Self-supervised learning

M.C. Escher, *Drawing Hands* (1948) – via A. Efros
Motivation

- Recall the idea of *transfer learning*: start with general-purpose feature representation pre-trained on a large, diverse dataset and adapt it to specialized tasks
- Challenge: overcoming reliance on *supervised* pre-training
Unsupervised vs. self-supervised learning

- **Unsupervised learning**: any kind of learning without labels
  - Clustering and quantization
  - Dimensionality reduction, manifold learning
  - Density estimation
  - Learning to sample

- **Self-supervised learning**: a type of unsupervised learning in which the learner “makes up” labels from the data and then solves a supervised task
Self-supervised learning: Outline

Data prediction

Data $x$ → Network → Data $x'$

Transformation prediction

Data $T(x)$ → Network → $T$

“Siamese” methods

Data $T(x)$ → Network → Network → Similarity score

Source: A. Efros
Data prediction: Inpainting

Figure 1. **Our MAE architecture.** During pre-training, a large random subset of image patches (e.g., 75%) is masked out. The encoder is applied to the small subset of visible patches. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images to produce representations for recognition tasks.

K. He et al. [Masked autoencoders are scalable vision learners](https://arxiv.org/abs/2111.06938). CVPR 2022
Figure 2. Example results on ImageNet validation images. For each triplet, we show the masked image (left), our MAE reconstruction† (middle), and the ground-truth (right). The masking ratio is 80%, leaving only 39 out of 196 patches. More examples are in the appendix.

†As no loss is computed on visible patches, the model output on visible patches is qualitatively worse. One can simply overlay the output with the visible patches to improve visual quality. We intentionally opt not to do this, so we can more comprehensively demonstrate the method’s behavior.

K. He et al. Masked autoencoders are scalable vision learners. CVPR 2022
Colorization

R. Zhang, P. Isola, and A. Efros, Colorful Image Colorization, ECCV 2016
Colorization: Architecture

R. Zhang, P. Isola, and A. Efros, Colorful Image Colorization, ECCV 2016
Colorization: Results

R. Zhang, P. Isola, and A. Efros, Colorful Image Colorization, ECCV 2016
Failure Cases

Source: A. Efros, R. Zhang
Inherent Ambiguity

Grayscale

Source: A. Efros, R. Zhang
Inherent Ambiguity

Prediction

Ground Truth

Source: A. Efros, R. Zhang
Biases

Source: A. Efros, R. Zhang
Biases

Source: A. Efros, R. Zhang
Self-supervised learning: Outline

Data prediction

Data $x$ → Network → Data $x'$

Transformation prediction

Data $T(x)$ → Network → $T$
Context prediction

- *Pretext task*: randomly sample a patch and one of 8 neighbors
- Guess the spatial relationship between the patches

**Question 1:**
- **A:** Bottom right

**Question 2:**
- **A:** Top center

Context prediction: Semantics from a non-semantic task

Source: A. Efros
Context prediction: Details

Prevent “cheating”: sample patches with gaps, pre-process to overcome chromatic aberration

Context prediction: Results

- Combine learned feature extractor with R-CNN to perform detection on PASCAL VOC 2007
- Unsupervised pre-training is 5% mAP better than training from scratch, but still 8% below pre-training with ImageNet label supervision

Jigsaw puzzle solving

Claim: jigsaw solving is easier than context prediction, trains faster, transfers better

Jigsaw puzzle solving: Details

Predetermined set of 1000 permutations (out of 362,880 possible)

M. Noroozi and P. Favaro. [Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles](https://doi.org/10.1007/978-3-319-46434-8_5). ECCV 2016
Rotation prediction

- Pretext task: recognize image rotation (0, 90, 180, 270 degrees)

Rotation prediction

During training, feed in all four rotated versions of an image in the same mini-batch

S. Gidaris, P. Singh, and N. Komodakis. *Unsupervised representation learning by predicting image rotations.* ICLR 2018
Rotation prediction: PASCAL VOC Transfer results

