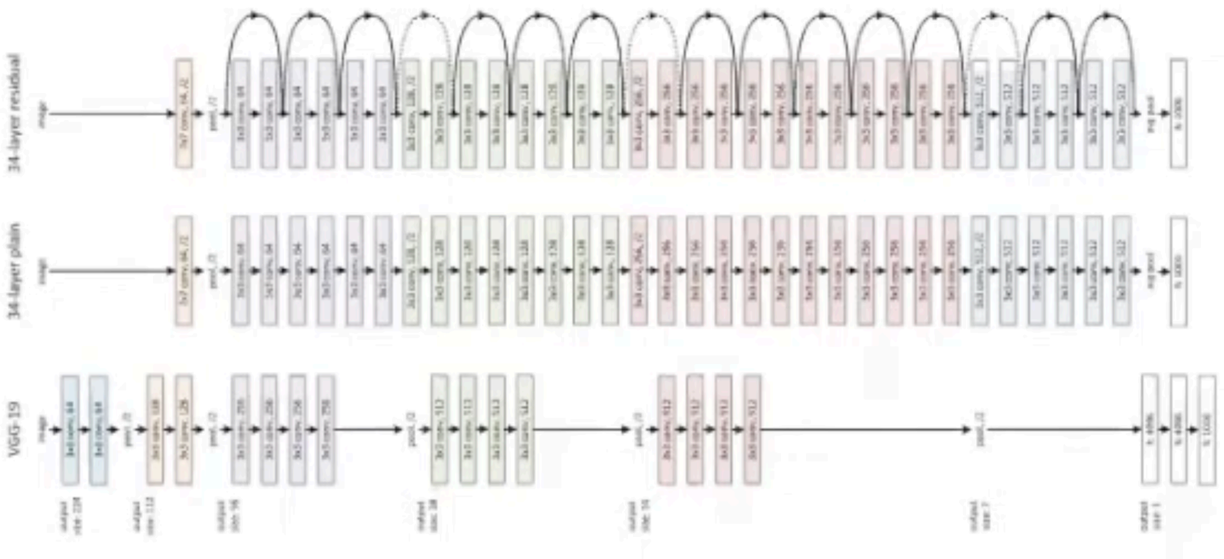


SimCLR - Architecture

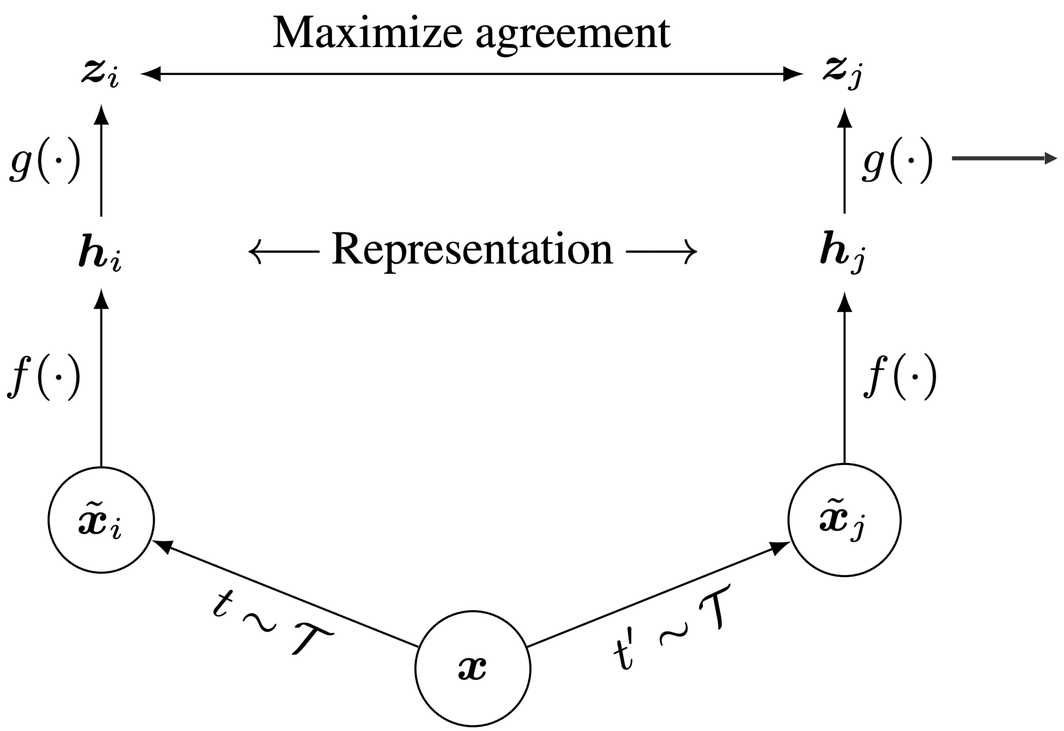


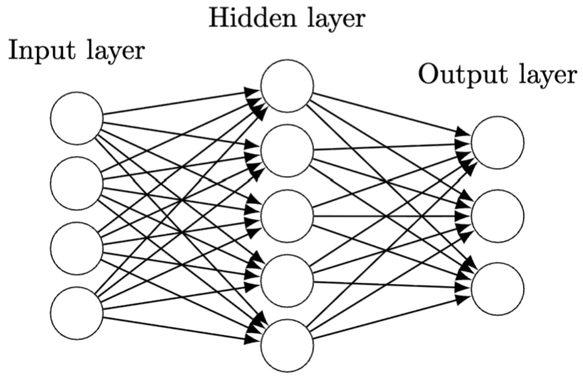
SimCLR - Architecture



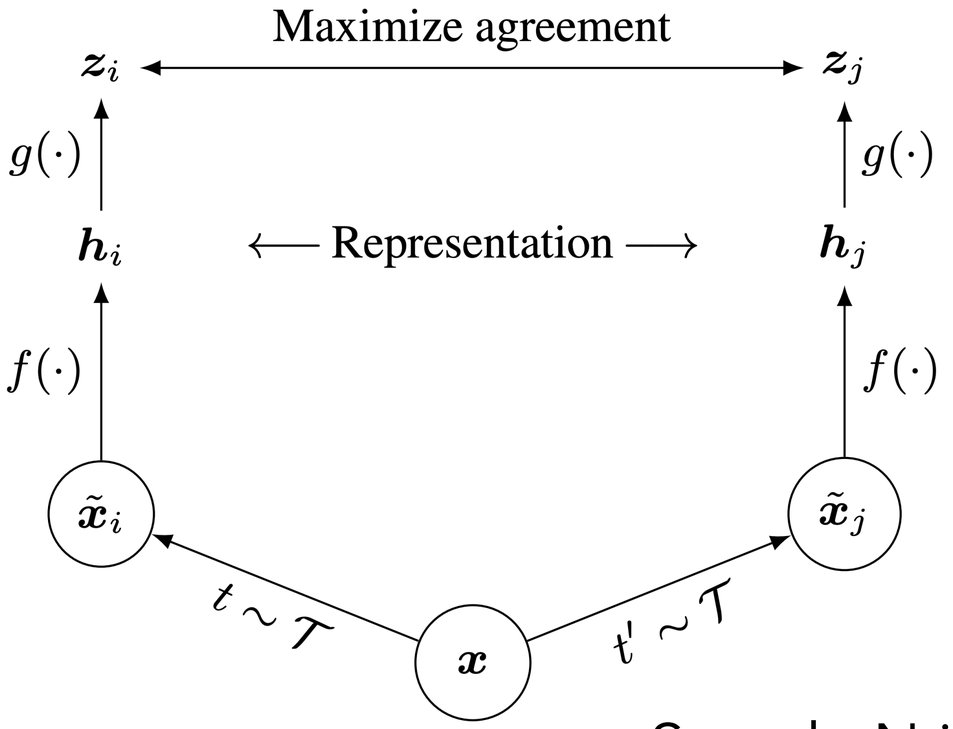


SimCLR - Architecture





SimCLR - Loss Function



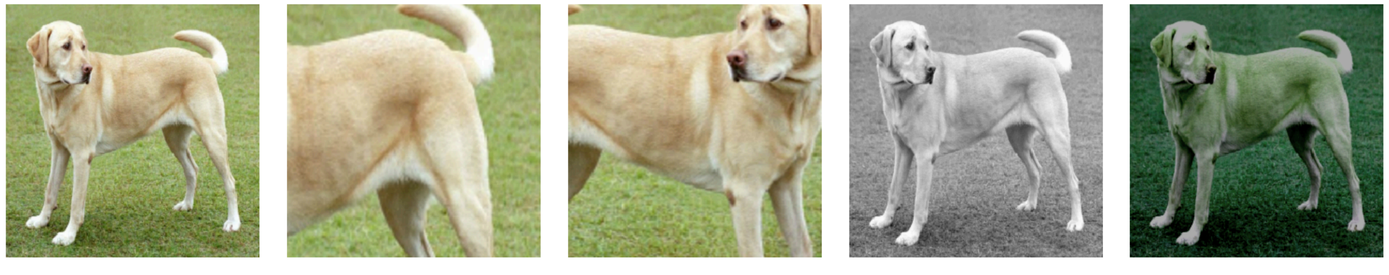
