Boulder

Advanced Topics; Self-Supervised Learning

YouTube Playlist

Maziar Raissi

Assistant Professor

Department of Applied Mathematics

University of Colorado Boulder

maziar.raissi@colorado.edu

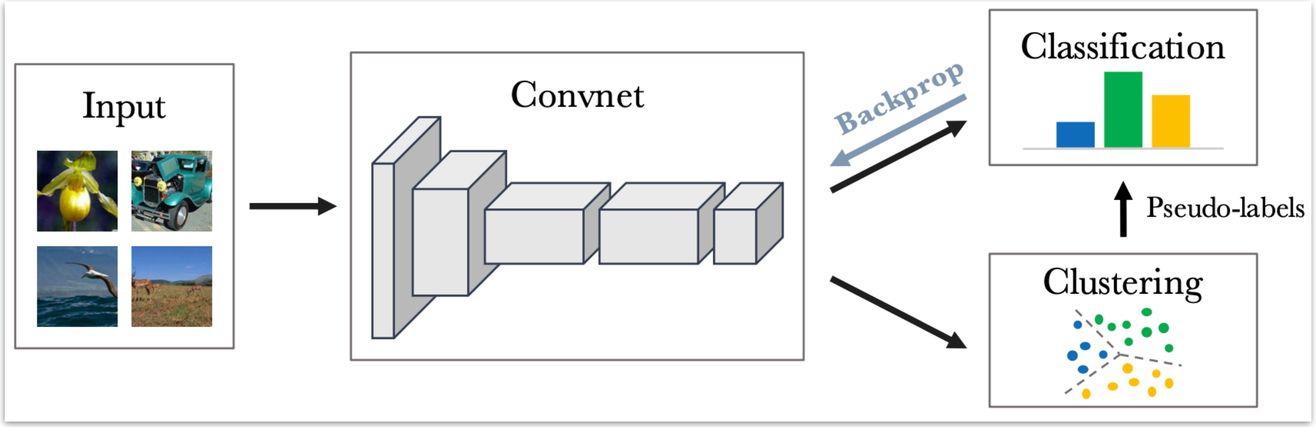
– un-supervised learning

– representation learning

Deep Clustering for Unsupervised Learning of Visual Features

YouTube Video

Key observation: A multilayer perceptron classifier on top of the last convolutional layer of a random AlexNet achieves 12% in accuracy on ImageNet while the chance is at 0.1%.
This is because a convolutional structure gives a strong prior.
$\{x_1, \ldots, x_N\} \rightarrow$ training set of N images
Problem setup: Find parameters θ^* such that the convnet mapping f_{θ^*} produces good general-purpose features (representations).



k-means: cluster the features $f_\theta(x_n)$ into k distinct groups
$C \in \mathbb{R}^{d \times k} \rightarrow$ centroid matrix
$y_n \rightarrow$ cluster assignment of each image

$$\min_{C \in \mathbb{R}^{d \times k}} \frac{1}{N} \sum_{n=1}^{N} \min_{y_n \in \{0,1\}^k} \|f_\theta(x_n) - C y_n\|_2^2 \quad \text{such that} \quad y_n^\top 1_k = 1.$$

$y_n^* \rightarrow$ optimal assignment (pseudo label)
$C^* \rightarrow$ optimal centroid matrix (not used)

$g_W \rightarrow$ parametrized classifier on top of the features $f_\theta(x_n)$

$$\min_{\theta, W} \frac{1}{N} \sum_{n=1}^{N} \ell\left(g_W\left(f_\theta(x_n)\right), y_n\right)$$

$\ell \rightarrow$ multinomial logistic loss

Avoiding Trivial Solutions

Empty clusters:
An optimal decision boundary is to assign all of the inputs to a single cluster.
When a cluster becomes empty, randomly select a non-empty cluster and use its centroid with a small random perturbation as the new centroid for the empty cluster. Then reassign the points belonging to the non-empty cluster to the two resulting clusters.

Trivial parametrization:
Unbalanced number of images per cluster
Sample images based on a uniform distribution over the classes, or pseudo-labels.

PASCAL VOC transfer tasks		Classification		Detection		Segmentation	
Method	Training set	FC6-8	ALL	FC6-8	ALL	FC6-8	ALL
Best competitor	ImageNet	63.0	67.7	43.4†	53.2	35.8†	37.7
DeepCluster	ImageNet	72.0	73.7	51.4	55.4	43.2	45.1
DeepCluster	YFCC100M	67.3	69.3	45.6	53.0	39.2	42.2

Pascal VOC 2007 object detection

Method	AlexNet	VGG-16
ImageNet labels	56.8	67.3
Random	47.8	39.7
Doersch *et al.* [13]	51.1	61.5
Wang and Gupta [63]	47.2	60.2
Wang *et al.* [64]	–	63.2
DeepCluster	**55.4**	**65.9**

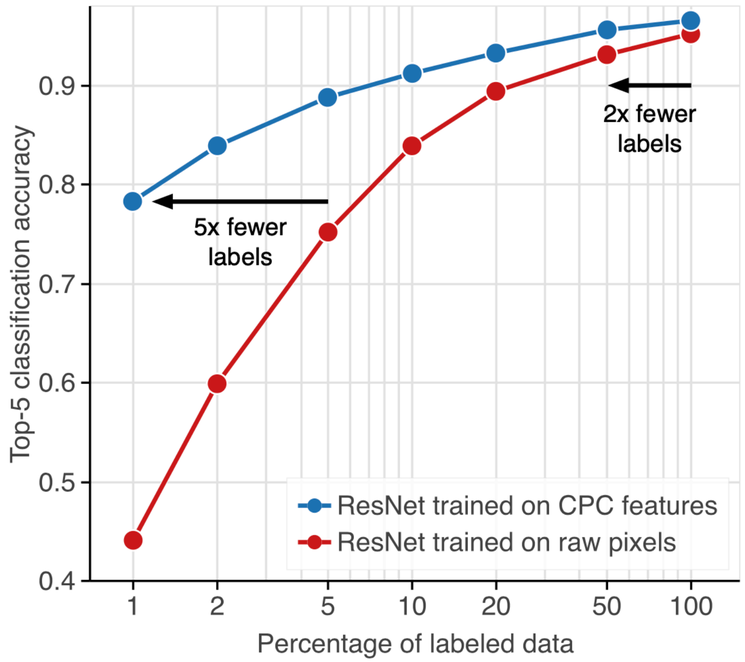
mAP on instance-level image retrieval on Oxford and Paris dataset with a VGG-16

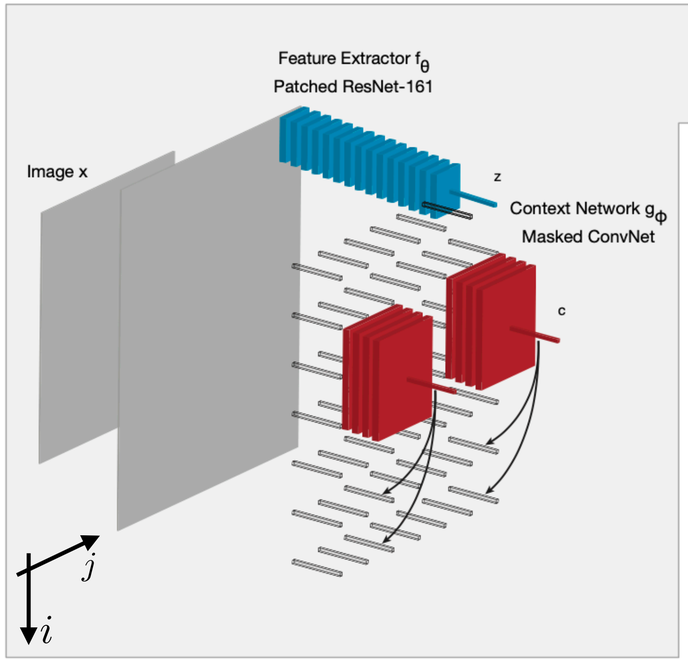
Method	Oxford5K	Paris6K
ImageNet labels	72.4	81.5
Random	6.9	22.0
Doersch *et al.* [13]	35.4	53.1
Wang *et al.* [64]	42.3	58.0
DeepCluster	**61.0**	**72.0**

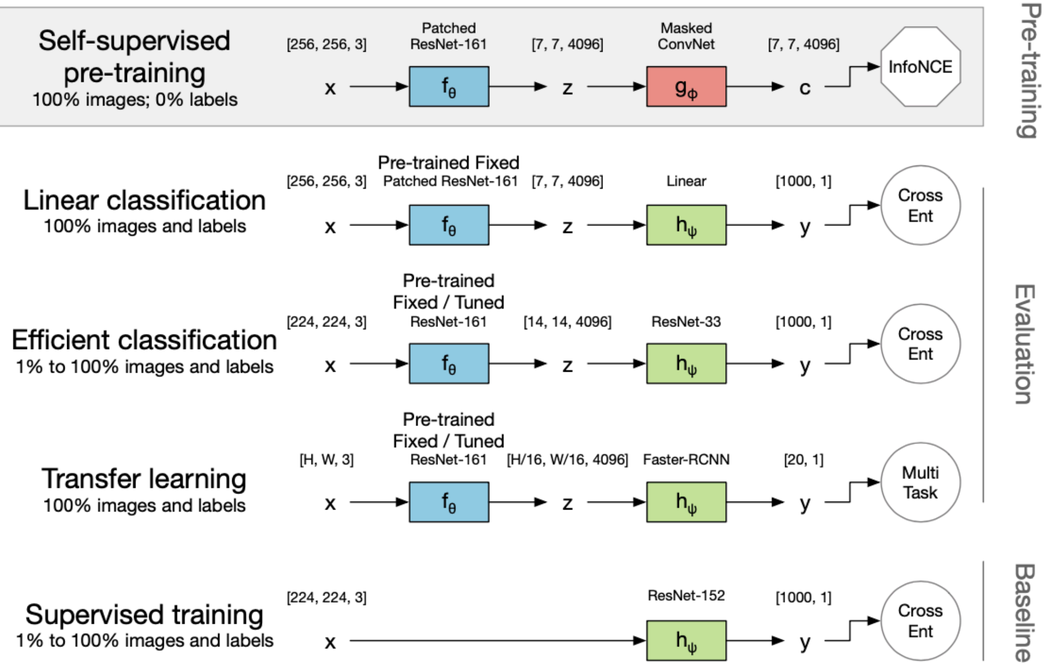
Caron, Mathilde, et al. "Deep clustering for unsupervised learning of visual features." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

Data-Efficient Image Recognition with Contrastive Predictive Coding

YouTube Video







$\{x_{i,j}\} \rightarrow$ a grid of overlapping patches dividing each input image

$i, j \rightarrow$ location of the patch

$z_{i,j} = f_\theta(x_{i,j}) \rightarrow$ encoded patch

$\quad$ ⤷ neural network

$g_\phi \rightarrow$ masked convolutional network (applied to the grid of feature vectors)

$c_{i,j} = g_\phi(\{z_{u,v}\}_{u \le i, v}) \rightarrow$ context vector

$\quad$ ⤷ feature vectors that lie above i, j

$z_{i+k,j} \rightarrow$ "future" feature vectors to be predicted from current context vector $c_{i,j}$