<table>
<thead>
<tr>
<th>Method</th>
<th>Classification</th>
<th>Detection (mAP)</th>
<th>Segmentation (mIoU)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised (ImageNet)</td>
<td>79.9</td>
<td>56.8</td>
<td>48.0</td>
</tr>
<tr>
<td>Colorization</td>
<td>65.6</td>
<td>46.9</td>
<td>35.6</td>
</tr>
<tr>
<td>Context</td>
<td>65.3</td>
<td>51.1</td>
<td></td>
</tr>
<tr>
<td>Jigsaw</td>
<td>67.6</td>
<td>53.2</td>
<td>37.6</td>
</tr>
<tr>
<td>Rotation</td>
<td>73.0</td>
<td>54.4</td>
<td>39.1</td>
</tr>
</tbody>
</table>
Self-supervised learning: Outline

Data prediction

Data $x$ → Network → Data $x'$

Transformation prediction

Data $T(x)$ → Network → $T$

“Siamese” methods

Data $T(x)$ → Network → Network

Data $T'(x)$ → Network → Network

Source: A. Efros
“Siamese” methods

• Extract representations from two transformed versions of a data point, encourage these representations to be similar (or to have other desirable properties)
  - **Contrastive methods**: train using both positive (similar) and negative (dissimilar) pairs
  - **Non-contrastive methods**: train with only positive examples

Source: A. Efros
Contrastive methods

- Encourage representations of transformed versions of the same image to be the same and different images to be different
Contrastive loss formulation

- Given:
  - *Query point* $x$
  - *Positive sample* $x^+$: version of $x$ subjected to a random transformation or augmentation (cropping, rotation, color change, etc.)
  - *Negative samples* $x^-$
Contrastive loss formulation

- Given: query $x$, positive sample $x^+$, negative samples $x^-$
- Measure similarity by dot product of L2-normalized feature representations:
  $$\text{sim}(x, y) = \frac{f(x)}{\|f(x)\|_2} \cdot \frac{f(y)}{\|f(y)\|_2}$$

- **Contrastive loss**: make $x$ similar to $x^+$, dissimilar from $x^-$:
  $$l(x, x^+) = -\log \frac{\exp(\text{sim}(x, x^+)/\tau)}{\exp(\text{sim}(x, x^+)/\tau) + \sum_{j=1}^N \exp(\text{sim}(x, x_j^-)/\tau)}$$

- Intuitively, this is the loss of a softmax classifier that tries to classify $x$ as $x^+$
Recall: Softmax with temperature

Low temperature: More concentrated distribution

Higher temperature: More uniform distribution

Figure source
Mechanisms for obtaining negative samples

- **End-to-end**: dictionary of negative examples is the current minibatch
  - Pro: dictionary is encoded using latest encoder parameters
  - Con: needs large minibatches to work well, limited by GPU memory

Mechanisms for obtaining negative samples

- **Memory bank** of encodings of many points in the dataset – each point is encoded when last seen
  - Pros: no need to back-propagate through or re-encode memory bank entries, not limited by minibatch size
  - Cons: memory bank entries are inconsistently encoded with different parameters, many become stale

Mechanisms for obtaining negative samples

- **Momentum encoder**: queue in which each example is encoded using slowly changing parameters $w_k$ updated as $w_k \leftarrow m w_k + (1 - m)w_q$, where $m = 0.999$ is the momentum coefficient.

- Pros: no need to re-encode dictionary examples, queue removes the most stale examples, slowly changing momentum encoder parameters allow for good learning.

Mechanisms for obtaining negative samples

![Graph showing comparison of three contrastive loss mechanisms](image)

Figure 3. **Comparison of three contrastive loss mechanisms** under the ImageNet linear classification protocol. We adopt the same pretext task (Sec. 3.3) and only vary the contrastive loss mechanism (Figure 2). The number of negatives is $K$ in memory bank and MoCo, and is $K - 1$ in end-to-end (offset by one because the positive key is in the same mini-batch). The network is ResNet-50.