$$\text{sim}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^\top \boldsymbol{v} / \|\boldsymbol{u}\| \|\boldsymbol{v}\|$$

$$\ell_{i,j} = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}$$

What does temperature do?

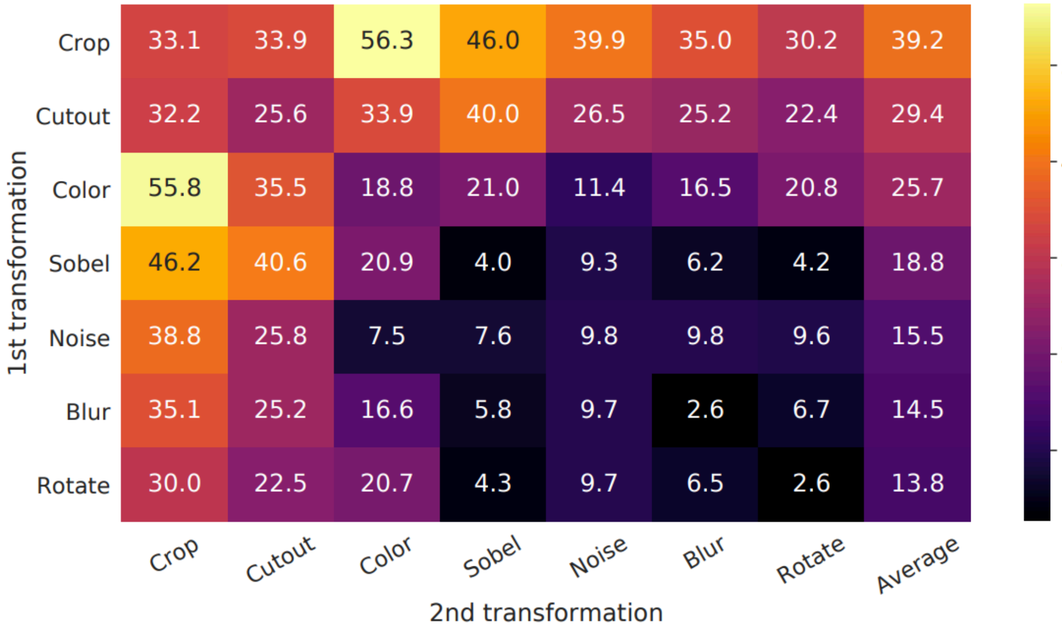
- Sample N images in a batch
- Augment every image -> 2N total image
- One positive example for every image, rest (2N-1) are negative

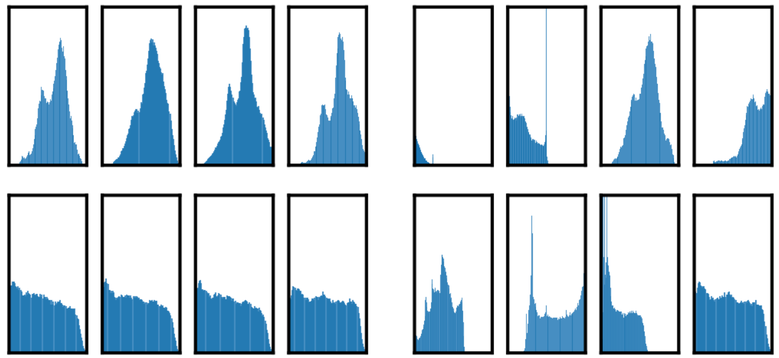
SimCLR - Augmentations



(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

(f) Rotate $\{90°, 180°, 270°\}$ (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering





(a) Without color distortion. (b) With color distortion.

