Self-Supervision

Luiz Velho
IMPA

Based on Slides from Stanford-CS231n-2025

Outlook

- Features and Representation
- Learning Modalities
- Self-Supervised Learning Methods
  - Pretext Prediction
  - Masked Auto-Encoder (MAE)
  - Contrastive Learning (SimCLR)
  - Self-Distillation (DINO)

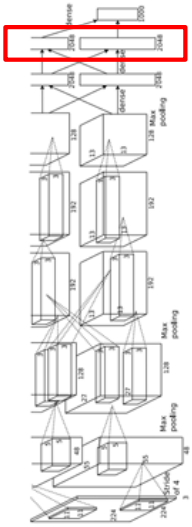
Features and Representations

Recall: Nearest neighbors in pixel space

Test image

L2 Nearest neighbors in feature space

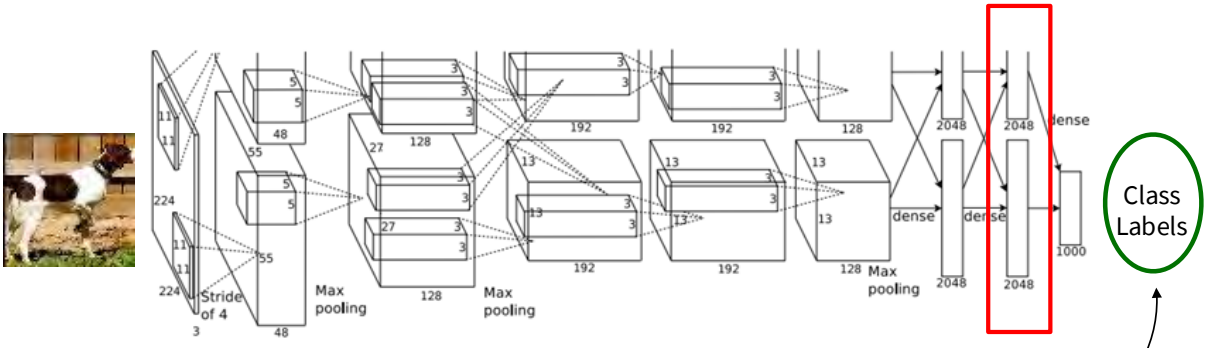
4096-dim vector



Krizhevsky et al, "ImageNet Classification with Deep Convolutional Neural Networks", NIPS 2012.
Figures reproduced with permission.

Supervised Learning

4096-dim vector



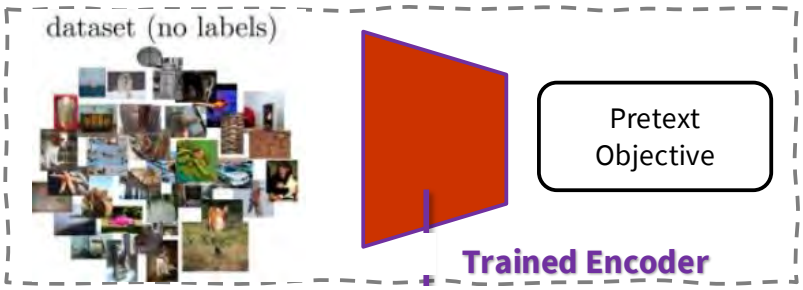
What is the problem with large-scale training?

- **We need a lot of labeled data**

Is there a way we can train neural networks without the need for huge manually labeled datasets?

Krizhevsky et al, "ImageNet Classification with Deep Convolutional Neural Networks", NIPS 2012.
Figures reproduced with permission.

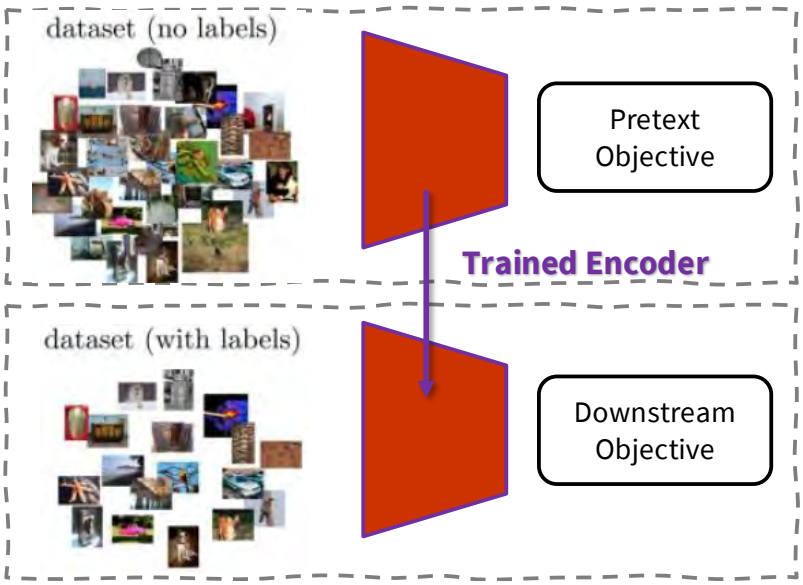
Unsupervised Learning



Pretext Task

- Define a task based on the data itself
- No manual annotation
- Could be considered an **unsupervised** task;

Unsupervised Transfer Learning



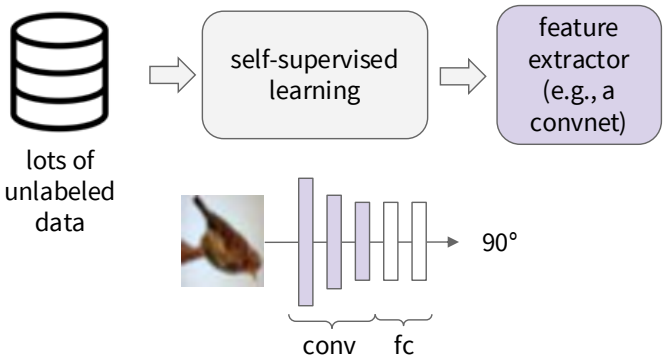
Pretext Task

- Define a task based on the data itself
- No manual annotation
- Could be considered an **unsupervised** task;