MoCo results

Comparison on linear ImageNet classification (supervised accuracy above 75%)

K. He et al. Momentum Contrast for Unsupervised Visual Representation Learning. CVPR 2020
MoCo results

Comparison on PASCAL object detection

<table>
<thead>
<tr>
<th>pre-train</th>
<th>AP$_{50}$</th>
<th>AP</th>
<th>AP$_{75}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>random init.</td>
<td>60.2</td>
<td>33.8</td>
<td>33.1</td>
</tr>
<tr>
<td>super. IN-1M</td>
<td>81.3 (+0.2)</td>
<td>53.5</td>
<td>58.8</td>
</tr>
<tr>
<td>MoCo IN-1M</td>
<td>81.5 (+0.2)</td>
<td>55.9 (+2.4)</td>
<td>62.6 (+3.8)</td>
</tr>
<tr>
<td>MoCo IG-1B</td>
<td>82.2 (+0.9)</td>
<td>57.2 (+3.7)</td>
<td>63.7 (+4.9)</td>
</tr>
</tbody>
</table>

(b) Faster R-CNN, R50-C4

Table 2. **Object detection fine-tuned on PASCAL VOC trainval07+12.** Evaluation is on test2007: AP$_{50}$ (default VOC metric), AP (COCO-style), and AP$_{75}$, averaged over 5 trials. All are fine-tuned for 24k iterations (~23 epochs). In the brackets are the gaps to the ImageNet supervised pre-training counterpart. In green are the gaps of at least +0.5 point.

SimCLR

- Instead of memory bank or queue, use large mini-batch size (on cloud TPU)
- Introduce nonlinear projection ($g$) between representation ($h$) and feature used for computing contrastive loss ($z$)

SimCLR

- Performed extensive ablation study of data augmentations
- Found that composing multiple augmentations gives the best results

Figure 4. Illustrations of the studied data augmentation operators. Each augmentation can transform data stochastically with some internal parameters (e.g. rotation degree, noise level). Note that we only test these operators in ablation, the augmentation policy used to train our models only includes random crop (with flip and resize), color distortion, and Gaussian blur. (Original image cc-by: Von-grzanka)

Figure 5. Linear evaluation (ImageNet top-1 accuracy) under individual or composition of data augmentations, applied only to one branch. For all columns but the last, diagonal entries correspond to single transformation, and off-diagonals correspond to composition of two transformations (applied sequentially). The last column reflects the average over the row.

SimCLR: Evaluation

No detection evaluation

Non-contrastive methods

• Extract representations from two transformed versions of a data point, encourage these representations to be similar (or to have other desirable properties)
  • **Contrastive methods**: train using both positive (similar) and negative (dissimilar) pairs
    • Key challenge: sampling of negative pairs
  • **Non-contrastive methods**: train with only positive examples
    • Key challenge: avoiding degenerate solutions (all representations collapsing to constant output value)
BYOL

- Use momentum encoder, but without the queue of negative examples
- Use projection head like SimCLR, add prediction head to online network

Figure 2: BYOL’s architecture. BYOL minimizes a similarity loss between $q_\theta(z_\theta)$ and $sg(z'_\xi)$, where $\theta$ are the trained weights, $\xi$ are an exponential moving average of $\theta$ and $sg$ means stop-gradient. At the end of training, everything but $f_\theta$ is discarded, and $y_\theta$ is used as the image representation.

BYOL: Evaluation

Figure 1: Performance of BYOL on ImageNet (linear evaluation) using ResNet-50 and our best architecture ResNet-200 (2×), compared to other unsupervised and supervised (Sup.) baselines [8].
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 network 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.

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

```python
# 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(x1), augment(x2) # 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()
```

Figure 1: **Self-attention from a Vision Transformer with $8 \times 8$ 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.
SimSiam

Like BYOL, but without the momentum encoder
Like SimCLR, but without the negative pairs

Figure 1. **SimSiam architecture.** Two augmented views of one image are processed by the same encoder network $f$ (a backbone plus a projection MLP). Then a prediction MLP $h$ is applied on one side, and a stop-gradient operation is applied on the other side. The model maximizes the similarity between both sides. It uses neither negative pairs nor a momentum encoder.