Figure 6. Histograms of pixel intensities (over all channels) for different crops of two different images (i.e. two rows). The image for the first row is from Figure 4. All axes have the same range.

Contrastive

- Do we need to preserve all information in the image?
    - What if I do not care about low-level details?
- Contrastive learning!
    - Augmentations of an image should have similar representations
- **Think of invariances**

Three classes of self-supervision

- Generative
- Contrastive
- **Distillation**

Distillation-based Self-Supervision

- Starts from knowledge distillation

Distilling the Knowledge in a Neural Network

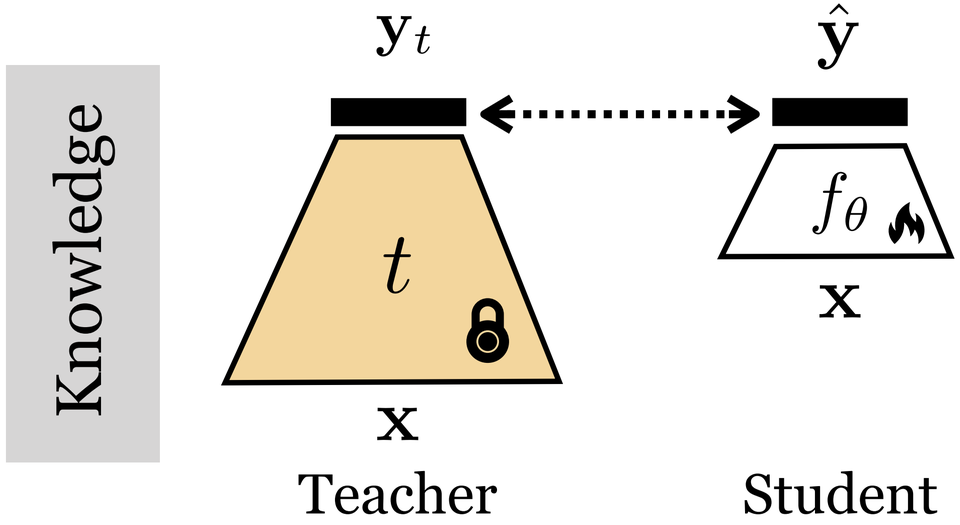
Geoffrey Hinton[*†]
Google Inc.
Mountain View
geoffhinton@google.com

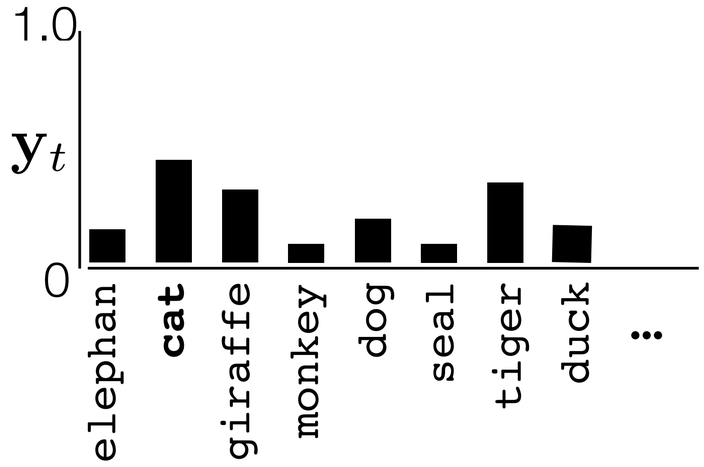
Oriol Vinyals[†]
Google Inc.
Mountain View
vinyals@google.com

Jeff Dean
Google Inc.
Mountain View
jeff@google.com

Many insects have a larval form that is optimized for extracting energy and nutrients from the environment and a completely different adult form that is optimized for the very different requirements of traveling and reproduction. In large-scale machine learning, we typically use very similar models for the training stage and the deployment stage despite their very different requirements: For tasks like speech and object recognition, training must extract structure from very large, highly redundant datasets but it does not need to operate in real time and it can use a huge amount of computation. Deployment to a large number of users, however, has much more stringent requirements on latency and computational resources. The analogy with insects suggests that we should be willing to train very cumbersome models if that makes it easier to extract structure from the data. The cumbersome model could be an ensemble of separately trained models or a single very large model trained with a very strong regularizer such as dropout [9]. Once the cumbersome model has been trained, we can then use a different kind of training, which we call "distillation" to transfer the knowledge from the cumbersome model to a small model that is more suitable for deployment. A version of this