Downstream Task

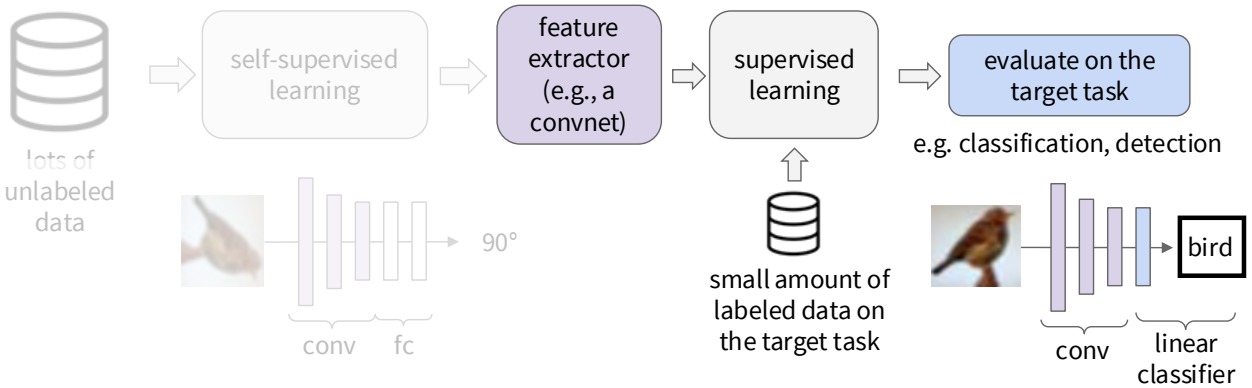
- The application you care about
- You do not have large datasets
- The dataset is labeled

Example: Extracting Features (1)



1. Learn good feature extractors from self-supervised pretext tasks, e.g.,

Example: Fine-Tuning with Labels (2)



1. Learn good feature extractors from self-supervised pretext tasks, e.g., predicting image rotations

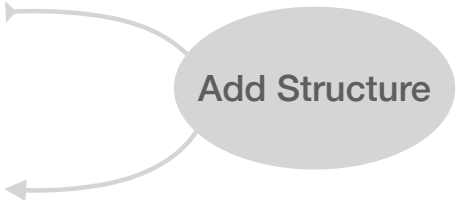
2. Attach a shallow network on the feature extractor; train the shallow network on the target task with small amount of labeled data

Self-Supervised Learning

Self-Supervised Paradigm

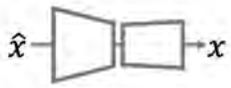
Create a pretext task based on data augmentation
whose solution forces the model to learn useful representations.

- Pretext Task Principles
  - Reconstruction
  - Invariance
- Data Augmentation
  - Transformations
  - Cropping / Masking



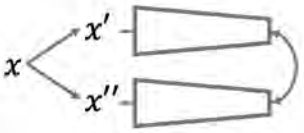
Network Architectures

- Reconstructive / Auto-Encoding



Reconstruction

- Contrastive / Siamese



Invariance

Learn the Essence of the Data

Self-Supervised Methods

- Auto-Encoding
  - Variation Prediction
    Learn by reconstructing input variations
  - Masked Prediction
    Hide part of the input and predict the whole
- Siamese
  - Contrastive
    Bring similar examples together and pushing different ones apart
  - Self-Distillation
    Network teaches itself to predict a representation of augmented view

Variation Prediction

Self-supervised pretext tasks

Example: learn to predict image transformations / complete corrupted images

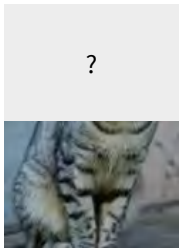
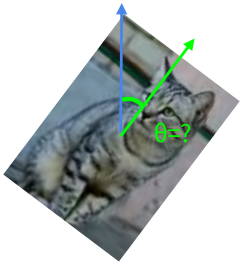


image completion



rotation prediction

"jigsaw puzzle"

colorization

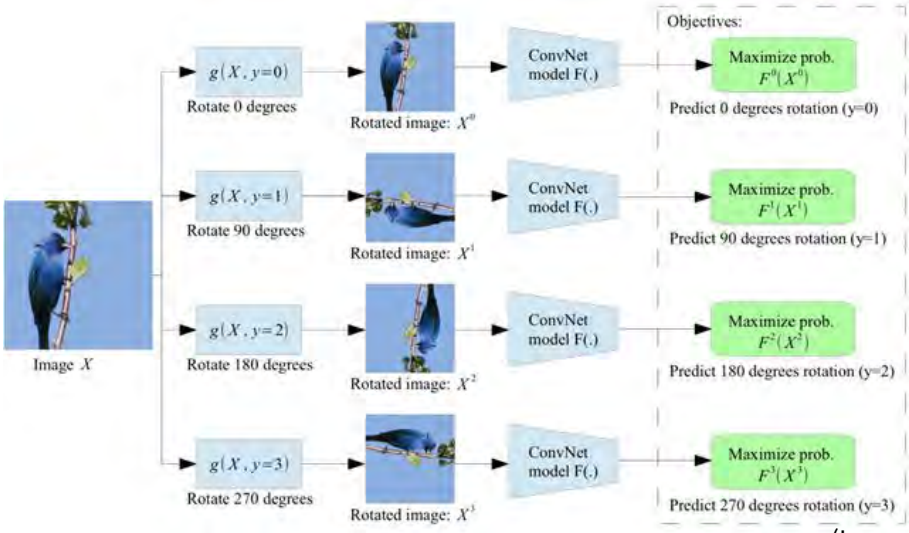
1. Solving the pretext tasks allow the model to learn good features.
2. We can automatically generate labels for the pretext tasks.