## SimSiam: Evaluation

<table>
<thead>
<tr>
<th>method</th>
<th>batch size</th>
<th>negative pairs</th>
<th>momentum encoder</th>
<th>100 ep</th>
<th>200 ep</th>
<th>400 ep</th>
<th>800 ep</th>
</tr>
</thead>
<tbody>
<tr>
<td>SimCLR (repro.+)</td>
<td>4096</td>
<td>✓</td>
<td></td>
<td>66.5</td>
<td>68.3</td>
<td>69.8</td>
<td>70.4</td>
</tr>
<tr>
<td>MoCo v2 (repro.+)</td>
<td><strong>256</strong></td>
<td>✓</td>
<td>✓</td>
<td>67.4</td>
<td>69.9</td>
<td>71.0</td>
<td>72.2</td>
</tr>
<tr>
<td>BYOL (repro.)</td>
<td>4096</td>
<td></td>
<td>✓</td>
<td>66.5</td>
<td><strong>70.6</strong></td>
<td>73.2</td>
<td><strong>74.3</strong></td>
</tr>
<tr>
<td>SwAV (repro.+)</td>
<td>4096</td>
<td></td>
<td></td>
<td>66.5</td>
<td>69.1</td>
<td>70.7</td>
<td>71.8</td>
</tr>
<tr>
<td>SimSiam</td>
<td><strong>256</strong></td>
<td></td>
<td></td>
<td><strong>68.1</strong></td>
<td>70.0</td>
<td>70.8</td>
<td>71.3</td>
</tr>
</tbody>
</table>

| Table 4. **Comparisons on ImageNet linear classification.** All are based on ResNet-50 pre-trained with two $224 \times 224$ views. Evaluation is on a single crop. All competitors are from our reproduction, and “+” denotes improved reproduction vs. original papers (see supplement).
SimSiam: Evaluation

<table>
<thead>
<tr>
<th>pre-train</th>
<th>VOC 07 detection AP50</th>
<th>AP</th>
<th>AP75</th>
<th>VOC 07+12 detection AP50</th>
<th>AP</th>
<th>AP75</th>
<th>COCO detection AP50</th>
<th>AP</th>
<th>AP75</th>
<th>COCO instance seg. APmask50</th>
<th>APmask</th>
<th>APmask</th>
</tr>
</thead>
<tbody>
<tr>
<td>scratch</td>
<td>35.9</td>
<td>16.8</td>
<td>13.0</td>
<td>60.2</td>
<td>33.8</td>
<td>33.1</td>
<td>44.0</td>
<td>26.4</td>
<td>27.8</td>
<td>46.9</td>
<td>29.3</td>
<td>30.8</td>
</tr>
<tr>
<td>ImageNet supervised</td>
<td>74.4</td>
<td>42.4</td>
<td>42.7</td>
<td>81.3</td>
<td>53.5</td>
<td>58.8</td>
<td>58.2</td>
<td>38.2</td>
<td>41.2</td>
<td>54.7</td>
<td>33.3</td>
<td>35.2</td>
</tr>
<tr>
<td>SimCLR (repro.+</td>
<td>75.9</td>
<td>46.8</td>
<td>50.1</td>
<td>81.8</td>
<td>55.5</td>
<td>61.4</td>
<td>57.7</td>
<td>37.9</td>
<td>40.9</td>
<td>54.6</td>
<td>33.3</td>
<td>35.3</td>
</tr>
<tr>
<td>MoCo v2 (repro.+</td>
<td>77.1</td>
<td>48.5</td>
<td>52.5</td>
<td>82.3</td>
<td>57.0</td>
<td>63.3</td>
<td>58.8</td>
<td>39.2</td>
<td>42.5</td>
<td>55.5</td>
<td>34.3</td>
<td>36.6</td>
</tr>
<tr>
<td>BYOL (repro.)</td>
<td>77.1</td>
<td>47.0</td>
<td>49.9</td>
<td>81.4</td>
<td>55.3</td>
<td>61.1</td>
<td>57.8</td>
<td>37.9</td>
<td>40.9</td>
<td>54.3</td>
<td>33.2</td>
<td>35.0</td>
</tr>
<tr>
<td>SwAV (repro.+</td>
<td>75.5</td>
<td>46.5</td>
<td>49.6</td>
<td>81.5</td>
<td>55.4</td>
<td>61.4</td>
<td>57.6</td>
<td>37.6</td>
<td>40.3</td>
<td>54.2</td>
<td>33.1</td>
<td>35.1</td>
</tr>
<tr>
<td>SimSiam, base</td>
<td>75.5</td>
<td>47.0</td>
<td>50.2</td>
<td>82.0</td>
<td>56.4</td>
<td>62.8</td>
<td>57.5</td>
<td>37.9</td>
<td>40.9</td>
<td>54.2</td>
<td>33.2</td>
<td>35.2</td>
</tr>
<tr>
<td>SimSiam, optimal</td>
<td>77.3</td>
<td>48.5</td>
<td>52.5</td>
<td>82.4</td>
<td>57.0</td>
<td>63.7</td>
<td>59.3</td>
<td>39.2</td>
<td>42.1</td>
<td>56.0</td>
<td>34.4</td>
<td>36.7</td>
</tr>
</tbody>
</table>