Knowledge Distillation





$$\{\mathbf{x}^{(i)}\}_{i=1}^{N}$$
$$t : \mathcal{X} \to \triangle^{K-1}$$

$\longrightarrow$

Knowledge Distillation

Cross-entropy Objective

$$H(t(\mathbf{x}), f_\theta(\mathbf{x}))$$

Hypothesis space

$$f_\theta : \mathcal{X} \to \triangle^{K-1}$$

$\longrightarrow$ f_θ

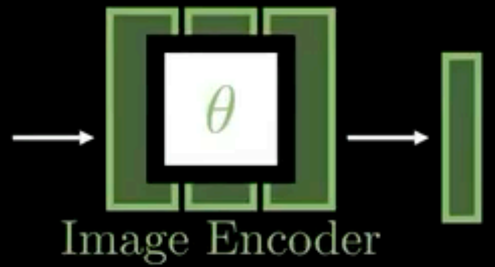
Knowledge **di**stillation *with* ***no*** *labels* (DINO)

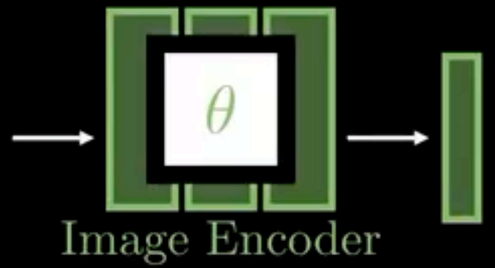
Unsupervised Learning of Visual Features by Contrasting Cluster Assignments

Mathilde Caron[1,2] **Ishan Misra**[2] **Julien Mairal**[1]

Priya Goyal[2] **Piotr Bojanowski**[2] **Armand Joulin**[2]

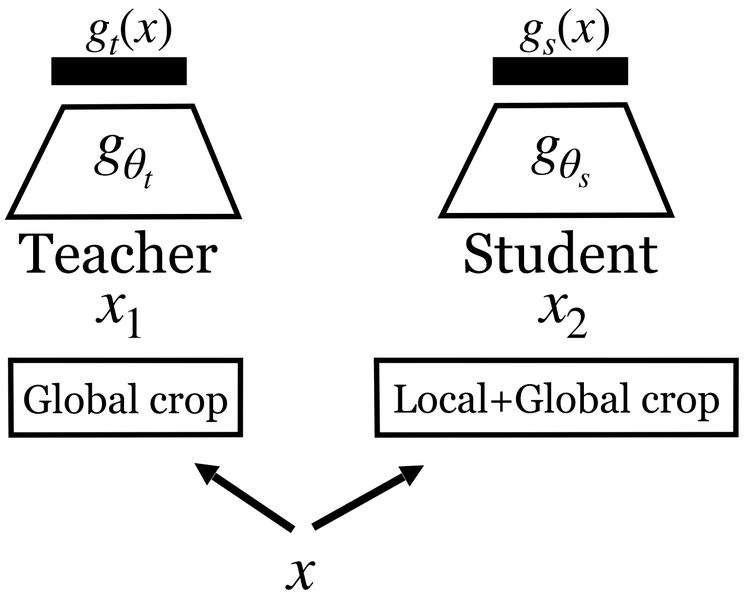
[1] Inria* [2] Facebook AI Research


θ
Image Encoder


θ
Image Encoder

Knowledge **di**stillation *with **no** labels* (DINO)

- Teacher and student have identical architecture
- Crops
  - Two global (>50%) and several local views (<50%)
  - Teacher only sees global views