Pretext task: predict missing pixels (inpainting)

Context Encoders: Feature Learning by Inpainting (Pathak et al., 2016)

Source: Pathak et al., 2016

Pretext task: predict rotations

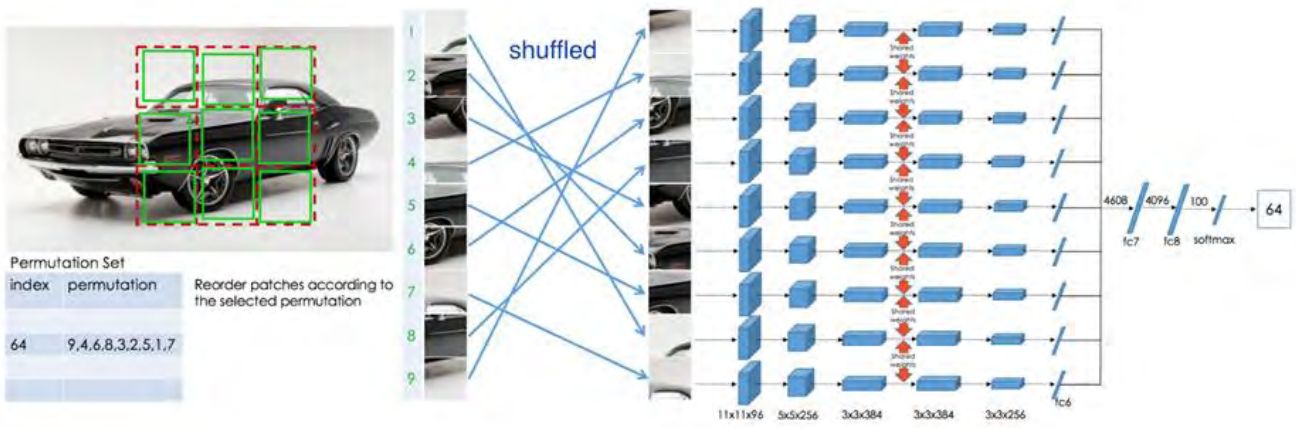


Self-supervised learning by rotating the entire input images.

The model learns to predict which rotation is applied (4-way classification)

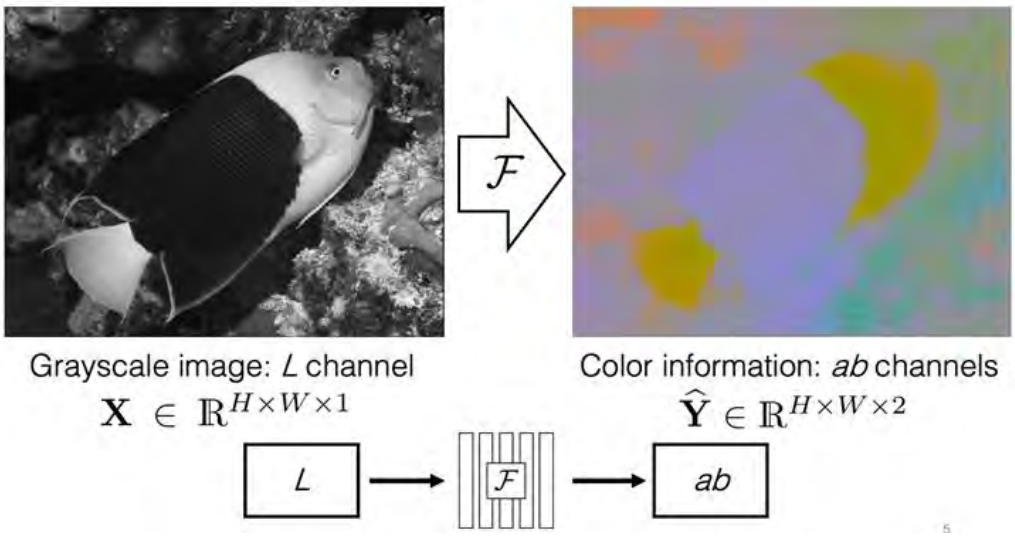
(Image source: Gidaris et al. 2018)

Pretext task: solving “jigsaw puzzles”



(Image source: Noroozi & Favaro, 2016)

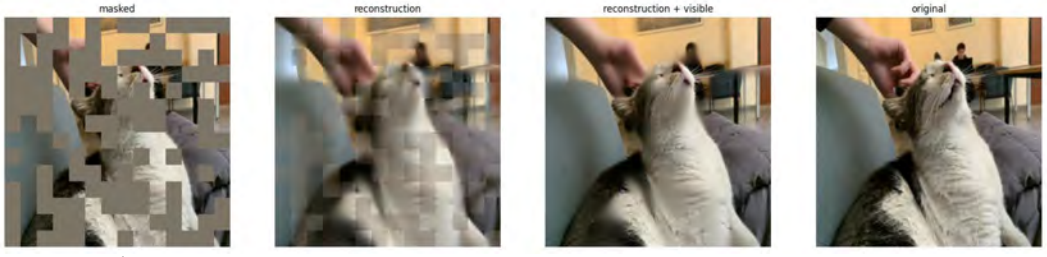
Pretext task: image coloring



Source: Richard Zhang / Phillip Isola

Masked Prediction

Masked Auto Encoders (MAE)
Reconstruction with a larger masked portion



50% masking ratio

75% masking ratio

He et al., 2021 Masked Autoencoders Are Scalable Vision Learners

Masking Methods

- Similar to the original ViT, divide the input into non-overlapping patches.
- Uniformly sample a very large proportion (75%) of these patches and mask them
- Masking a high ratio makes predicting the task challenging and meaningful.
- Also, not using mask tokens and picking a high sampling ratio (masking most of the image, e.g., 75%, and sampling a small visible portion, e.g., 25%, to feed into the encoder) enables the encoder to be very large.