Table 5. Transfer Learning. All unsupervised methods are based on 200-epoch pre-training in ImageNet. *VOC 07 detection*: Faster R-CNN [30] fine-tuned in VOC 2007 trainval, evaluated in VOC 2007 test; *VOC 07+12 detection*: Faster R-CNN fine-tuned in VOC 2007 trainval + 2012 train, evaluated in VOC 2007 test; *COCO detection* and *COCO instance segmentation*: Mask R-CNN [18] (1× schedule) fine-tuned in COCO 2017 train, evaluated in COCO 2017 val. All Faster/Mask R-CNN models are with the C4-backbone [13]. All VOC results are the average over 5 trials. **Bold entries** are within 0.5 below the best.
Barlow Twins

- Get rid of asymmetric encoders, stop-gradients
- To avoid degeneracy, force cross-correlation matrix of “twin” embeddings to be close to identity – this tries to make individual dimensions of embeddings non-redundant
- Use projection head that expands dimensions

J. Zbontar et al. Barlow Twins: Self-Supervised Learning via Redundancy Reduction. ICML 2021
Barlow Twins

\[ \mathcal{L}_{BT} \triangleq \sum_i (1 - C_{ii})^2 + \lambda \sum_i \sum_{j \neq i} C_{ij}^2 \]

\[ C_{ij} \triangleq \frac{\sum_b z_{b,i}^A z_{b,j}^B}{\sqrt{\sum_b (z_{b,i}^A)^2} \sqrt{\sum_b (z_{b,j}^B)^2}} \]

Figure 1. BARLOW TWINS’s objective function measures the cross-correlation matrix between the embeddings of two identical networks fed with distorted versions of a batch of samples, and tries to make this matrix close to the identity. This causes the embedding vectors of distorted versions of a sample to be similar, while minimizing the redundancy between the components of these vectors. BARLOW TWINS is competitive with state-of-the-art methods for self-supervised learning while being conceptually simpler, naturally avoiding trivial constant (i.e. collapsed) embeddings, and being robust to the training batch size.

J. Zbontar et al. Barlow Twins: Self-Supervised Learning via Redundancy Reduction. ICML 2021
Barlow Twins: Evaluation

<table>
<thead>
<tr>
<th>Method</th>
<th>Top-1</th>
<th>Top-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised</td>
<td>76.5</td>
<td></td>
</tr>
<tr>
<td>MoCo</td>
<td>60.6</td>
<td></td>
</tr>
<tr>
<td>PIRL</td>
<td>63.6</td>
<td>-</td>
</tr>
<tr>
<td>SIMCLR</td>
<td>69.3</td>
<td>89.0</td>
</tr>
<tr>
<td>MoCo v2</td>
<td>71.1</td>
<td>90.1</td>
</tr>
<tr>
<td>SIMSIAM</td>
<td>71.3</td>
<td></td>
</tr>
<tr>
<td>SWAV (w/o multi-crop)</td>
<td>71.8</td>
<td>-</td>
</tr>
<tr>
<td>BYOL</td>
<td>74.3</td>
<td>91.6</td>
</tr>
<tr>
<td>SWAV</td>
<td>75.3</td>
<td></td>
</tr>
<tr>
<td>Barlow Twins (ours)</td>
<td>73.2</td>
<td>91.0</td>
</tr>
</tbody>
</table>