$k \rightarrow$ prediction length

$\hat{z}_{i+k,j} = W_k c_{i,j}$

$\quad$ ⤷ prediction matrix

Correctly recognize the target $z_{i+k,j}$ among a set of randomly sampled feature vectors $\{z_l\}$ from the dataset (i.e., other locations in the image and other images in the mini-batch)

$$\mathcal{L}_{\text{CPC}} = -\sum_{i,j,k} \log p(\boldsymbol{z}_{i+k,j} | \hat{\boldsymbol{z}}_{i+k,j}, \{\boldsymbol{z}_l\})$$

$$= -\sum_{i,j,k} \log \frac{\exp(\hat{\boldsymbol{z}}_{i+k,j}^T \boldsymbol{z}_{i+k,j})}{\exp(\hat{\boldsymbol{z}}_{i+k,j}^T \boldsymbol{z}_{i+k,j}) + \sum_l \exp(\hat{\boldsymbol{z}}_{i+k,j}^T \boldsymbol{z}_l)}$$

InfoNCE (Noise Contrastive Estimation)

– increase depth & width – layer normalization – making predictions in all four directions – extensive patch-based data augmentation

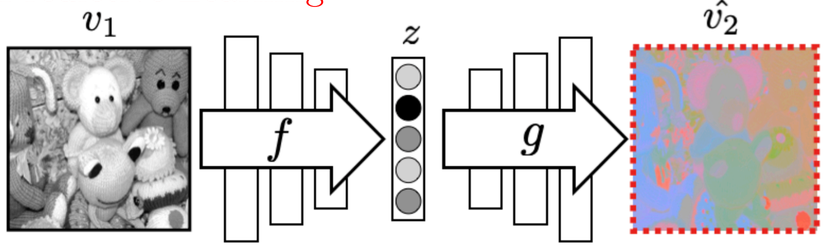
Henaff, Olivier. "Data-efficient image recognition with contrastive predictive coding." *International Conference on Machine Learning*. PMLR, 2020.

Contrastive Multiview Coding

YouTube Playlist

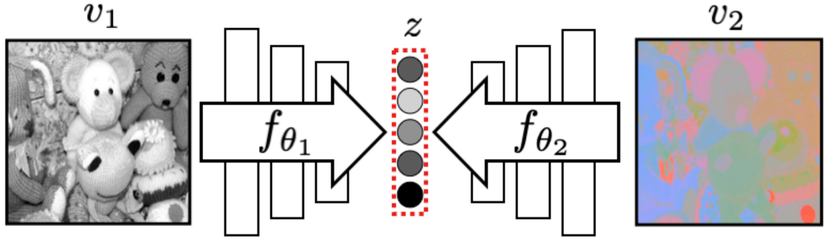
A "dog" can be seen, heard and felt!
$V_1, V_2, \ldots, V_M \rightarrow$ a collection of M views of the data
$v_i \sim \mathcal{P}(V_i) \rightarrow$ a random variable representing samples

Predictive Learning



$V_1, V_2 \rightarrow$ two views of a dataset
(e.g., luminance and chrominance)
$z = f(v_1) \rightarrow$ encoder
$\hat{v}_2 = g(z) \rightarrow$ decoder
$\hookrightarrow$ prediction of v_2 given v_1
$z \rightarrow$ latent variables
Bring $\hat{v}_2$ close to $v_2 \rightarrow$ objective function (L_1 or L_2)
Independence assumption between elements or pixels of v_2 given v_1!
$p(v_2|v_1) = \Pi_i p(v_{2i}|v_1)$

Contrastive Learning with Two Views



$\{v_1^i, v_2^i\}_{i=1}^N \rightarrow$ collection of samples from V_1 and V_2
$x = \{v_1^i, v_2^i\} \rightarrow$ positives (congruent pairs), $x \sim p(v_1, v_2) \rightarrow$ joint distribution
$y = \{v_1^i, v_2^j\} \rightarrow$ negatives (incongruent pairs), $y \sim p(v_1)p(v_2) \rightarrow$ product of marginals
Select a single positive sample x out of a set $S = \{x, y_1, y_2, \ldots, y_k\}$ that contains k negative samples

$$\mathcal{L}_{contrast} = -\mathop{\mathbb{E}}_{S}\left[\log \frac{h_\theta(x)}{h_\theta(x) + \sum_{i=1}^{k} h_\theta(y_i)}\right]$$

$$h_\theta(\{v_1, v_2\}) = \exp\left(\frac{f_{\theta_1}(v_1) \cdot f_{\theta_2}(v_2)}{\|f_{\theta_1}(v_1)\| \cdot \|f_{\theta_2}(v_2)\|} \cdot \frac{1}{\tau}\right)$$

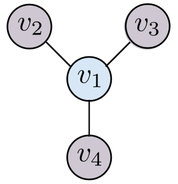
Connecting to Mutual Information

$$h_\theta^*(\{v_1, v_2\}) \propto \frac{p(z_1, z_2)}{p(z_1)p(z_2)} \propto \frac{p(z_1|z_2)}{p(z_1)} \rightarrow \text{point-wise mutual information}$$

$$I(v_i; v_j) \geq I(z_i; z_j) \geq \log(k) - \mathcal{L}_{contrast}$$

Contrastive Learning with More Than Two Views

$$\mathcal{L}_C = \sum_{j=2}^{M} \mathcal{L}(V_1, V_j) \rightarrow \text{core view}$$



$$\mathcal{L}_F = \sum_{1 \leq i < j \leq M} \mathcal{L}(V_i, V_j) \rightarrow \text{full graph}$$

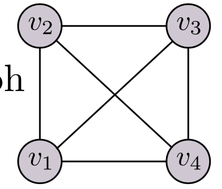


Image Views
luminance (L channel), chrominance (ab channel)

Video Views
$i_t \rightarrow$ image at time t, $i_{t+k} \rightarrow$ neighboring frame, $f_t \rightarrow$ optical flow (10 frames centered at time t)
$(i_t, i_{t+k}), (i_t, f_t) \rightarrow$ two separate contrastive learning objectives

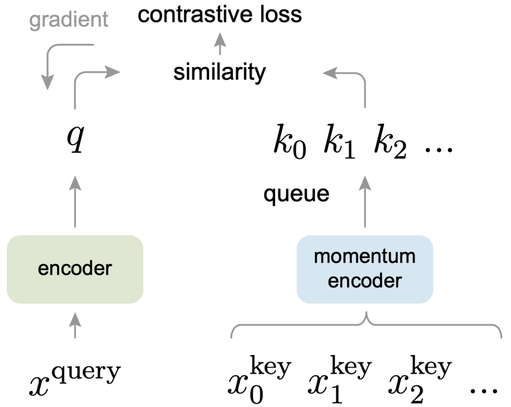
Multiple Image Views
luminance (L channel), chrominance (ab channel), depth, surface normal, and semantic labels

Tian, Yonglong, Dilip Krishnan, and Phillip Isola. "Contrastive multiview coding." Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16. Springer International Publishing, 2020.

Momentum Contrast for Unsupervised Visual Representation Learning

Boulder

YouTube Video



Keys (tokens): samples from the data (e.g., images or patches) represented by an encoder network

Unsupervised Learning: dictionary look-up

An encoded "query" should be similar to its matching "key" and dissimilar to others

Instance discrimination task

A query matches a key if they are encoded views (e.g., different crops) of the same image

Contrastive learning

$q \rightarrow$ encoded query

$\{k_0, k_1, \ldots\} \rightarrow$ set of encoded samples (keys of a dictionary)

$k_+ \rightarrow$ the single key in the dictionary that matches q

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

InfoNCE (noise-contrastive estimation) loss

$\tau \rightarrow$ temperature hyper-parameter

$q = f_q(x^q)$ query sample

encoder network

$k = f_k(x^k)$

Momentum Contrast (MoCo)

① dynamic ② large ③ consistent dictionary

dictionary as queue

The current mini-batch is enqueued to the dictionary, and the oldest mini-batch in the queue is removed

$\theta_k \rightarrow$ parameters of f_k

$\theta_q \rightarrow$ parameters of f_q

$\theta_k \leftarrow m\theta_k + (1-m)\theta_q, m \in [0,1), m = 0.999$

$\theta_q \rightarrow$ updated by backprop

Object detection fine-tuned on PASCAL VOC

pre-train	AP_{50}	AP	AP_{75}
random init.	60.2	33.8	33.1
super. IN-1M	81.3	53.5	58.8
MoCo IN-1M	81.5 (+0.2)	55.9 (**+2.4**)	62.6 (**+3.8**)
MoCo IG-1B	82.2 (**+0.9**)	57.2 (**+3.7**)	63.7 (**+4.9**)

Faster R-CNN

ImageNet-1M (IN-1M), Instagram-1B (IG-1B), super. (supervised)

Algorithm 1 Pseudocode of MoCo in a PyTorch-like style.

```
# f_q, f_k: encoder networks for query and key
# queue: dictionary as a queue of K keys (CxK)
# m: momentum
# t: temperature

f_k.params = f_q.params # initialize
for x in loader: # load a minibatch x with N samples
   x_q = aug(x) # a randomly augmented version
   x_k = aug(x) # another randomly augmented version

   q = f_q.forward(x_q) # queries: NxC
   k = f_k.forward(x_k) # keys: NxC
   k = k.detach() # no gradient to keys

   # positive logits: Nx1
   l_pos = bmm(q.view(N,1,C), k.view(N,C,1))

   # negative logits: NxK
   l_neg = mm(q.view(N,C), queue.view(C,K))

   # logits: Nx(1+K)
   logits = cat([l_pos, l_neg], dim=1)

   # contrastive loss, Eqn.(1)
   labels = zeros(N) # positives are the 0-th
   loss = CrossEntropyLoss(logits/t, labels)

   # SGD update: query network
   loss.backward()
   update(f_q.params)

   # momentum update: key network
   f_k.params = m*f_k.params+(1-m)*f_q.params

   # update dictionary
   enqueue(queue, k) # enqueue the current minibatch
   dequeue(queue) # dequeue the earliest minibatch
```

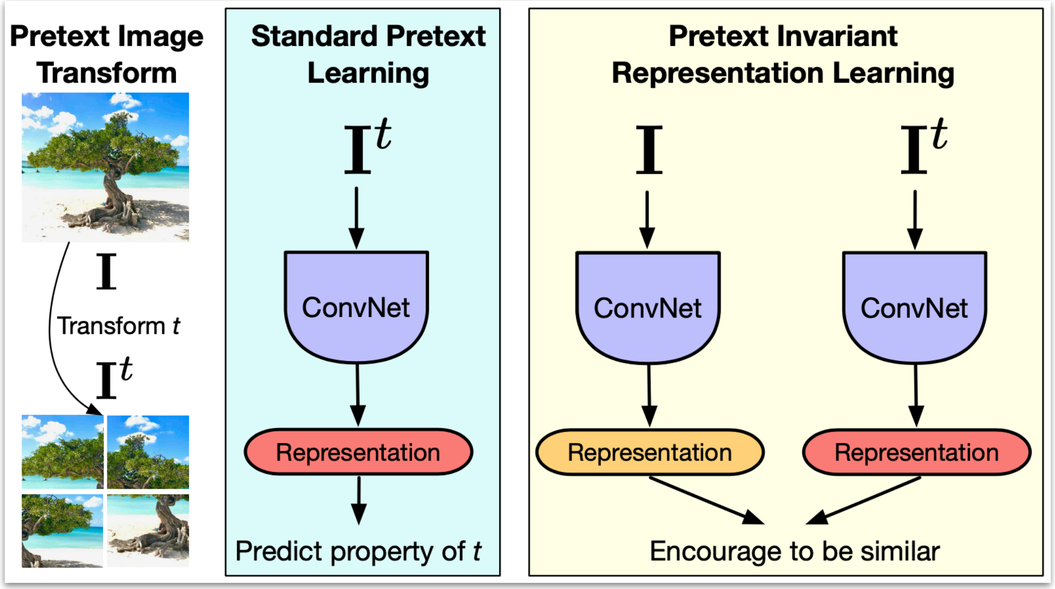
bmm: batch matrix multiplication; mm: matrix multiplication; cat: concatenation.