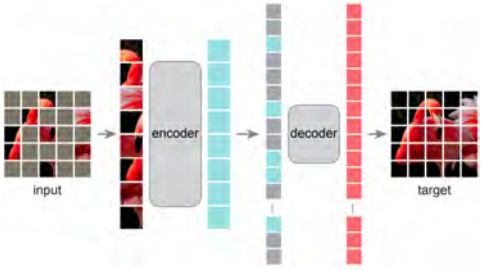
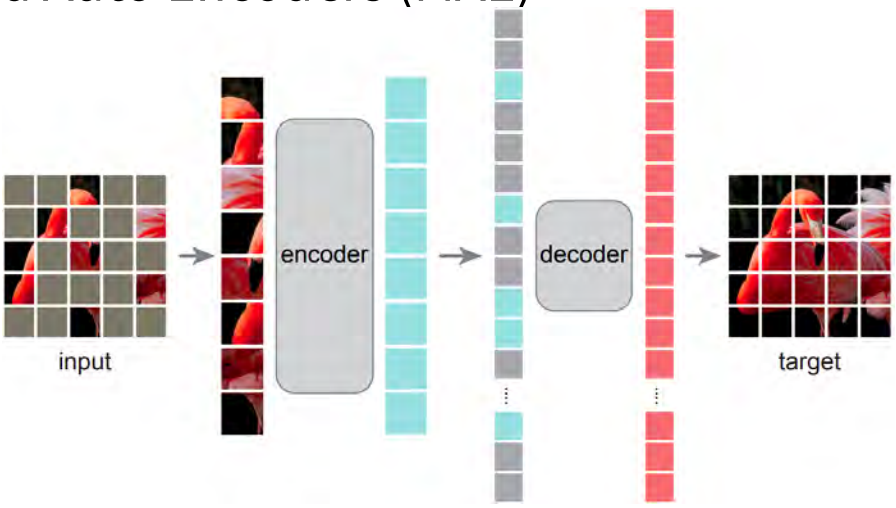


Figure 6. MAE architecture from the paper

He et al., 2021 Masked Autoencoders Are Scalable Vision Learners

Masked Auto Encoders (MAE)



He et al., 2021 Masked Autoencoders Are Scalable Vision Learners

MAE Encoder

- The encoder only operates on unmasked patches (25%)
- Embeds the patches by linear projection and add positional embeddings
- Uses transformer blocks
- Since the input patches are a small part of the input, the encoder is chosen to be very large. (encoder has over 9 times computations per token vs decoder)

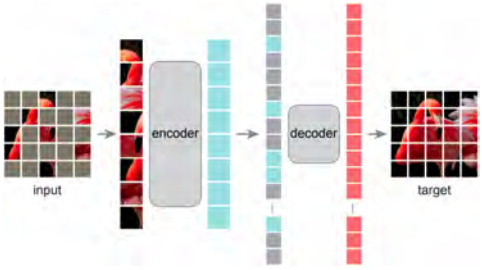


Figure 6. MAE architecture from the paper

He et al., 2021 Masked Autoencoders Are Scalable Vision Learners

MAE Decoder

- Merges the encoder outputs with the shared mask tokens in previously masked places, adding positional encodings to them.
- Uses transformer blocks, followed by a linear projection for finalizing pixel reconstruction.
- Is solely responsible for reconstruction, meaning it is not used post-training. Hence, it is independent of the encoder design, making it flexible (unlike traditional AEs or UNet). This is an asymmetrical autoencoder design.

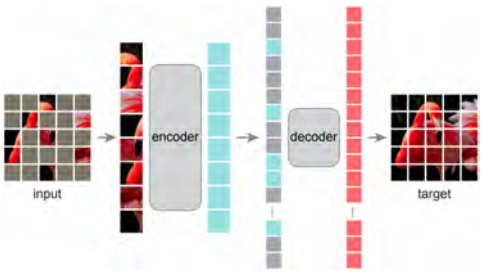


Figure 6. MAE architecture from the paper

He et al., 2021 Masked Autoencoders Are Scalable Vision Learners

Reconstruction

- The MSE (mean squared error loss) in the pixel space between the input image and the reconstructed image is adopted.
- Loss is only computed for masked patches

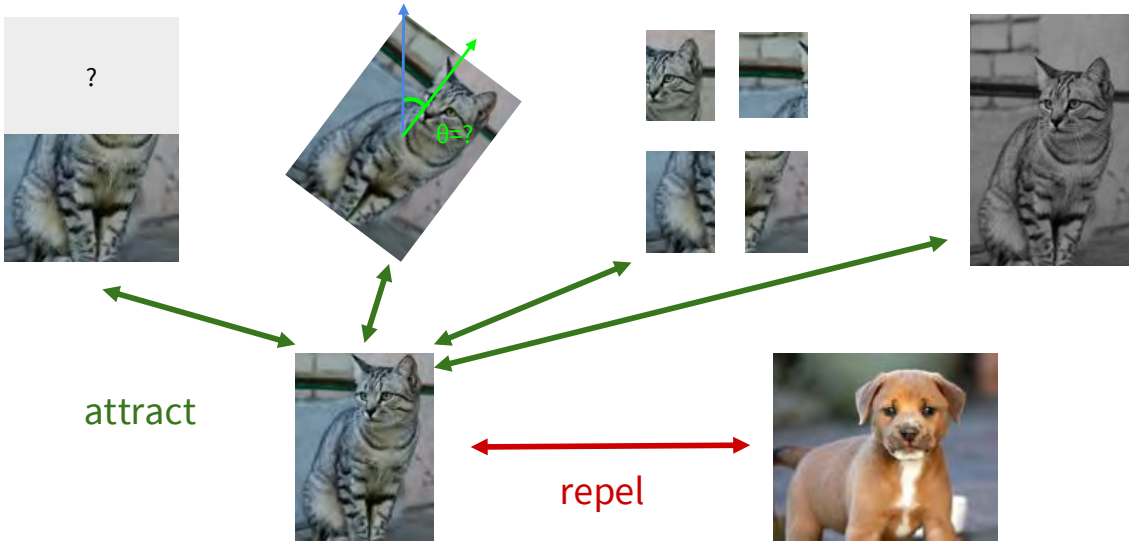
He et al., 2021 Masked Autoencoders Are Scalable Vision Learners