*Table 1. Top-1 and top-5 accuracies (in %) under linear evaluation on ImageNet. All models use a ResNet-50 encoder. Top-3 best self-supervised methods are underlined.*

Barlow Twins: Evaluation

Table 4. **Transfer learning: object detection and instance segmentation.** We benchmark learned representations on the object detection task on VOC07+12 using Faster R-CNN (Ren et al., 2015) and on the detection and instance segmentation task on COCO using Mask R-CNN (He et al., 2017). All methods use the C4 backbone variant (Wu et al., 2019) and models on COCO are finetuned using the $1 \times$ schedule. Best results are in **bold**.

<table>
<thead>
<tr>
<th>Method</th>
<th>VOC07+12 det</th>
<th>COCO det</th>
<th>COCO instance seg</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AP&lt;sub&gt;all&lt;/sub&gt;</td>
<td>AP&lt;sub&gt;50&lt;/sub&gt;</td>
<td>AP&lt;sub&gt;75&lt;/sub&gt;</td>
</tr>
<tr>
<td>Sup.</td>
<td>53.5</td>
<td>81.3</td>
<td>58.8</td>
</tr>
<tr>
<td>MoCo-v2</td>
<td>57.4</td>
<td>82.5</td>
<td><strong>64.0</strong></td>
</tr>
<tr>
<td>SwAV</td>
<td>56.1</td>
<td><strong>82.6</strong></td>
<td>62.7</td>
</tr>
<tr>
<td>SimSiam</td>
<td>57</td>
<td>82.4</td>
<td>63.7</td>
</tr>
<tr>
<td>BT (ours)</td>
<td>56.8</td>
<td><strong>82.6</strong></td>
<td>63.4</td>
</tr>
</tbody>
</table>

VICReg

- **Invariance**: the mean square distance between the embedding vectors.

- **Variance**: a hinge loss to maintain the standard deviation (over a batch) of each variable of the embedding above a given threshold. This term forces the embedding vectors of samples within a batch to be different.

- **Covariance**: a term that attracts the covariances (over a batch) between every pair of (centered) embedding variables towards zero. This term decorrelates the variables of each embedding and prevents an *informational collapse* in which the variables would vary together or be highly correlated.

A. Bardes et al. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. ICLR 2022
VICReg

- Advantages over Barlow Twins:
  - More general and applicable to cross-modal data: “does not require that the weights of the two branches be shared, not that the architectures be identical, nor that the inputs be of the same nature”
  - Including the variance preservation term removes the need to use normalized correlation matrix

A. Bardes et al. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. ICLR 2022
## VICReg: Evaluation

Table 1: **Evaluation on ImageNet.** Evaluation of the representations obtained with a ResNet-50 backbone pretrained with VICReg on: (1) linear classification on top of the frozen representations from ImageNet; (2) semi-supervised classification on top of the fine-tuned representations from 1% and 10% of ImageNet samples. We report Top-1 and Top-5 accuracies (in %). Top-3 best self-supervised methods are underlined.