He, Kaiming, et al. "Momentum contrast for unsupervised visual representation learning." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

Self-Supervised Learning of Pretext-Invariant Representations

YouTube Video

Pretext-Invariant Representation Learning (PIRL)



$\mathcal{D} = \{I_1, \ldots, I_{|\mathcal{D}|}\} \rightarrow$ image dataset

$I_n \in \mathbb{R}^{H \times W \times C}$

$\mathcal{T} \rightarrow$ set of image transformations (e.g., reshuffling of image patches, image rotation, etc.)

$\phi_\theta(\cdot) \rightarrow$ CNN

$v_I = \phi_\theta(I) \rightarrow$ image representation

$I^t = t(I) \rightarrow$ image I after the transformation $t \in \mathcal{T}$

$$\ell_{co}(\theta; \mathcal{D}) = \mathbb{E}_{t \sim p(\mathcal{T})} \left[\frac{1}{|\mathcal{D}|} \sum_{\mathbf{I} \in \mathcal{D}} L_{co}\left(\mathbf{v_I}, z(t)\right) \right]$$

$\rightarrow$ Many pretext tasks (e.g., solving jigsaw puzzles) lead to representations that are covariant with image transformations

$z \rightarrow$ a function that measures some properties of transformation t

Make representations invariant under transformations $t \in \mathcal{T}$

$$\ell_{inv}(\theta; \mathcal{D}) = \mathbb{E}_{t \sim p(\mathcal{T})} \left[\frac{1}{|\mathcal{D}|} \sum_{\mathbf{I} \in \mathcal{D}} L\left(\mathbf{v_I}, \mathbf{v_{I^t}}\right) \right]$$

Loss Function

L measures the similarity between two image representations

$(I, I^t) \rightarrow$ "positive" pair

$N \rightarrow$ number of corresponding "negative" pairs (computing features from other images $I' \neq I$)

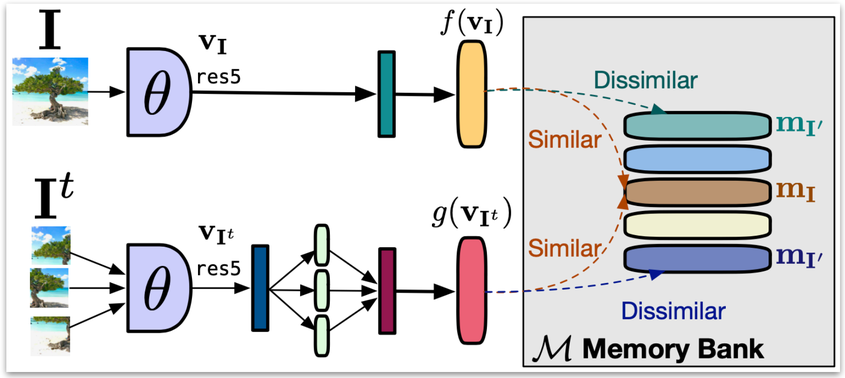
$$h(\mathbf{v_I}, \mathbf{v_{I^t}}) = \frac{\exp\left(\frac{s(\mathbf{v_I}, \mathbf{v_{I^t}})}{\tau}\right)}{\exp\left(\frac{s(\mathbf{v_I}, \mathbf{v_{I^t}})}{\tau}\right) + |\mathcal{D}_N|/|\mathcal{D}|}$$

$\rightarrow$ probability of the binary event that (I, I_t) originates from data distribution

$s(\cdot, \cdot) \rightarrow$ cosine similarity

$$L_{\text{NCE}}\left(\mathbf{I}, \mathbf{I}^t\right) = -\log\left[h\left(f(\mathbf{v_I}), g(\mathbf{v_{I^t}})\right)\right] - \sum_{\mathbf{I}' \in \mathcal{D}_N} \log\left[1 - h\left(g(\mathbf{v}_\mathbf{I}^t), f(\mathbf{v_{I'}})\right)\right]$$

noise contrastive estimator

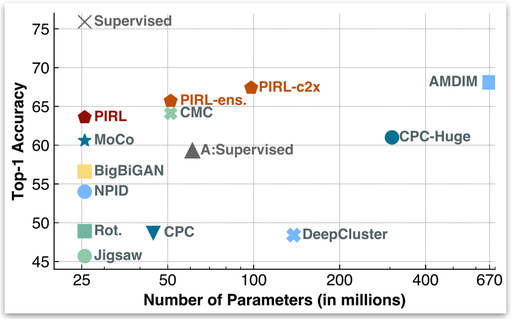


$f, g \rightarrow$ two different heads

$m_I \rightarrow$ exponential moving average of feature representations $f(v_I)$

Final loss function

$$L\left(\mathbf{I}, \mathbf{I}^t\right) = \lambda L_{\text{NCE}}(\mathbf{m_I}, g(\mathbf{v_{I^t}})) + (1-\lambda) L_{\text{NCE}}(\mathbf{m_I}, f(\mathbf{v_I})).$$

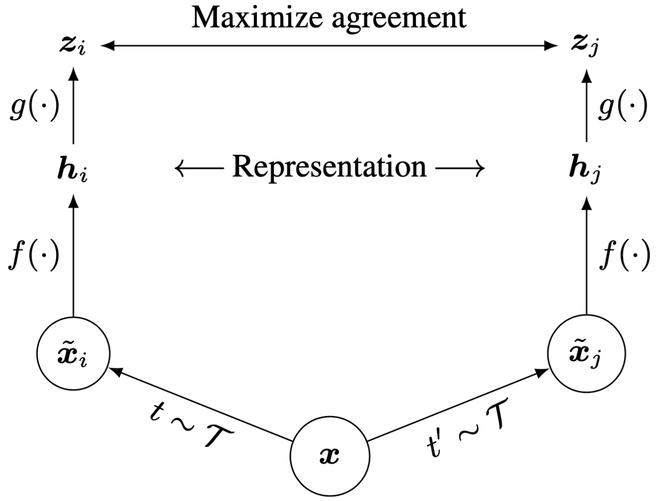


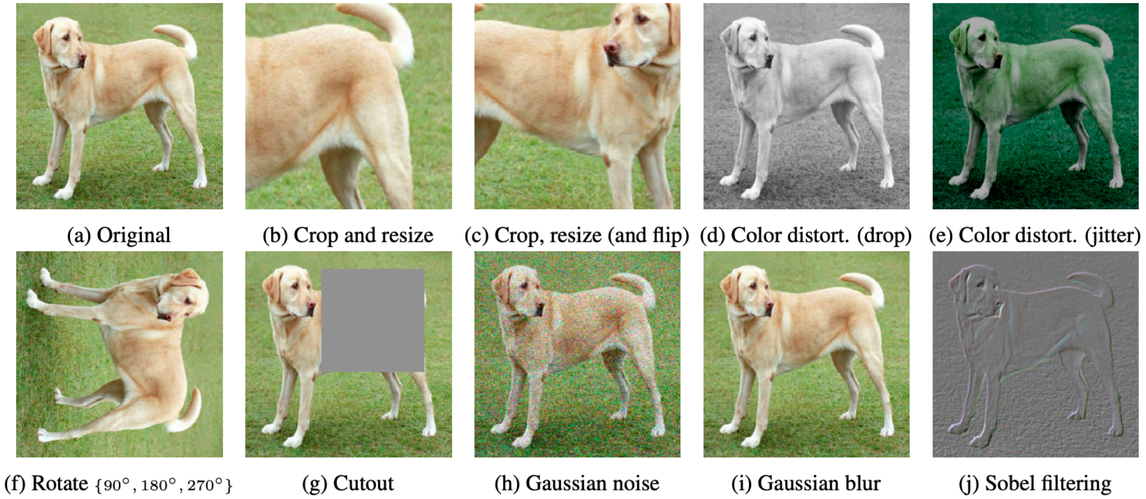
Misra, Ishan, and Laurens van der Maaten. "Self-supervised learning of pretext-invariant representations." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

A Simple Framework for Contrastive Learning of Visual Representations

Boulder

YouTube Video





$\tilde{x}_i, \tilde{x}_j \rightarrow$ two correlated views of the same example

Data Augmentation

Random cropping and resizing to the original size
Random color distortions
Random Gaussian blur

Base Encoder

$h_i = f(\tilde{x}_i) = \text{ResNet}(\tilde{x}_i)$

$h_i \in \mathbb{R}^d \rightarrow$ output after average pooling layer

Projection Head

$z_i = g(h_i) = W^2\sigma(W^1 h_i)$ (σ: ReLU)

Contrastive Loss

Contrastive Prediction Task: Given a set $\{\tilde{x}_k\}$ including a positive pair of examples $\tilde{x}_i$ and $\tilde{x}_j$, identify $\tilde{x}_j$ in $\{\tilde{x}_k\}_{k \neq i}$ for a given $\tilde{x}_i$.

minibatch: N examples $\Longrightarrow$ $2N$ data datapoints

Given a positive pair, treat the other $2(N-1)$ augmented examples as negatives

$\text{sim}(u, v) = u^T v / \|u\| \|v\| \rightarrow$ cosine similarity

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

NT-Xent Loss (normalized temperature-scaled cross entropy loss)

$\tau \rightarrow$ temperature

The final loss is computed over all positive pairs (i, j) & (j, i).

Transfer learning performance

	Food	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
Linear evaluation:												
SimCLR (ours)	**76.9**	**95.3**	80.2	48.4	**65.9**	60.0	61.2	**84.2**	**78.9**	89.2	**93.9**	**95.0**
Supervised	75.2	**95.7**	**81.2**	**56.4**	64.9	**68.8**	**63.8**	83.8	**78.7**	**92.3**	**94.1**	94.2
Fine-tuned:												
SimCLR (ours)	**89.4**	**98.6**	**89.0**	**78.2**	**68.1**	**92.1**	**87.0**	**86.6**	**77.8**	92.1	**94.1**	97.6
Supervised	88.7	98.3	**88.7**	**77.8**	67.0	91.4	**88.0**	86.5	**78.8**	**93.2**	**94.2**	**98.0**
Random init	88.3	96.0	81.9	**77.0**	53.7	91.3	84.8	69.4	64.1	82.7	72.5	92.5

Self-supervised learning on ImageNet

Method	Architecture	Param (M)	Top 1	Top 5
Methods using ResNet-50:				
Local Agg.	ResNet-50	24	60.2	-
MoCo	ResNet-50	24	60.6	-
PIRL	ResNet-50	24	63.6	-
CPC v2	ResNet-50	24	63.8	85.3
SimCLR (ours)	ResNet-50	24	**69.3**	**89.0**
Methods using other architectures:				
Rotation	RevNet-50 (4×)	86	55.4	-
BigBiGAN	RevNet-50 (4×)	86	61.3	81.9
AMDIM	Custom-ResNet	626	68.1	-
CMC	ResNet-50 (2×)	188	68.4	88.2
MoCo	ResNet-50 (4×)	375	68.6	-
CPC v2	ResNet-161 (*)	305	71.5	90.1
SimCLR (ours)	ResNet-50 (2×)	94	74.2	92.0
SimCLR (ours)	ResNet-50 (4×)	375	**76.5**	**93.2**