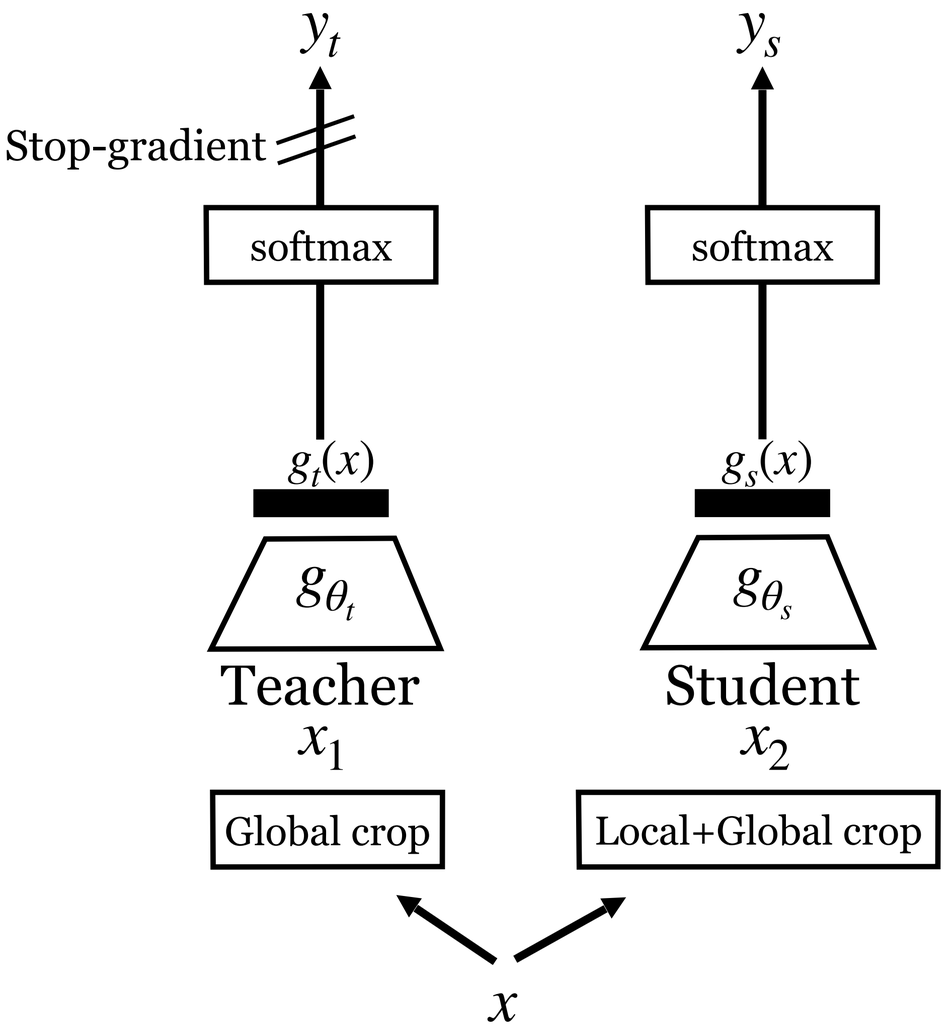



Orig. image


Global crop

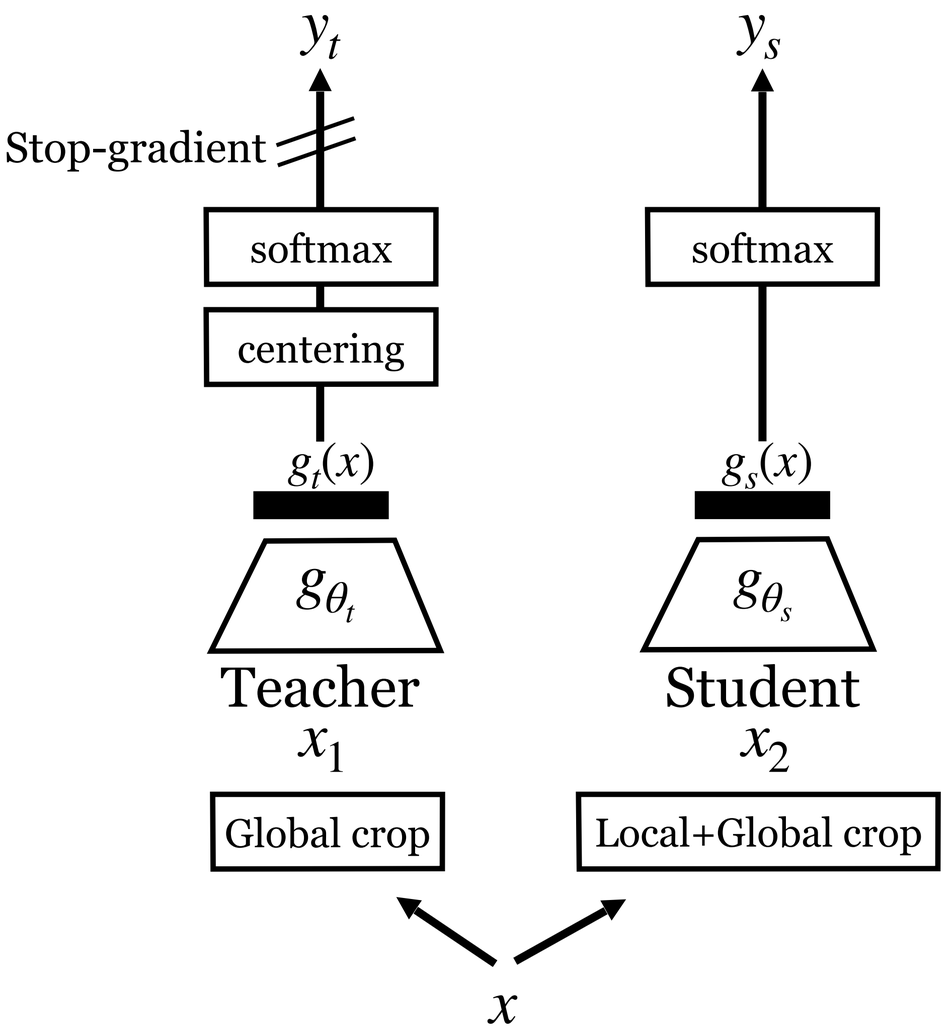

Local crop

Knowledge **di**stillation *with **no** labels* (DINO)



- Cross-entropy loss $H(y_t, y_s)$
  - Softmax applied before loss
- We do not have a teacher?!
  - Teacher is an exponential moving average (EMA) of the student $\theta_t = \lambda\theta_t + (1-\lambda)\theta_s$

Knowledge **di**stillation *with **no** labels* (DINO)



- What if output labels all collapse?
- Centering (all images are in the same class)