<table>
<thead>
<tr>
<th>Method</th>
<th>Linear</th>
<th></th>
<th>Semi-supervised</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Top-1</td>
<td>Top-5</td>
<td>Top-1 1% 10%</td>
<td>Top-5 1% 10%</td>
</tr>
<tr>
<td>Supervised</td>
<td>76.5</td>
<td>-</td>
<td>25.4 56.4 48.4 80.4</td>
<td></td>
</tr>
<tr>
<td>MoCo He et al. (2020)</td>
<td>60.6</td>
<td>-</td>
<td>-     - -</td>
<td>-</td>
</tr>
<tr>
<td>PIRL Misra &amp; Maaten (2020)</td>
<td>63.6</td>
<td>-</td>
<td>-     - - - 57.2 83.8</td>
<td></td>
</tr>
<tr>
<td>CPC v2 Hénaff et al. (2019)</td>
<td>63.8</td>
<td>-</td>
<td>-     - - -</td>
<td>-</td>
</tr>
<tr>
<td>CMC Tian et al. (2019)</td>
<td>66.2</td>
<td>-</td>
<td>-     - -</td>
<td>-</td>
</tr>
<tr>
<td>SimCLR Chen et al. (2020a)</td>
<td>69.3 89.0</td>
<td></td>
<td>48.3 65.6 75.5 87.8</td>
<td></td>
</tr>
<tr>
<td>MoCo v2 Chen et al. (2020c)</td>
<td>71.1</td>
<td>-</td>
<td>-     - -</td>
<td>-</td>
</tr>
<tr>
<td>SimSiam Chen &amp; He (2020)</td>
<td>71.3</td>
<td>-</td>
<td>-     - -</td>
<td>-</td>
</tr>
<tr>
<td>SwAV Caron et al. (2020)</td>
<td>71.8</td>
<td>-</td>
<td>-     - -</td>
<td>-</td>
</tr>
<tr>
<td>InfoMin Aug Tian et al. (2020)</td>
<td>73.0 91.1</td>
<td></td>
<td>-     - -</td>
<td>-</td>
</tr>
<tr>
<td>OBoW Gidaris et al. (2021)</td>
<td>73.8</td>
<td>-</td>
<td>-     - - 82.9 90.7</td>
<td></td>
</tr>
<tr>
<td>BYOL Grill et al. (2020)</td>
<td>74.3 91.6</td>
<td></td>
<td>53.2 68.8 78.4 89.0</td>
<td></td>
</tr>
<tr>
<td>SwAV (w/ multi-crop) Caron et al. (2020)</td>
<td>75.3</td>
<td>-</td>
<td>53.9 70.2 78.5 89.9</td>
<td></td>
</tr>
<tr>
<td>Barlow Twins Zbontar et al. (2021)</td>
<td>73.2 91.0</td>
<td></td>
<td>55.0 69.7 79.2 89.3</td>
<td></td>
</tr>
<tr>
<td>VICReg (ours)</td>
<td>73.2 91.1</td>
<td></td>
<td>54.8 69.5 79.4 89.5</td>
<td></td>
</tr>
</tbody>
</table>

VICReg: Evaluation

Table 2: **Transfer learning on downstream tasks.** Evaluation of the representations from a ResNet-50 backbone pretrained with VICReg on: (1) linear classification tasks on top of frozen representations, we report Top-1 accuracy (in %) for Places205 Zhou et al. (2014) and iNat18 Horn et al. (2018), and mAP for VOC07 Everingham et al. (2010); (2) object detection with fine-tuning, we report AP$_{50}$ for VOC07+12 using Faster R-CNN with C4 backbone Ren et al. (2015); (3) object detection and instance segmentation, we report AP for COCO Lin et al. (2014) using Mask R-CNN with FPN backbone He et al. (2017). We use † to denote the experiments run by us. Top-3 best self-supervised methods are underlined.

<table>
<thead>
<tr>
<th>Method</th>
<th>Linear Classification</th>
<th></th>
<th>Object Detection</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Places205</td>
<td>VOC07</td>
<td>iNat18</td>
<td>VOC07+12</td>
</tr>
<tr>
<td>Supervised</td>
<td>53.2</td>
<td>87.5</td>
<td>46.7</td>
<td>81.3</td>
</tr>
<tr>
<td>MoCo He et al. (2020)</td>
<td>46.9</td>
<td>79.8</td>
<td>31.5</td>
<td>-</td>
</tr>
<tr>
<td>PIRL Misra &amp; Maaten (2020)</td>
<td>49.8</td>
<td>81.1</td>
<td>34.1</td>
<td>-</td>
</tr>
<tr>
<td>SimCLR Chen et al. (2020a)</td>
<td>52.5</td>
<td>85.5</td>
<td>37.2</td>
<td>-</td>
</tr>
<tr>
<td>MoCo v2 Chen et al. (2020c)</td>
<td>51.8</td>
<td>86.4</td>
<td>38.6</td>
<td>82.5</td>
</tr>
<tr>
<td>SimSiam Chen &amp; He (2020)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>82.4</td>
</tr>
<tr>
<td>BYOL Grill et al. (2020)</td>
<td>54.0</td>
<td>86.6</td>
<td>47.6</td>
<td>-</td>
</tr>
<tr>
<td>SwAV (m-c) Caron et al. (2020)</td>
<td>56.7</td>
<td>88.9</td>
<td>48.6</td>
<td>82.6</td>
</tr>
<tr>
<td>OBoW Gidaris et al. (2021)</td>
<td>56.8</td>
<td>89.3</td>
<td>-</td>
<td>82.9</td>
</tr>
<tr>
<td>Barlow Twins Grill et al. (2020)</td>
<td>54.1</td>
<td>86.2</td>
<td>46.5</td>
<td>82.6</td>
</tr>
<tr>
<td>VICReg (ours)</td>
<td>54.3</td>
<td>86.6</td>
<td>47.0</td>
<td>82.4</td>
</tr>
</tbody>
</table>