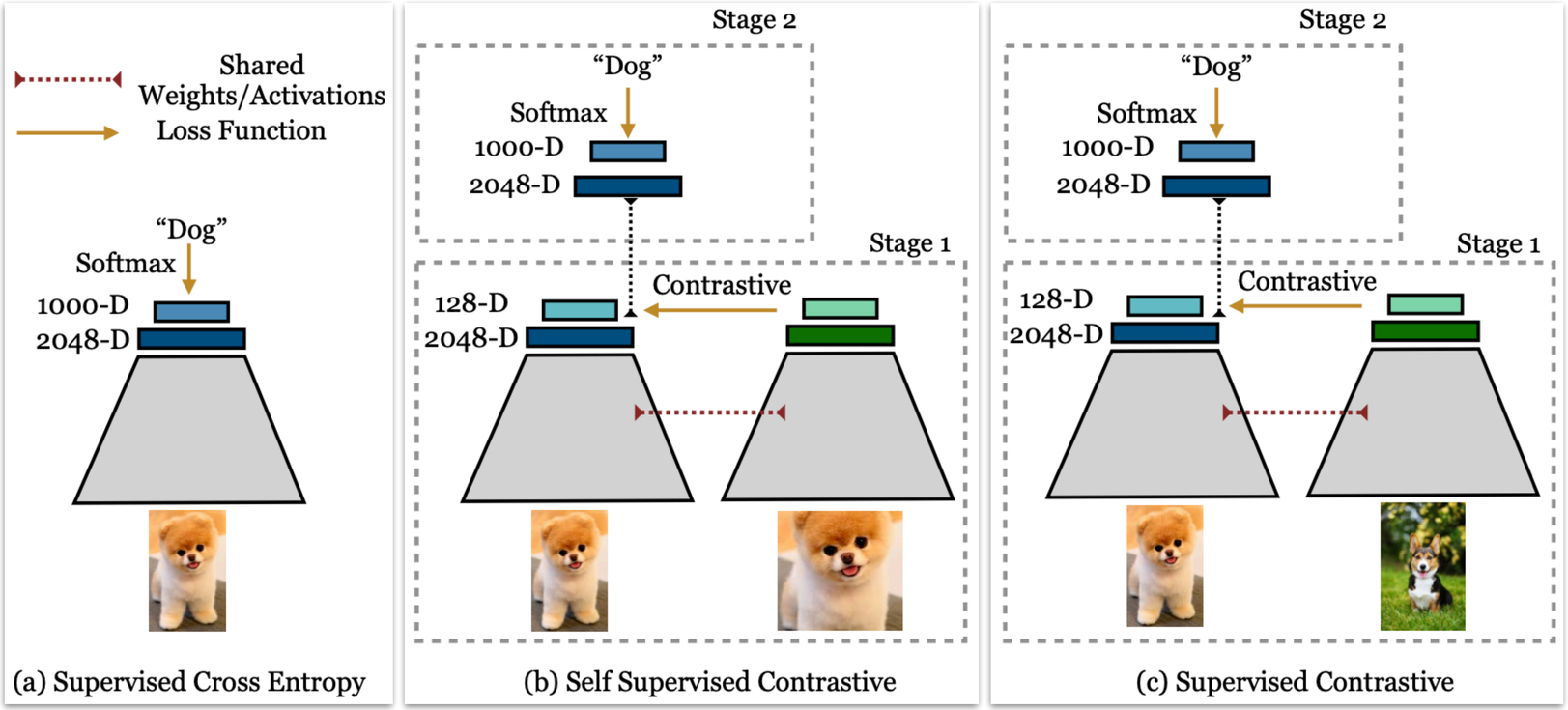
Semi-supervised learning on ImageNet

Method	Architecture	Label fraction (Top 5) 1%	10%
Supervised baseline	ResNet-50	48.4	80.4
Methods using other label-propagation:			
Pseudo-label	ResNet-50	51.6	82.4
VAT+Entropy Min.	ResNet-50	47.0	83.4
UDA (w. RandAug)	ResNet-50	-	88.5
FixMatch (w. RandAug)	ResNet-50	-	89.1
S4L (Rot+VAT+En. M.)	ResNet-50 (4×)	-	91.2
Methods using representation learning only:			
InstDisc	ResNet-50	39.2	77.4
BigBiGAN	RevNet-50 (4×)	55.2	78.8
PIRL	ResNet-50	57.2	83.8
CPC v2	ResNet-161(*)	77.9	91.2
SimCLR (ours)	ResNet-50	75.5	87.8
SimCLR (ours)	ResNet-50 (2×)	83.0	91.2
SimCLR (ours)	ResNet-50 (4×)	**85.8**	**92.6**

Chen, Ting, et al. "A simple framework for contrastive learning of visual representations." *arXiv preprint arXiv:2002.05709* (2020).

Supervised Contrastive Learning

YouTube Video



$\{x_k, y_k\}_{k=1}^{N} \rightarrow$ a set of N randomly sampled example/label pairs (batch)

$\{\tilde{x}_\ell, \tilde{y}_\ell\}_{\ell=1}^{2N} \rightarrow$ corresponding batch used for training (multi-viewed batch)

$\tilde{x}_{2k}, \tilde{x}_{2k-1} \rightarrow$ two random augmentations (a.k.a "views") of x_k for $k = 1, \ldots, N$

$\tilde{y}_{2k} = \tilde{y}_{2k-1} = y_k$

Self-Supervised Contrastive Loss

$i \in I := \{1, \ldots, 2N\} \rightarrow$ index of an arbitrary augmented sample

$j(i) \rightarrow$ index of the other augmented sample originating from the same source sample

$$\mathcal{L}^{self} = \sum_{i \in I} \mathcal{L}_i^{self} = -\sum_{i \in I} \log \frac{\exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_{j(i)}/\tau\right)}{\sum_{a \in A(i)} \exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_a/\tau\right)}$$

$z_\ell = \text{Proj}(\text{Enc}(\tilde{x}_\ell)) \in \mathbb{R}^{D_P}$

$A(i) = I \setminus \{i\}$

$i \rightarrow$ anchor, $j(i) \rightarrow$ positive, $k \in A(i) \setminus \{j(i)\} \rightarrow$ negatives

one positive and $2(N-1)$ negatives

Supervised Contrastive Loss

$P(i) \equiv \{p \in A(i) : \tilde{\boldsymbol{y}}_p = \tilde{\boldsymbol{y}}_i\} \rightarrow$ set of indices of all positives in the multi-viewed batch distinct from i

$$\mathcal{L}_{out}^{sup} = \sum_{i \in I} \mathcal{L}_{out,i}^{sup} = \sum_{i \in I} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_p/\tau\right)}{\sum_{a \in A(i)} \exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_a/\tau\right)}$$

– Generalization to an arbitrary number of positives

– Intrinsic ability to perform hard positive/negative mining

Supervised Contrastive (SupCon) Loss

$\text{Aug}(\cdot) \rightarrow$ data augmentation module

$\tilde{x} = \text{Aug}(x) \rightarrow$ for each sample x generate two random augmentations

$\text{Enc}(\cdot) \rightarrow$ encoder network

$r = \text{Enc}(x) \in \mathbb{R}^{D_E} \rightarrow$ maps x to a representation vector

$\llcorner$ normalized to the unit hyper-sphere ($D_E = 2048$)

$\text{Proj}(\cdot) \rightarrow$ projection network – MLP (to be discarded)

$z = \text{Proj}(r) \in \mathbb{R}^{D_P} \rightarrow$ maps r to a vector z ($D_P = 128$)

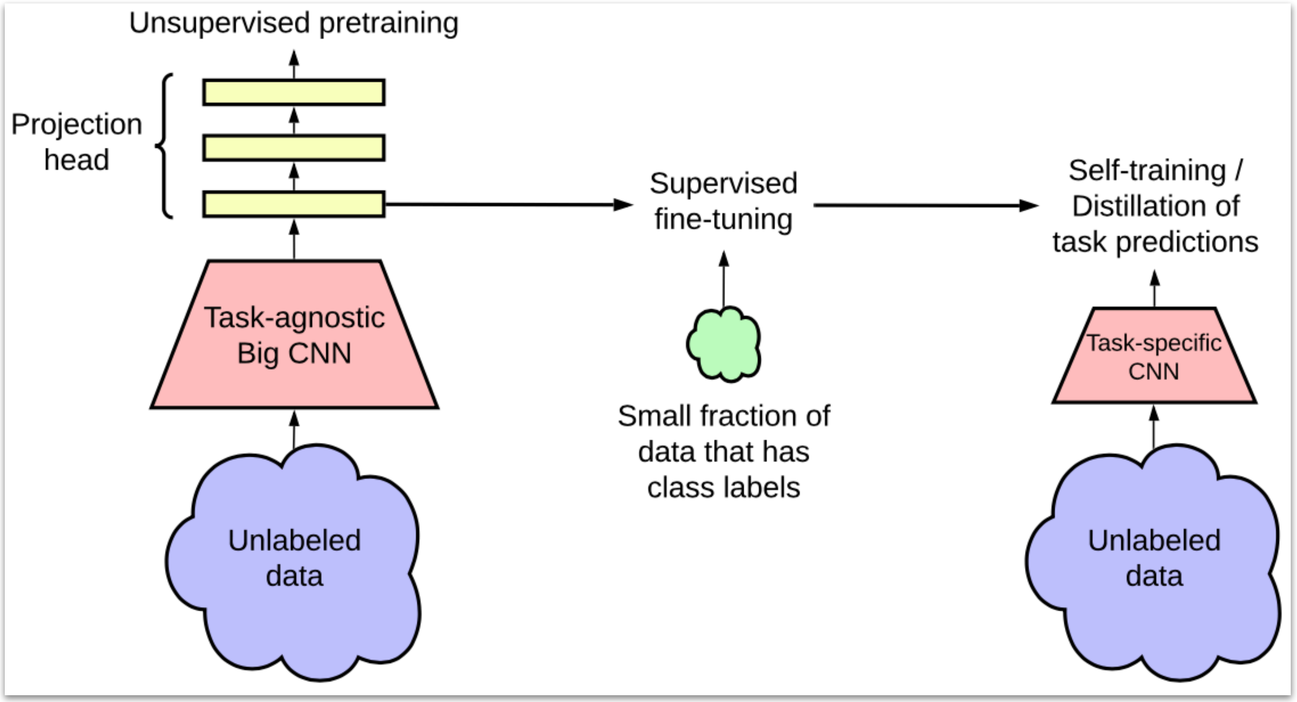
$\llcorner$ normalized to the unit hyper-sphere

Khosla, Prannay, et al. "Supervised contrastive learning." *arXiv preprint arXiv:2004.11362* (2020).