Summary: pretext tasks from image transformations

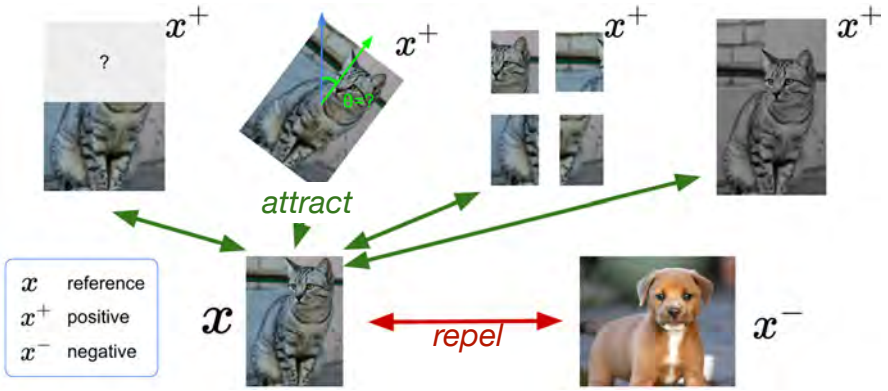
- Pretext tasks focus on "visual common sense", e.g., predict rotations, inpainting, rearrangement, and colorization.
- The models are forced learn good features about natural images, e.g., semantic representation of an object category, in order to solve the pretext tasks.
- We often do not care about the performance of these pretext tasks, but rather how useful the learned features are for downstream tasks (classification, detection, segmentation).
- Problems: 1) coming up with individual pretext tasks is tedious, and 2) the learned representations may not be general.

Contrastive

Contrastive Representation Learning



Contrastive Learning Score Function



- Given a chosen **score function** $s(\cdot, \cdot)$, we want to learn an encoder f that yields **high score for positive pairs** (x, x^+) and **low score for negative pairs** (x, x^-):

$$s(f(x), f(x^+)) \gg s(f(x), f(x^-))$$

Contrastive Learning

Assume we have1 reference (x), 1 positive (x^+) and N-1 negative (x_j^-) examples. Consider the following **multi-class cross entropy loss function:**

$$\mathcal{L} = -\mathbb{E}_X \left[\log \frac{\exp(s(f(x), f(x^+)))}{\exp(s(f(x), f(x^+))) + \sum_{j=1}^{N-1} \exp(s(f(x), f(x_j^-)))} \right]$$

This is commonly known as the **InfoNCE loss** and its negative is a **lower bound** on the **mutual information** between $f(x)$ and $f(x^+)$ (See [Oord, 2018] for details):

$$MI[f(x), f(x^+)] \geq \log(N) - \mathcal{L}$$

Key idea: maximizing mutual information between features extracted from multiple "views" forces the features to capture information about higher-level factors.
Important remark: The larger the negative sample size (N), the tighter the bound.

A formulation of contrastive learning

Loss function given 1 positive sample and N - 1 negative samples:

$$L = -\mathbb{E}_X \left[\log \frac{\exp(s(f(x), f(x^+))}{\exp(s(f(x), f(x^+)) + \sum_{j=1}^{N-1} \exp(s(f(x), f(x_j^-))} \right]$$

score for the positive pair

score for the N-1 negative pairs

Cross entropy loss for a N-way softmax classifier!
I.e., learn to find the positive sample from the N samples

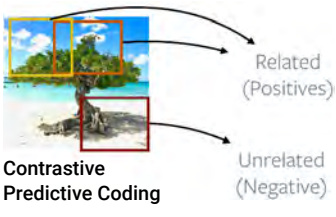
Design Choices

1. Score Function:

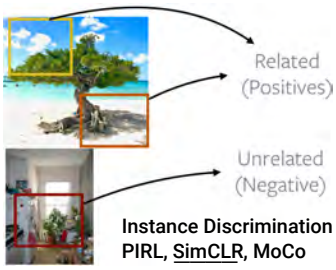
$$s(\mathbf{f}_1, \mathbf{f}_2) = \frac{\mathbf{f}_1^\top \mathbf{f}_2}{\|\mathbf{f}_1\| \, \|\mathbf{f}_2\|}$$

- Cosine similarity
- Commonly used

2. Examples:

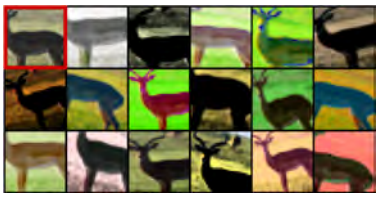


Contrastive Predictive Coding



Instance Discrimination
PIRL, SimCLR, MoCo

3. Augmentations:



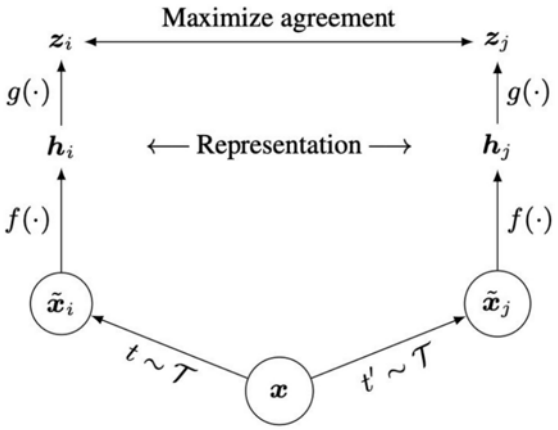
- Crop, resize, flip
- Rotation, cutout
- Color drop/jitter
- Gaussian noise/blur
- Sobel filter

A Simple Framework for Contrastive Learning

- **Cosine similarity** as score function:

$$s(\mathbf{z}_i, \mathbf{z}_j) = \frac{\mathbf{z}_i^\top \mathbf{z}_j}{\|\mathbf{z}_i\| \, \|\mathbf{z}_j\|}$$