A. Bardes et al. **VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning.** ICLR 2022
Self-supervised learning: Outline

- Data prediction
  - Colorization, inpainting
- Transformation prediction
  - Context prediction, jigsaw puzzle solving, rotation prediction
- “Siamese” methods
  - Contrastive methods
  - Non-contrastive methods
- Self-supervision beyond still images
  - Video, audio, language
Learning from audio

Ego-motion features

D. Jayaraman and K. Grauman. Learning image representations tied to ego-motion. ICCV 2015
Video correspondence features

Future prediction

Future prediction

C. Finn and S. Levine. Deep Visual Foresight for Planning Robot Motion. ICRA 2017
Self-supervised learning in NLP (coming up)

- word2vec, GloVe, BERT, ELMO, GPT, …
Figure 1. **Our MAE architecture.** During pre-training, a large random subset of image patches (e.g., 75%) is masked out. The encoder is applied to the small subset of visible patches. Mask tokens are introduced after the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images to produce representations for recognition tasks.

K. He et al. [Masked autoencoders are scalable vision learners](https://arxiv.org/abs/2110.04499). CVPR 2022
Figure 2. Example results on ImageNet validation images. For each triplet, we show the masked image (left), our MAE reconstruction† (middle), and the ground-truth (right). The masking ratio is 80%, leaving only 39 out of 196 patches. More examples are in the appendix. †As no loss is computed on visible patches, the model output on visible patches is qualitatively worse. One can simply overlay the output with the visible patches to improve visual quality. We intentionally opt not to do this, so we can more comprehensively demonstrate the method’s behavior.

K. He et al. Masked autoencoders are scalable vision learners. CVPR 2022
Figure 8. **MAE pre-training vs. supervised pre-training**, evaluated by fine-tuning in ImageNet-1K (224 size). We compare with the original ViT results [16] trained in IN1K or JFT300M.

K. He et al. *Masked autoencoders are scalable vision learners*. CVPR 2022

<table>
<thead>
<tr>
<th>method</th>
<th>pre-train data</th>
<th>APbox ViT-B</th>
<th>APbox ViT-L</th>
<th>APmask ViT-B</th>
<th>APmask ViT-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>supervised</td>
<td>IN1K w/ labels</td>
<td>47.9</td>
<td>49.3</td>
<td>42.9</td>
<td>43.9</td>
</tr>
<tr>
<td>MoCo v3</td>
<td>IN1K</td>
<td>47.9</td>
<td>49.3</td>
<td>42.7</td>
<td>44.0</td>
</tr>
<tr>
<td>BEiT</td>
<td>IN1K+DALLIE</td>
<td>49.8</td>
<td><strong>53.3</strong></td>
<td>44.4</td>
<td>47.1</td>
</tr>
<tr>
<td>MAE</td>
<td>IN1K</td>
<td><strong>50.3</strong></td>
<td><strong>53.3</strong></td>
<td>44.9</td>
<td>47.2</td>
</tr>
</tbody>
</table>

Table 4. **COCO object detection and segmentation** using a ViT Mask R-CNN baseline. All entries are based on our implementation. Self-supervised entries use IN1K data *without* labels. Mask AP follows a similar trend as box AP.
For further reading

https://github.com/jason718/awesome-self-supervised-learning
https://people.eecs.berkeley.edu/~efros/gelato_bet.html