Big Self-Supervised Models are Strong Semi-Supervised Learners

Boulder

YouTube Video



Learning from few labeled examples
(1) unsupervised or self-supervised pretraining
(2) supervised fine-tuning, and
(3) distillation using unlabeled data

Self-supervised pretraining with SimCLRv2

x_{2k-1}, x_{2k} → two views of the same example augmented twice in a randomly sampled mini-batch of images
augmented twice using random crop, color distortion and Gaussian blur
$h_{2k-1}, h_{2k} = f(x_{2k-1}), f(x_{2k})$
f → encoder network (e.g., ResNet)

$z_{2k-1}, z_{2k} = g(h_{2k-1}), g(h_{2k})$ → used for contrastive loss
g → non-linear transformation network (MLP projection head)
i, j → positive examples (i.e., augmented from the same image)

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

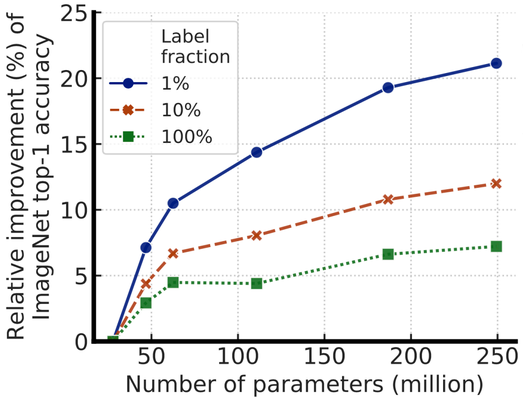
Improvements

(1) Larger ResNet models
ResNet-152(3×+SK)
SK → selective kernels (a channel-wise attention mechanism)

(2) Making g deeper and fine-tuning from a middle layer of g (instead of discarding g entirely)

(3) Incorporate the memory mechanism from MOCO
A memory network (with a moving average of weights for stabilization) whose output will be buffered as negative examples

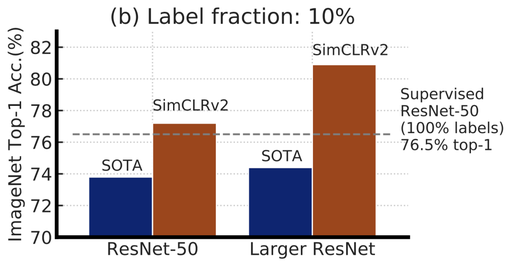
Bigger models yield larger gains when fine-tuning with fewer labeled examples



Self-training / knowledge distillation via unlabeled examples

Teacher: Fine-tuned network

$$\mathcal{L}^{\text{distill}} = -\sum_{\boldsymbol{x}_i \in \mathcal{D}} \left[\sum_y P^T(y|\boldsymbol{x}_i; \tau) \log P^S(y|\boldsymbol{x}_i; \tau) \right]$$

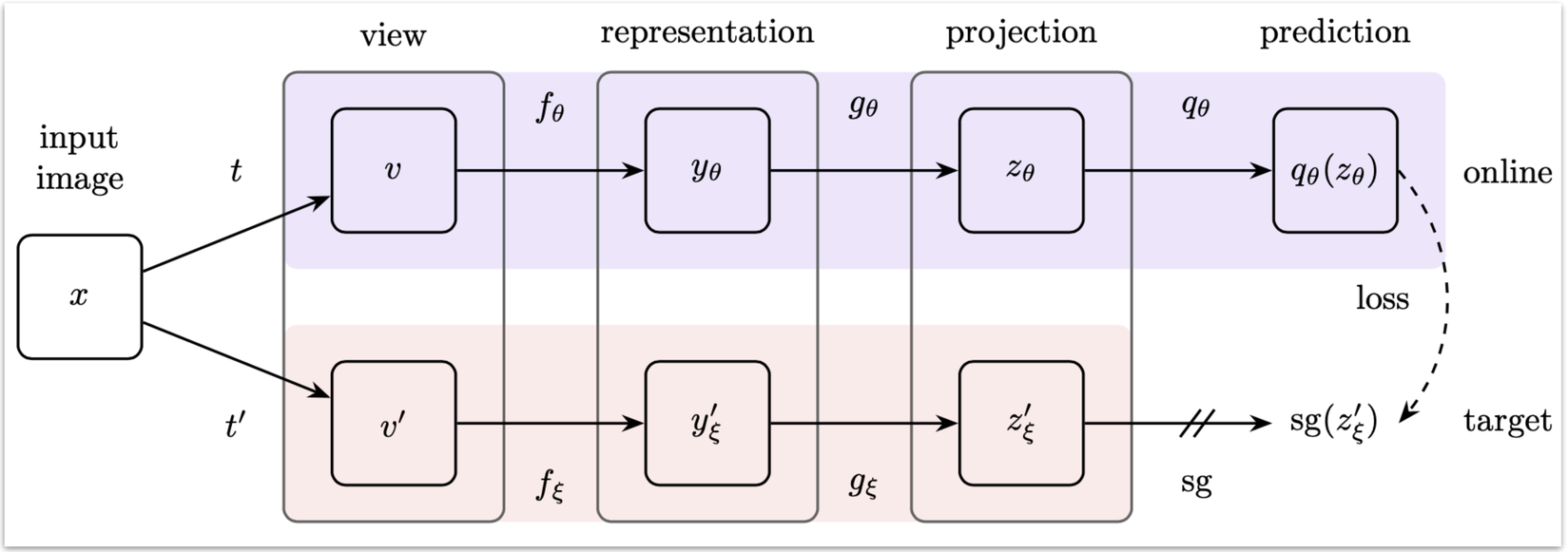


Chen, Ting, et al. "Big self-supervised models are strong semi-supervised learners." *arXiv preprint arXiv:2006.10029* (2020).

Bootstrap your own latent: A new approach to self-supervised Learning

YouTube Playlist

Bootstrap Your Own Latent (BYOL)
Self-supervised image representation learning without using negative pairs! ($\star$)
Learn a representation y_θ which can be used for downstream tasks!



Three stages of the online network:
1. $f_\theta \rightarrow$ encoder
2. $g_\theta \rightarrow$ projector
3. $q_\theta \rightarrow$ predictor ($\star$)

$\theta \rightarrow$ online parameters
$\xi \rightarrow$ parameters of the target network
$\xi \leftarrow \tau\xi + (1-\tau)\theta$ ($\star$)
$\mathcal{D} \rightarrow$ set of images
$x \sim \mathcal{D} \rightarrow$ an image sampled uniformly from $\mathcal{D}$
$\mathcal{T}, \mathcal{T}' \rightarrow$ two distributions of image augmentations

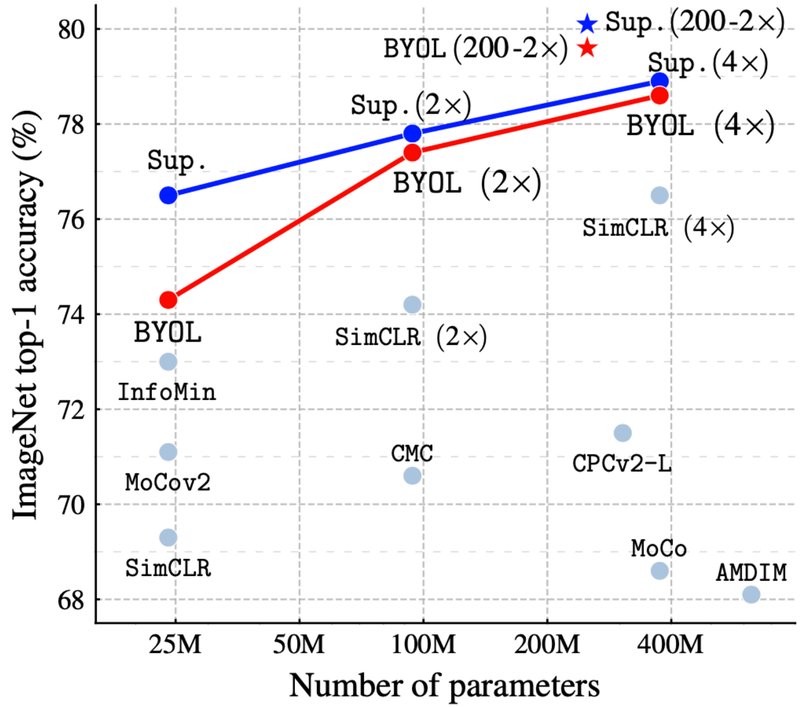
$v = t(x), v' = t'(x) \rightarrow$ two augmented views from x
$t \sim \mathcal{T}, t' \sim \mathcal{T}'$
$y_\theta = f_\theta(v) \rightarrow$ representation
$z_\theta = g_\theta(y_\theta) \rightarrow$ projection
$y'_\xi = f'_\xi(v') \rightarrow$ target representation
$z'_\xi = g'_\xi(y'_\xi) \rightarrow$ target projection
$q_\theta(z_\theta) \rightarrow$ prediction of z'_ξ
$\bar{q}_\theta(z_\theta), \bar{z}'_\xi \rightarrow l_2$ normalized

$$\mathcal{L}_{\theta,\xi} \triangleq \left\|\overline{q_\theta}(z_\theta) - \bar{z}'_\xi\right\|_2^2 = 2 - 2 \cdot \frac{\langle q_\theta(z_\theta), z'_\xi \rangle}{\left\|q_\theta(z_\theta)\right\|_2 \cdot \left\|z'_\xi\right\|_2}$$

$$\mathcal{L}^{\mathrm{BYOL}}_{\theta,\xi} = \mathcal{L}_{\theta,\xi} + \widetilde{\mathcal{L}}_{\theta,\xi}$$

(feeding v' to the online network and v to the target network

While this objective admits collapsed solutions, e.g., outputting the same vector for all images, the paper empirically shows that BYOL does not converge to such solutions. It hypothesizes (see Section 3.2) that the combination of (i) the addition of a predictor to the online network and (ii) the use of a slow-moving average of the online parameters as the target network encourages encoding more and more information within the online projection and avoids collapsed solutions.

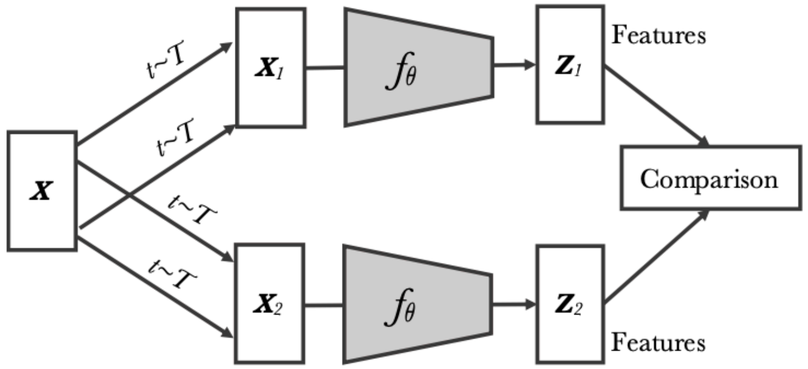


Grill, Jean-Bastien, et al. "Bootstrap your own latent: A new approach to self-supervised learning." *arXiv preprint arXiv:2006.07733* (2020).