- SimCLR uses a **projection network** $g(\cdot)$ to project features to a space where contrastive learning is applied
- The projection improves learning (more relevant information preserved in $\mathbf{h}$ which is discarded in $\mathbf{z}$)



A Simple Framework for Contrastive Learning

Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, constant τ, structure of $f, g, \mathcal{T}$.
for sampled minibatch $\{x_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{x}_{2k-1} = t(x_k)$
    $h_{2k-1} = f(\tilde{x}_{2k-1})$ # representation
    $z_{2k-1} = g(h_{2k-1})$ # projection
    # the second augmentation
    $\tilde{x}_{2k} = t'(x_k)$
    $h_{2k} = f(\tilde{x}_{2k})$ # representation
    $z_{2k} = g(h_{2k})$ # projection
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = z_i^\top z_j / (\|z_i\| \|z_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i, j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network $f(\cdot)$, and throw away $g(\cdot)$

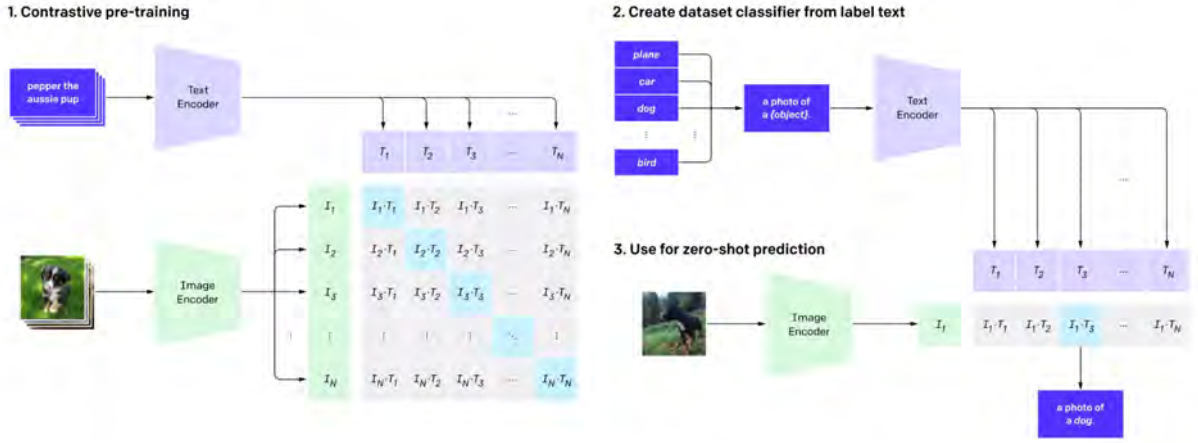
Generate a positive pair by sampling data augmentation functions

Iterate through and use each of the 2N sample as reference, compute average loss

InfoNCE loss: Use all non-positive samples in the batch as x^-

Other examples

Contrastive learning between image and natural language sentences



CLIP (*Contrastive Language–Image Pre-training*) Radford *et al.*, 2021

Self-Distillation

Self-Distillation Strategy

- What we want $f_\theta(I) = f_\theta(\text{augment}(I))$
- How we do it $f_\theta^{\text{student}}(I) = f_\theta^{\text{teacher}}(\text{augment}(I))$
- Prevent trivial solutions by asymmetry
  - Asymmetric **learning rule** between student teacher
  - Asymmetric **architecture** between student teacher

DINO: Self-Distillation with No Labels

Emerging Properties in Self-Supervised Vision Transformers

Mathilde Caron[1,2] Hugo Touvron[1,3] Ishan Misra[1] Hervé Jegou[1]
Julien Mairal[2] Piotr Bojanowski[1] Armand Joulin[1]

[1] Facebook AI Research [2] Inria* [3] Sorbonne University

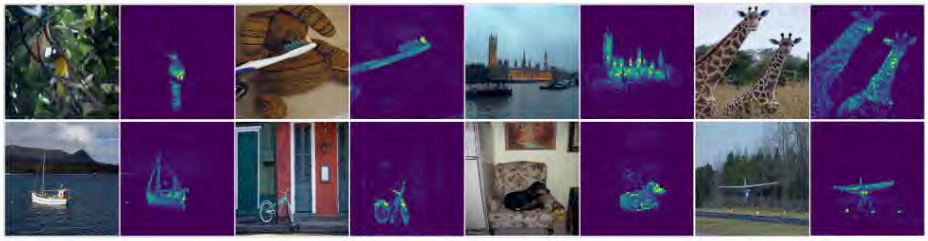
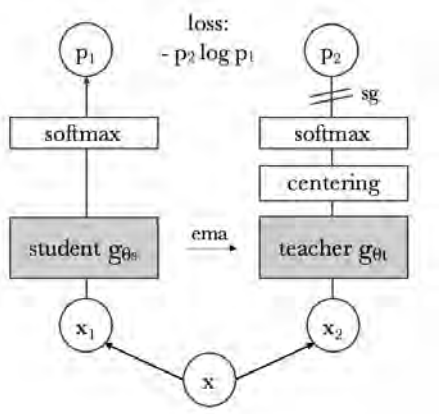


Figure 1: **Self-attention from a Vision Transformer with 8×8 patches trained with no supervision.** We look at the self-attention of the [CLS] token on the heads of the last layer. This token is not attached to any label nor supervision. These maps show that the model automatically learns class-specific features leading to unsupervised object segmentations.