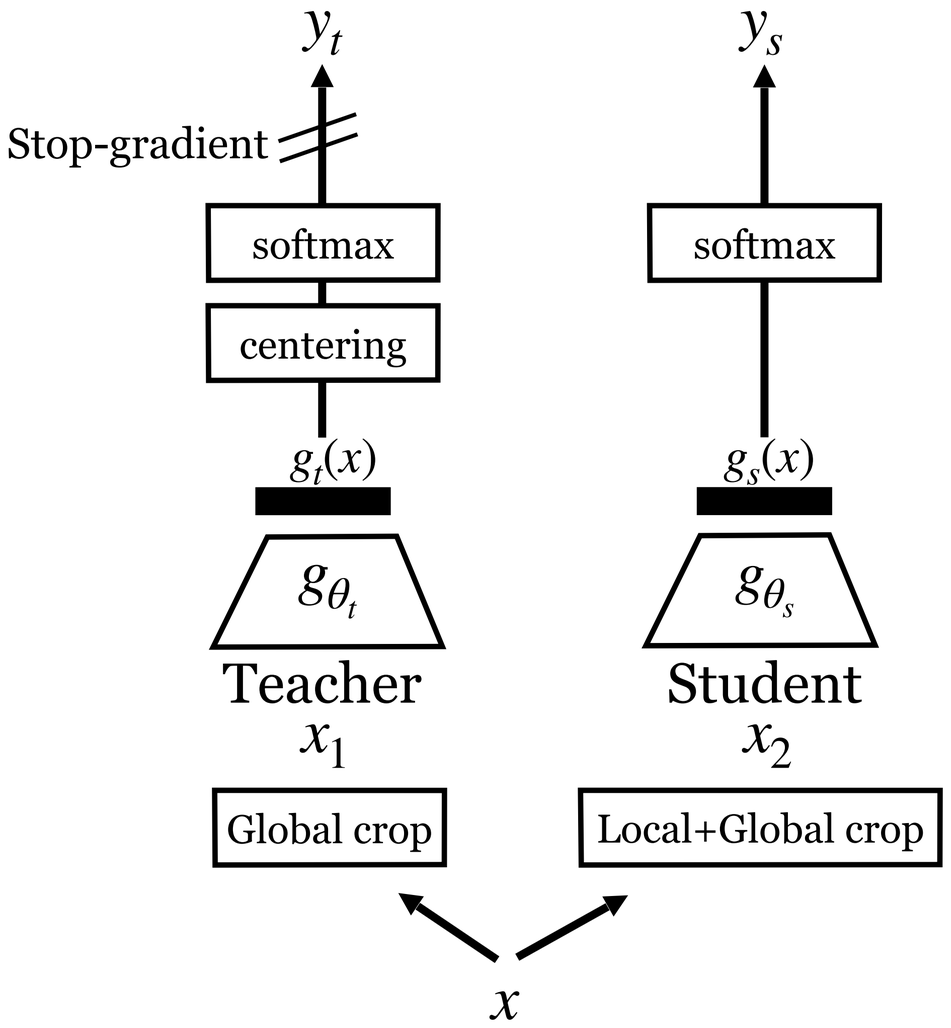
$$g_t(x) = g_t(x) - c$$

$$c = mc + (1-m)\frac{1}{B}\sum_{i=1}^{B} g_t(x)$$

- Sharpening (all uniform probability)
  - low temperature in softmax

$$\frac{exp(y_t^{(i)}/\tau)}{\sum_k exp(y_t^{(k)}/\tau)}$$

Knowledge **di**stillation *with* ***no*** *labels* (DINO)



Algorithm 1 DINO PyTorch pseudocode w/o multi-crop.

```
# gs, gt: student and teacher networks
# C: center (K)
# tps, tpt: student and teacher temperatures
# l, m: network and center momentum rates
gt.params = gs.params
for x in loader: # load a minibatch x with n samples
    x1, x2 = augment(x), augment(x) # random views

    s1, s2 = gs(x1), gs(x2) # student output n-by-K
    t1, t2 = gt(x1), gt(x2) # teacher output n-by-K

    loss = H(t1, s2)/2 + H(t2, s1)/2
    loss.backward() # back-propagate

    # student, teacher and center updates
    update(gs) # SGD
    gt.params = l*gt.params + (1-l)*gs.params
    C = m*C + (1-m)*cat([t1, t2]).mean(dim=0)

def H(t, s):
    t = t.detach() # stop gradient
    s = softmax(s / tps, dim=1)
    t = softmax((t - C) / tpt, dim=1) # center + sharpen
    return - (t * log(s)).sum(dim=1).mean()
```

Knowledge **di**stillation *with **no** labels* (DINO)

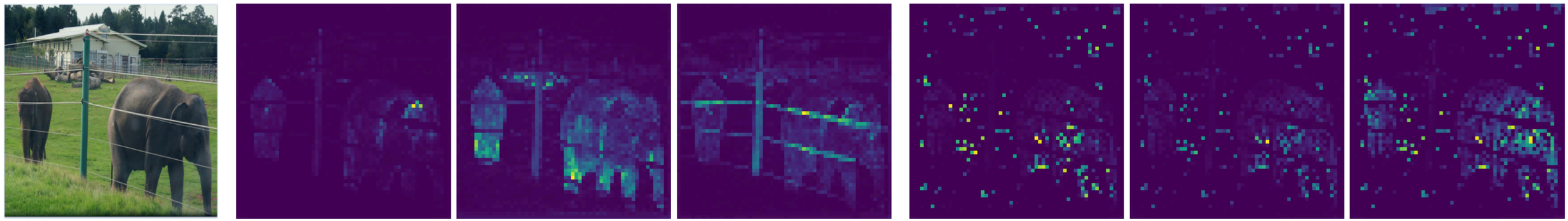
Why does it learn local information?

Knowledge **di**stillation *with **no** labels* (DINO)

Knowledge **di**stillation *with **no** labels* (DINO)

Knowledge **di**stillation *with **no** labels* (DINO)

Knowledge **di**stillation *with **no** labels* (DINO)



Three classes of self-supervision

- Generative
- Contrastive
- Distillation

What is self supervision?

- A general recipe:
  - Collect large quantities of unlabelled data
  - Define an auxiliary task
  - Train a model so solve this task
  - **Pray**

Advanced Computer Vision: Self-Supervised Learning

MLMI17

Ayush Tewari

UNIVERSITY OF CAMBRIDGE