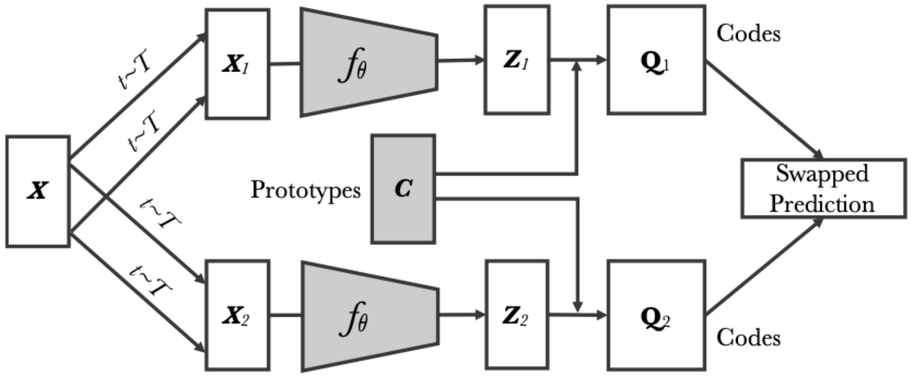
Unsupervised Learning of Visual Features by Contrasting Cluster Assignments

Boulder

YouTube Video



Contrastive instance learning



Swapping Assignments between Views (SwAV)

Swapped prediction problem

$$\ell(\mathbf{z}_t, \mathbf{q}_s) = -\sum_k \mathbf{q}_s^{(k)} \log \mathbf{p}_t^{(k)}$$

$$\mathbf{p}_t^{(k)} = \frac{\exp\left(\frac{1}{\tau}\mathbf{z}_t^\top \mathbf{c}_k\right)}{\sum_{k'} \exp\left(\frac{1}{\tau}\mathbf{z}_t^\top \mathbf{c}_{k'}\right)}$$

$$-\frac{1}{N}\sum_{n=1}^{N}\sum_{s,t\sim\mathcal{T}}\left[\frac{1}{\tau}\mathbf{z}_{nt}^\top \mathbf{C}\mathbf{q}_{ns} + \frac{1}{\tau}\mathbf{z}_{ns}^\top \mathbf{C}\mathbf{q}_{nt} - \log\sum_{k=1}^{K}\exp\left(\frac{\mathbf{z}_{nt}^\top \mathbf{c}_k}{\tau}\right) - \log\sum_{k=1}^{K}\exp\left(\frac{\mathbf{z}_{ns}^\top \mathbf{c}_k}{\tau}\right)\right]$$

Minimized with respect to the prototypes C and the parameters θ

DeepCluster (offline) v.s. SwAV (online)

Offline: requires a pass over the entire dataset to form image "codes" (i.e., cluster assignments)

$z_t, z_s \rightarrow$ two image features from two different augmentations of the same image

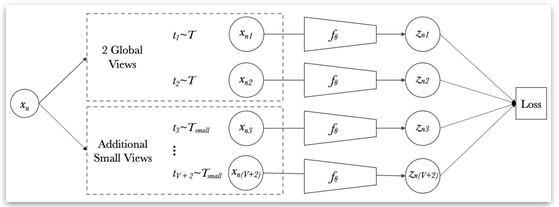
$\{c_1, \ldots, c_K\} \rightarrow$ set of K prototypes

$q_t, q_s \rightarrow$ codes (matching z_t, z_s to the set of K prototypes)

$L(z_t, z_s) = \ell(z_t, q_s) + \ell(z_s, q_t) \rightarrow$ "swapped" prediction problem

$\ell(z, q) \rightarrow$ measures the fit between features z and a code q

multi-crop strategy



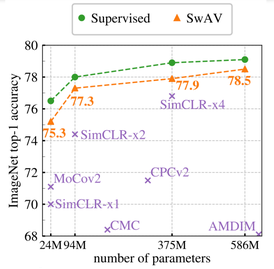
Online Clustering

$t \sim \mathcal{T} \rightarrow$ transformation t sampled from the set of image augmentations $\mathcal{T}$

$x_{nt} \rightarrow$ augmented view of image x_n transformed by t

$$z_{nt} = \frac{f_\theta(x_{nt})}{\|f_\theta(x_{nt})\|_2} \rightarrow \text{image features}$$

$q_{nt} \rightarrow$ code computed by mapping z_{nt} to a set of K trainable prototype vectors



Computing codes online

Trivial solution: every image having the same code

$Z = [z_1, \ldots, z_B] \rightarrow B$ feature vectors

$B \rightarrow$ mini-batch size

$Q = [q_1, \ldots, q_B] \rightarrow$ codes

$$\max_{\mathbf{Q}\in\mathcal{Q}} \operatorname{Tr}\left(\mathbf{Q}^\top \mathbf{C}^\top \mathbf{Z}\right) + \varepsilon H(\mathbf{Q})$$

$H(\mathbf{Q}) = -\sum_{ij} \mathbf{Q}_{ij} \log \mathbf{Q}_{ij} \rightarrow$ entropy

$$\mathcal{Q} = \left\{\mathbf{Q} \in \mathbb{R}_+^{K\times B} \mid \mathbf{Q}\mathbf{1}_B = \frac{1}{K}\mathbf{1}_K, \mathbf{Q}^\top \mathbf{1}_K = \frac{1}{B}\mathbf{1}_B\right\}$$

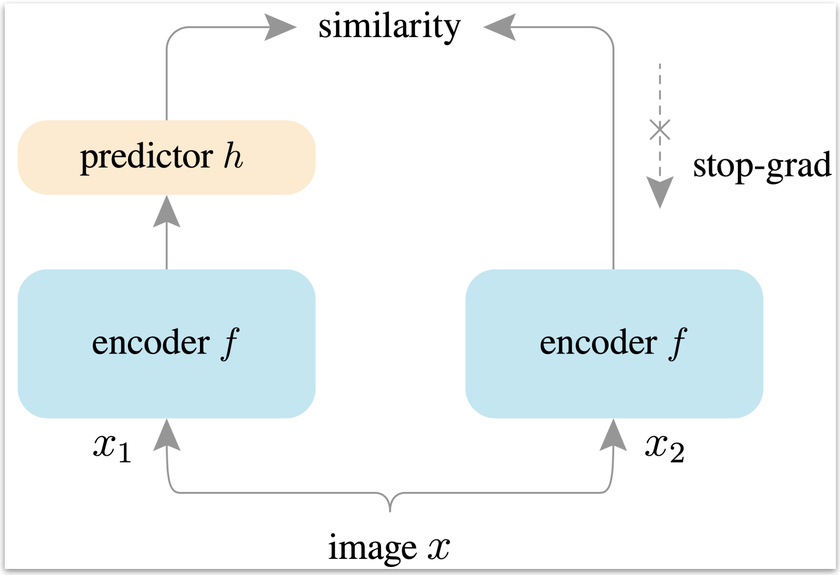
Enforce that on average each prototype is selected at least B/K times in the batch

Three iterations of iterative Sinkhorn-Knopp algorithm!

Caron, Mathilde, et al. "Unsupervised learning of visual features by contrasting cluster assignments." *arXiv preprint arXiv:2006.09882* (2020).

Exploring Simple Siamese Representation Learning

YouTube Video



$x_1, x_2 \rightarrow$ two randomly augmented views of an image x
$f \rightarrow$ encoder network (ResNet backbone and a projection MLP head)
$h \rightarrow$ prediction MLP head

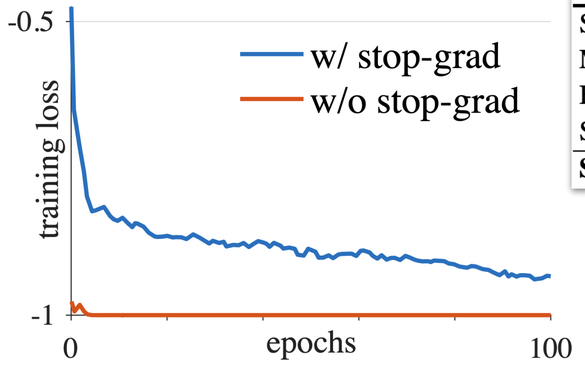
$$p_1 = h(\underbrace{f(x_1)}_{z_1})$$

$$p_2 = h(\underbrace{f(x_2)}_{z_2})$$

$$\mathcal{D}(p_1, z_2) = -\frac{p_1}{\|p_1\|_2} \frac{z_2}{\|z_2\|_2}$$ → negative cosine similarity (to be minimized)
equivalent to the mean squared error of l_2-normalized vectors, up to a scale of 2

$$\mathcal{L} = \frac{1}{2}\mathcal{D}(p_1, z_2) + \frac{1}{2}\mathcal{D}(p_2, z_1)$$ → symmetrized loss
stop gradients

SimSiam uses none of the following:
1) negative sample pairs
2) large batches
3) momentum encoders



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

Effect of prediction MLP

	pred. MLP h	acc. (%)
baseline	lr with cosine decay	67.7
(a)	no pred. MLP	0.1
(b)	fixed random init.	1.5
(c)	lr not decayed	68.1

Effect of batch sizes

batch size	64	128	256	512	1024	2048	4096
acc. (%)	66.1	67.3	68.1	68.1	68.0	67.9	64.0

Effect of batch normalization on MLP heads

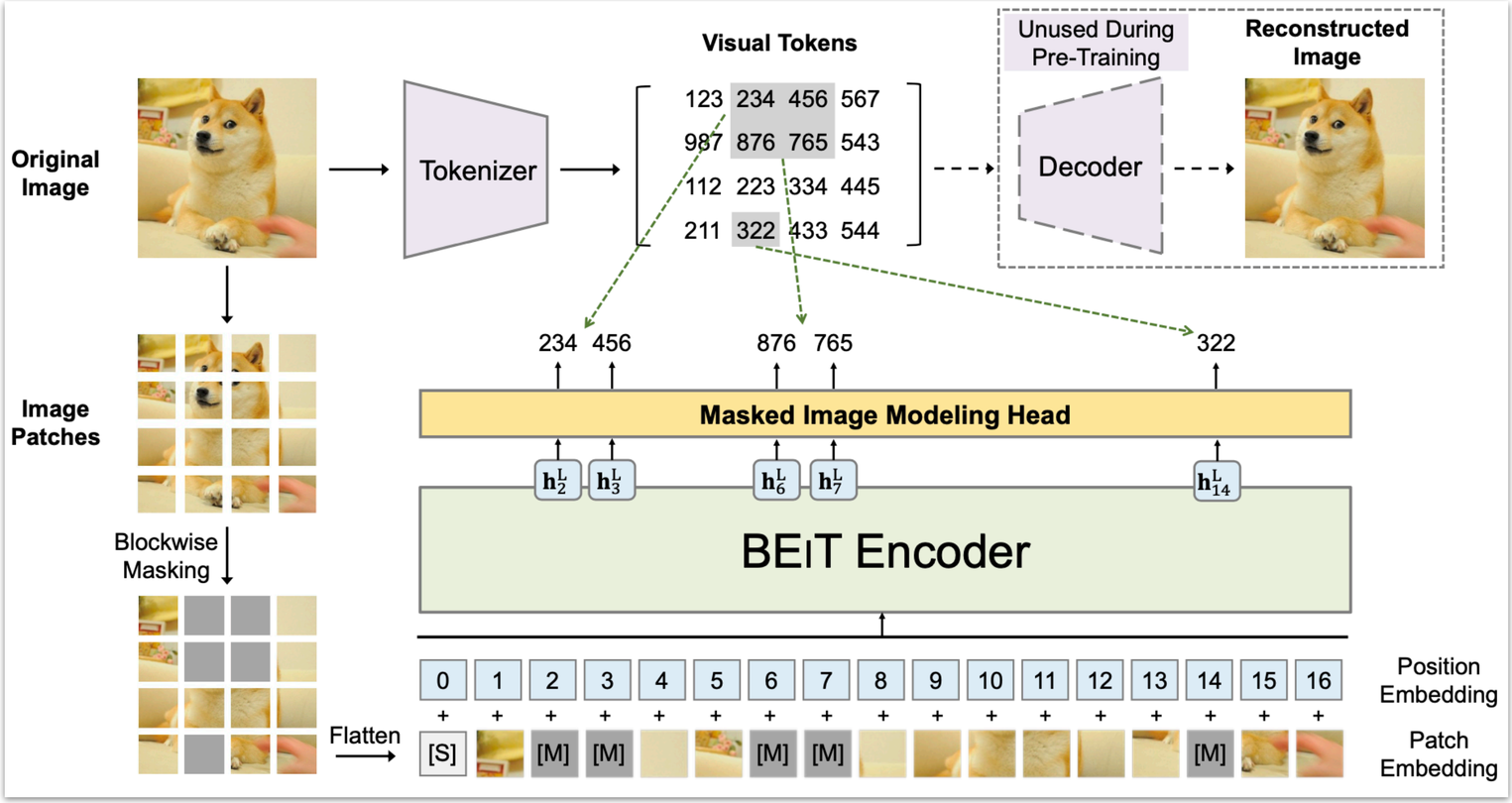
	case	proj. MLP's BN hidden	proj. MLP's BN output	pred. MLP's BN hidden	pred. MLP's BN output	acc. (%)
(a)	none	-	-	-	-	34.6
(b)	hidden-only	✓	-	✓	-	67.4
(c)	default	✓	✓	✓	-	68.1
(d)	all	✓	✓	✓	✓	unstable