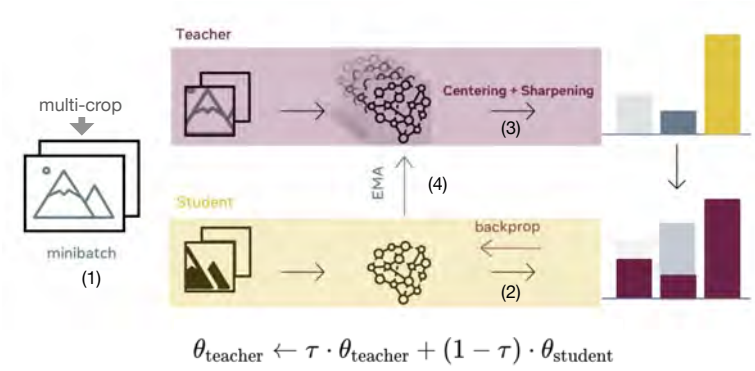
Caron et al. 2021 Emerging Properties in Self-Supervised Vision Transformers

DINO in a Nutshell

- Siamese Network Architecture

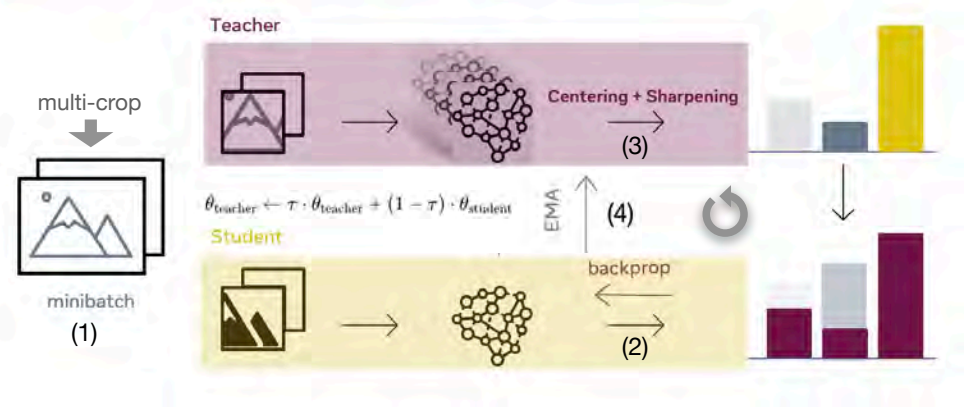


- "Teacher / Student" Training Strategy



DINO Training

1. Multi-Crop Data Augmentation
2. Train Student (BackProp)
3. Regularize (Center + Sharpen)
4. Update Teacher (EMA)



DINO Algorithm

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()
```

← Data Augmentation (*w/o multi-crop*)

← Student / Teacher Forward Pass

← Siamese Cross-Entropy

← Train Student

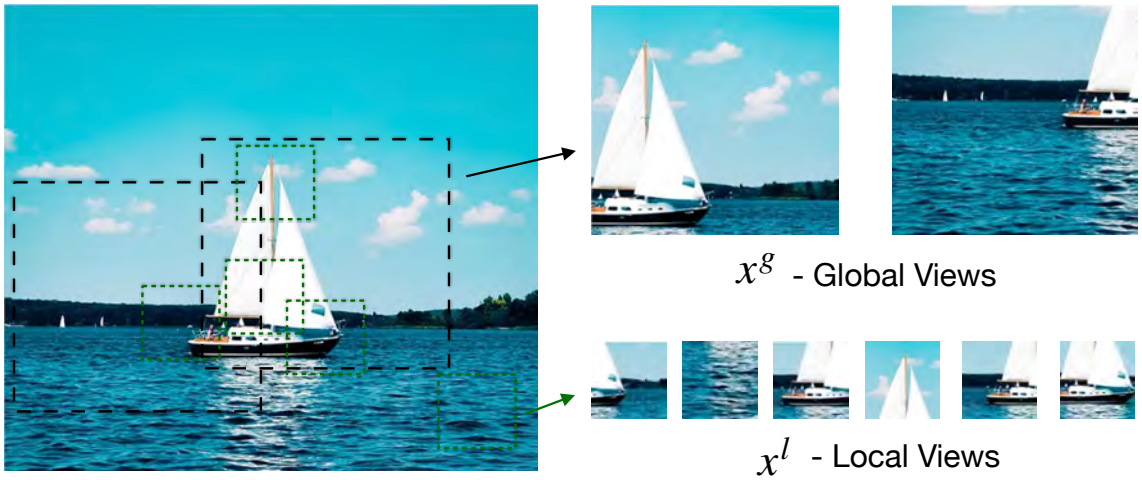
← Transfer to Teacher with EMA

← Center + Sharpen on Teacher

DINO Technical Details

- Dataset Augmentation
  - Local / Global Views
- Multi-Crop Strategy
  - Effective Representation Learning
- Centering / Sharpening
  - Collapse Avoidance Regularization

Dataset Augmentation



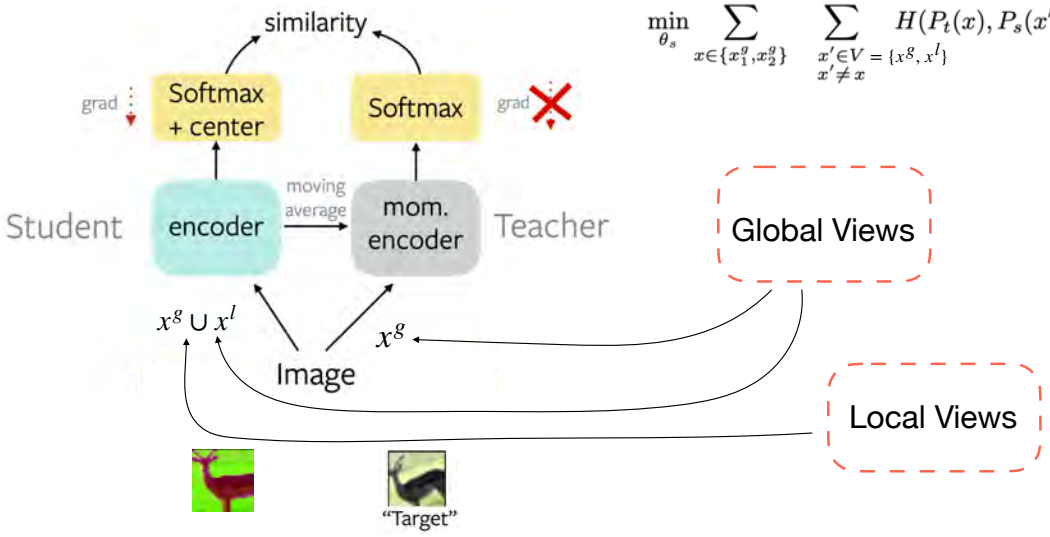
x^g - Global Views

x^l - Local Views

Multi-Crop Strategy

- Encourages Local to Global Correspondentes

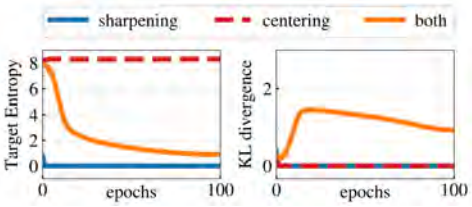
$$\min_{\theta_s} \sum_{x \in \{x_1^g, x_2^g\}} \sum_{\substack{x' \in V \\ x' \neq x}} H(P_t(x), P_s(x'))$$



Avoiding Collapse