Chen, Xinlei, and Kaiming He. "Exploring simple siamese representation learning." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

BEIT: BERT Pre-Training of Image Transformers

YouTube Video

Bidirectional Encoder representation from Image Transformers



Visual Token

image tokenizer (discrete variational autoencoder)

tokenize $x \in \mathbb{R}^{H \times W \times C}$ into $z \in \mathcal{V}^{h \times w}$

$\mathcal{V} = \{1, \ldots, |\mathcal{V}|\}$ → vocabulary containing discrete token indices

https://github.com/openai/DALL-E

$q_\phi(z|x)$ → tokenizer (maps image x into discrete tokens z according to a visual codebook/vocabulary)

$p_\psi(x|z)$ → decoder (reconstruct the input image x based on the visual tokens z)

$$\max_{\phi,\psi} \mathbb{E}_{z \sim q_\phi(z|x)}[\log p_\psi(x|z)] - \mathrm{KL}[q_\phi(z|x) \parallel p(z)]$$

Gumbel-softmax relaxation

Algorithm 1 Blockwise Masking

Input: $N(= h \times w)$ image patches
Output: Masked positions $\mathcal{M}$
$\mathcal{M} \leftarrow \{\}$
repeat
  $s \leftarrow \mathrm{Rand}(16, 0.4N - |\mathcal{M}|)$ ▷ *Block size*
  $r \leftarrow \mathrm{Rand}(0.3, \frac{1}{0.3})$ ▷ *Aspect ratio of block*
  $a \leftarrow \sqrt{s \cdot r}; b \leftarrow \sqrt{s/r}$
  $t \leftarrow \mathrm{Rand}(0, h - a)\,; l \leftarrow \mathrm{Rand}(0, w - b)$
  $\mathcal{M} \leftarrow \mathcal{M} \bigcup \{(i, j) : i \in [t, t + a), j \in [l, l + b)\}$
until $|\mathcal{M}| > 0.4N$ ▷ *Masking ratio is 40%*
return $\mathcal{M}$

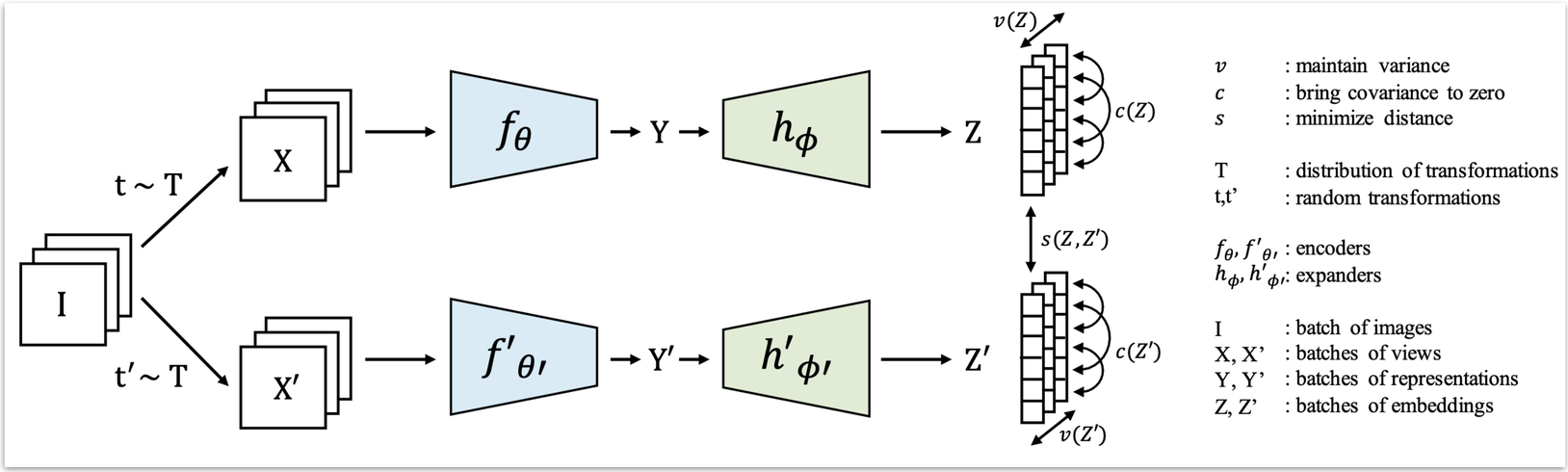
Models	CIFAR-100	ImageNet
Training from scratch (i.e., random initialization)		
ViT$_{384}$ (Dosovitskiy et al., 2020)	48.5*	77.9
DeiT (Touvron et al., 2020)	n/a	81.8
Supervised Pre-Training on ImageNet-1K (using labeled data)		
ViT$_{384}$ (Dosovitskiy et al., 2020)	87.1	77.9
DeiT (Touvron et al., 2020)	90.8	81.8
Self-Supervised Pre-Training on ImageNet-1K (without labeled data)		
iGPT-1.36B† (Chen et al., 2020a)	n/a	66.5
ViT$_{384}$-JFT300M‡ (Dosovitskiy et al., 2020)	n/a	79.9
DINO (Caron et al., 2021)	91.7	82.8
MoCo v3 (Chen et al., 2021)	87.1	n/a
BEIT (ours)	90.1	**83.2**
Self-Supervised Pre-Training, and Intermediate Fine-Tuning on ImageNet-1K		
BEIT (ours)	**91.8**	**83.2**

Top-1 accuracy of image classification

Bao, Hangbo, Li Dong, and Furu Wei. "BEiT: BERT Pre-Training of Image Transformers." *arXiv preprint arXiv:2106.08254* (2021).

VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning

YouTube Video



Method	Linear Top-1	Linear Top-5
Supervised	76.5	-
MoCo He et al. (2020)	60.6	-
PIRL Misra & Maaten (2020)	63.6	-
CPC v2 Hénaff et al. (2019)	63.8	-
CMC Tian et al. (2019)	66.2	-
SimCLR Chen et al. (2020a)	69.3	89.0
MoCo v2 Chen et al. (2020c)	71.1	-
SimSiam Chen & He (2020)	71.3	-
SwAV Caron et al. (2020)	71.8	-
InfoMin Aug Tian et al. (2020)	73.0	91.1
OBoW Gidaris et al. (2021)	73.8	-
BYOL Grill et al. (2020)	74.3	91.6
SwAV (w/ multi-crop) Caron et al. (2020)	75.3	-
Barlow Twins Zbontar et al. (2021)	73.2	91.0
VICReg (ours)	73.2	91.1

$i \sim \mathcal{D} \rightarrow$ an image sampled from a dataset

$t, t' \sim \mathcal{T} \rightarrow$ two transformations sampled from a distribution

$x = t(i), x' = t'(i) \rightarrow$ two different views of i

random crops of the image, followed by color distortions

$y = f_\theta(x), y' = f_\theta(x') \rightarrow$ representations

$f_\theta \rightarrow$ encoder (outputs the representations used for downstream tasks)

$z = h_\phi(y), z' = h_\phi(y') \rightarrow$ embeddings

$h_\phi \rightarrow$ expander (maps the representations into an embedding space where the loss function will be computed)

$Z = [z_1, \ldots, z_n], Z' = [z'_1, \ldots, z'_n] \rightarrow$ two batches composed of n vectors of dimension d

$$v(Z) = \frac{1}{d}\sum_{j=1}^{d} \max(0, \gamma - S(z^j, \epsilon)) \rightarrow$$ variance regularization term (hinge function on the standard deviation of the embeddings along the batch dimension)

$S(x, \epsilon) = \sqrt{\mathrm{Var}(x) + \epsilon} \rightarrow$ regularized standard deviation

$$C(Z) = \frac{1}{n-1}\sum_{i=1}^{n}(z_i - \bar{z})(z_i - \bar{z})^T, \quad \text{where} \quad \bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i$$

covariance matrix of Z

$$\text{covariance regularization} \leftarrow c(Z) = \frac{1}{d}\sum_{i \neq j}[C(Z)]_{i,j}^2$$

encourages the off-diagonal terms to be close to zero (decorrelating the different dimensions of the embeddings and preventing them from encoding similar information)

$$s(Z, Z') = \frac{1}{n}\sum_{i} \|z_i - z'_i\|_2^2$$

$$\ell(Z, Z') = \lambda s(Z, Z') + \mu[v(Z) + v(Z')] + \nu[c(Z) + c(Z')]$$

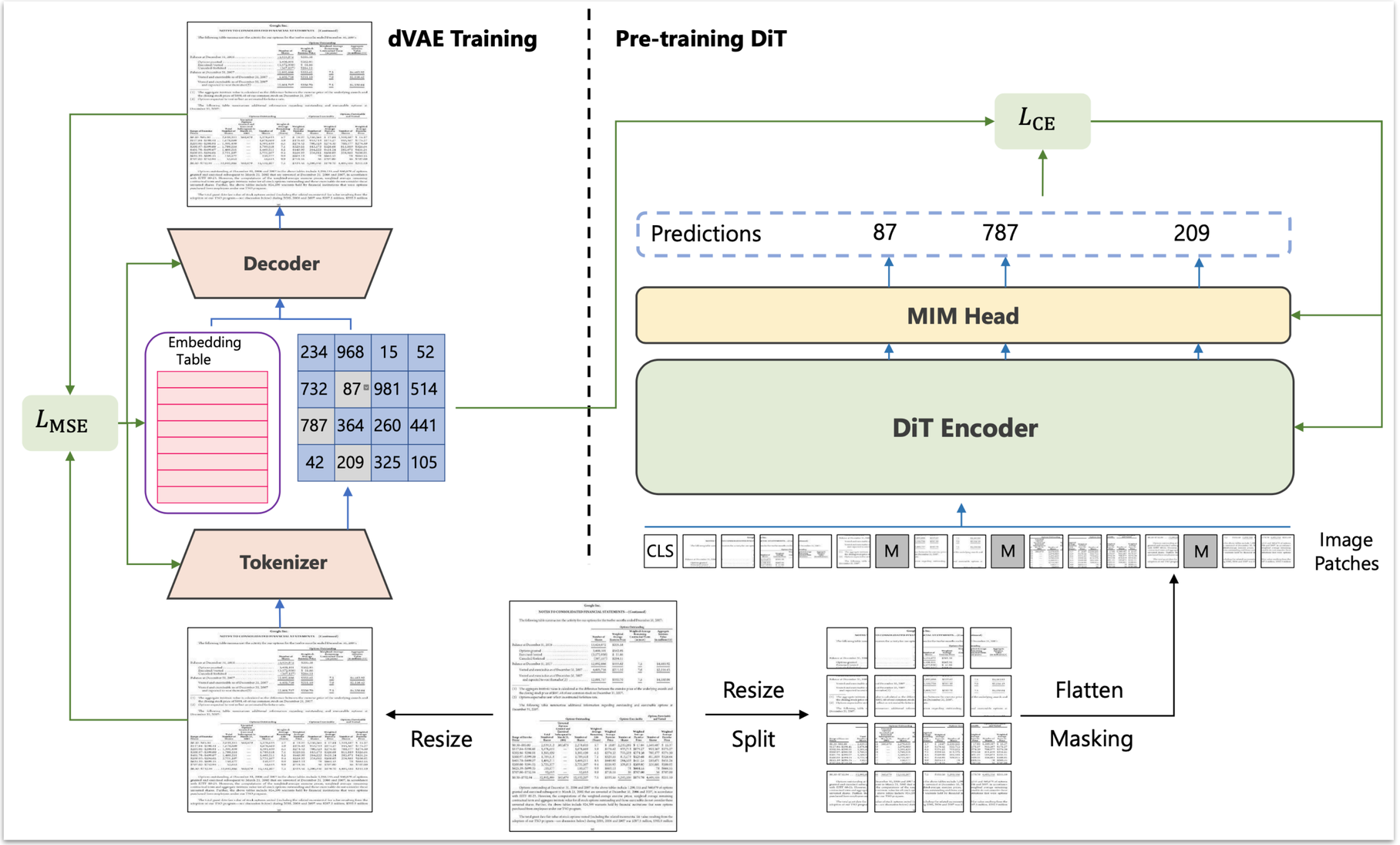
$$\mathcal{L} = \sum_{I \in \mathcal{D}} \sum_{t, t' \sim \mathcal{T}} \ell(Z^I, Z'^I)$$