How the model finds trivial ways to solve the SSL	… and how to prevent it.
Collapse all the representations to a constant output.	Centering + sharpening

- Centering -> (*move to average score*)
- Sharpening -> (*force a peak*)



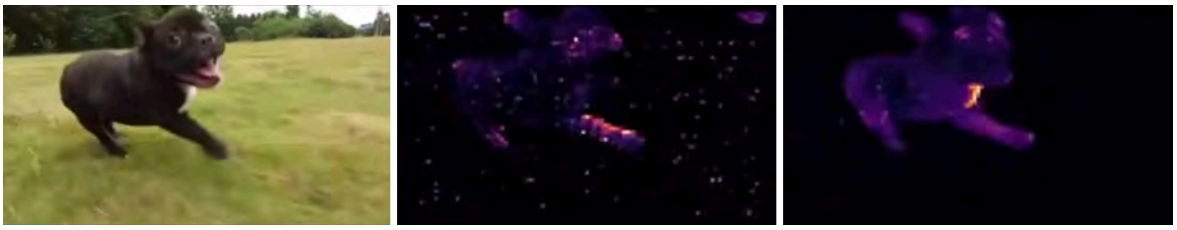
Centering + Sharpening

DINO Properties

- Nearest Neighbor Retrieval with DINO ViT
- Discovering the Semantic Layout of Scenes
- Transfer Learning on Downstream Tasks

ViT Attention Maps trained with DINO

- Supervised vs. Self-Supervised Comparison



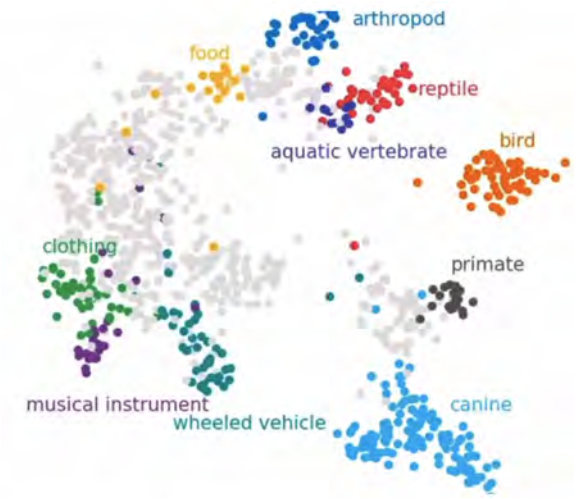
Input Baseline SSL (DINO)

Feature Representation for ImageNet

- Clusters

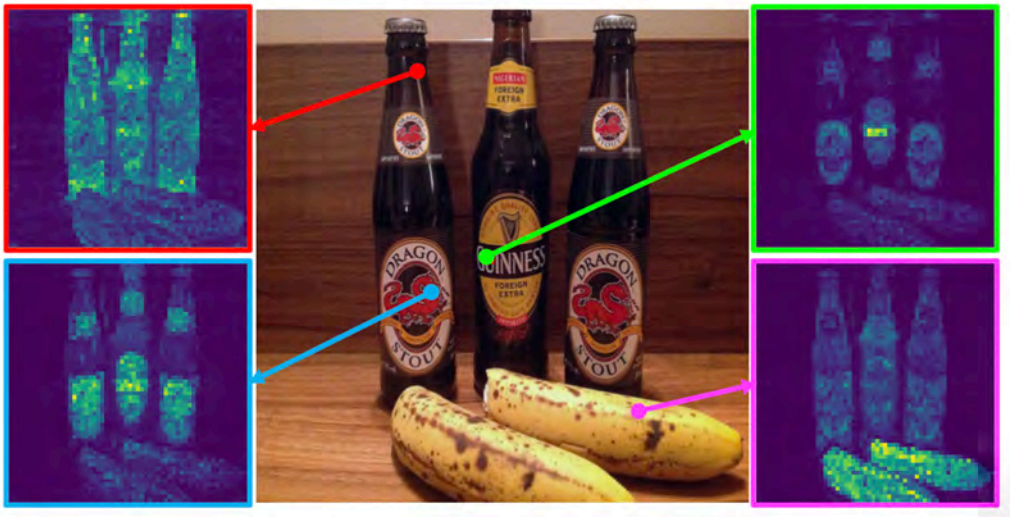
Feature Representation for ImageNet

- Classes



Object Discrimination

- Self-Attention for a Set of Reference Points



DINO v2

Figure 1: **Visualization of the first PCA components.** We compute a PCA between the patches of the images from the same column (a, b, c and d) and show their first 3 components. Each component is matched to a different color channel. Same parts are matched between related images despite changes of pose, style or even objects. Background is removed by thresholding the first PCA component.

Caron et al. 2021 Emerging Properties in Self-Supervised Vision Transformers