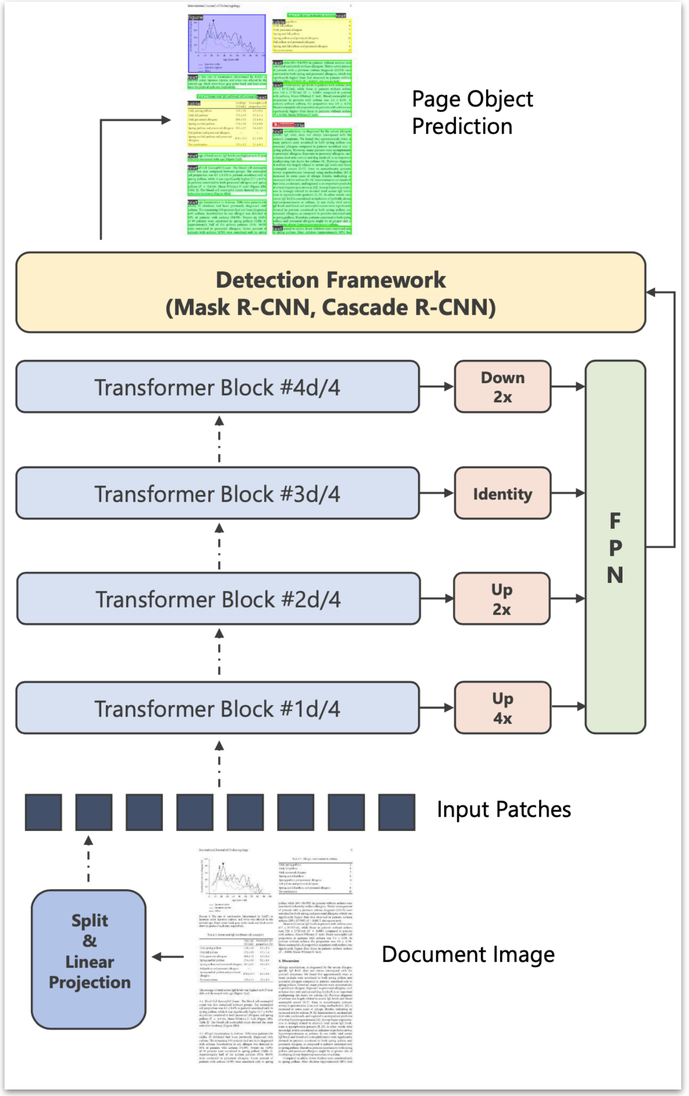
Bardes, Adrien, Jean Ponce, and Yann LeCun. "Vicreg: Variance-invariance-covariance regularization for self-supervised learning." *arXiv preprint arXiv:2105.04906* (2021).

DiT: Self-supervised Pre-training for Document Image Transformer

YouTube Video

forms, invoice/receipts, reports, etc. document image classification, document layout analysis, as well as table detection





Bao, Hangbo, Li Dong, and Furu Wei. "BEiT: BERT Pre-Training of Image Transformers." *arXiv preprint arXiv:2106.08254* (2021).

Unsupervised Semantic Segmentation by Distilling Feature Correspondences

separate feature learning from cluster compactification
STEGO (**S**elf-supervised **T**ransformer with **E**nergy-based **G**raph **O**ptimization)
– CocoStuff – Cityscapes

Feature Correspondences Predict Class Co-occurrence

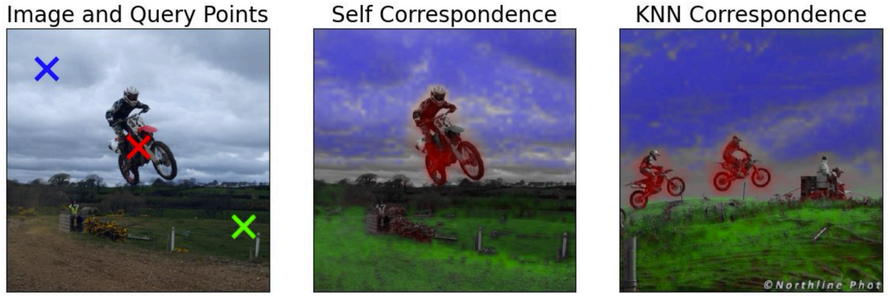
$f \in \mathbb{R}^{CWH}, g \in \mathbb{R}^{CIJ} \rightarrow$ feature tensors for two different images

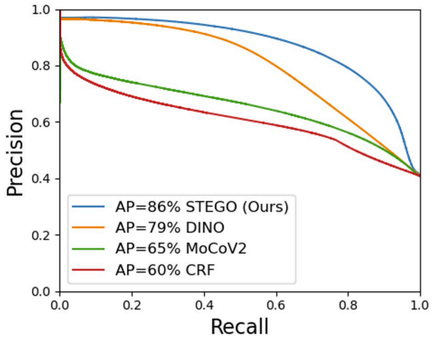
$$F_{hwij} := \sum_c \frac{f_{chw}}{|f_{hw}|} \frac{g_{cij}}{|g_{ij}|} \rightarrow \text{feature correspondence tensor}$$

$k \in \mathcal{C}^{HW}, l \in \mathcal{C}^{IJ} \rightarrow$ ground-truth semantic segmentation labels
$\mathcal{C} \rightarrow$ set of possible classes

$$L_{hwij} := \begin{cases} 1, & \text{if } l_{hw} = k_{ij} \\ 0, & \text{if } l_{hw} \neq k_{ij} \end{cases} \rightarrow \text{label co-occurrence tensor}$$

examine how well F (logits) predicts L





Feature correspondences from DINO.

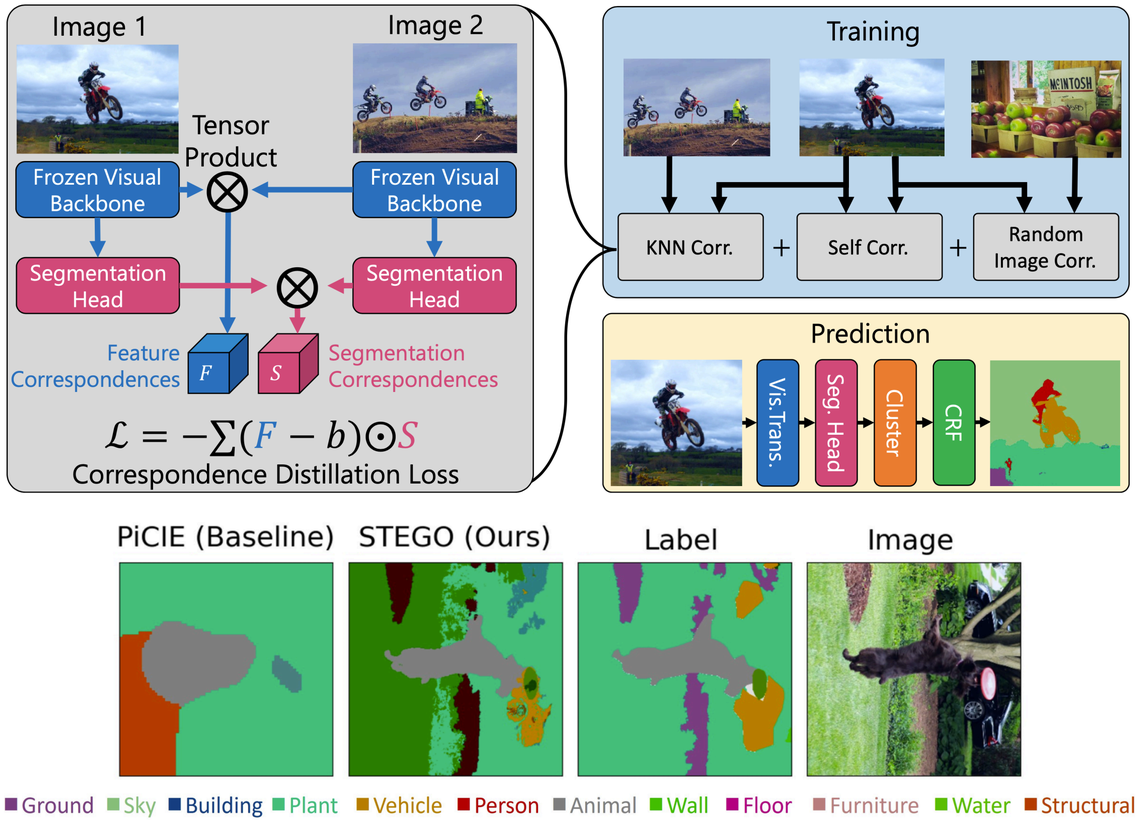
Distilling Feature Correspondences

$S \rightarrow$ segmentation head (MLP)
$s := S(f), t := S(g) \rightarrow$ semantic feature of two feature tensors f, g from a pair of images x, y
$S \rightarrow$ segmentation correlation tensor from s and t

$$\mathcal{L}_{simple-corr}(x, y, b) := -\sum_{hwij} (F_{hwij} - b) S_{hwij}$$

$$F^{SC}_{hwij} := F_{hwij} - \frac{1}{IJ} \sum_{i'j'} F_{hwi'j'}$$

$$\mathcal{L}_{corr}(x, y, b) := -\sum_{hwij} (F^{SC}_{hwij} - b) max(S_{hwij}, 0)$$



Hamilton, Mark, et al. "Unsupervised Semantic Segmentation by Distilling Feature Correspondences." *arXiv preprint arXiv:2203.08414* (2022).

Boulder

Questions?

YouTube Playlist