Barlow Twins: Self-Supervised Learning via Redundancy Reduction

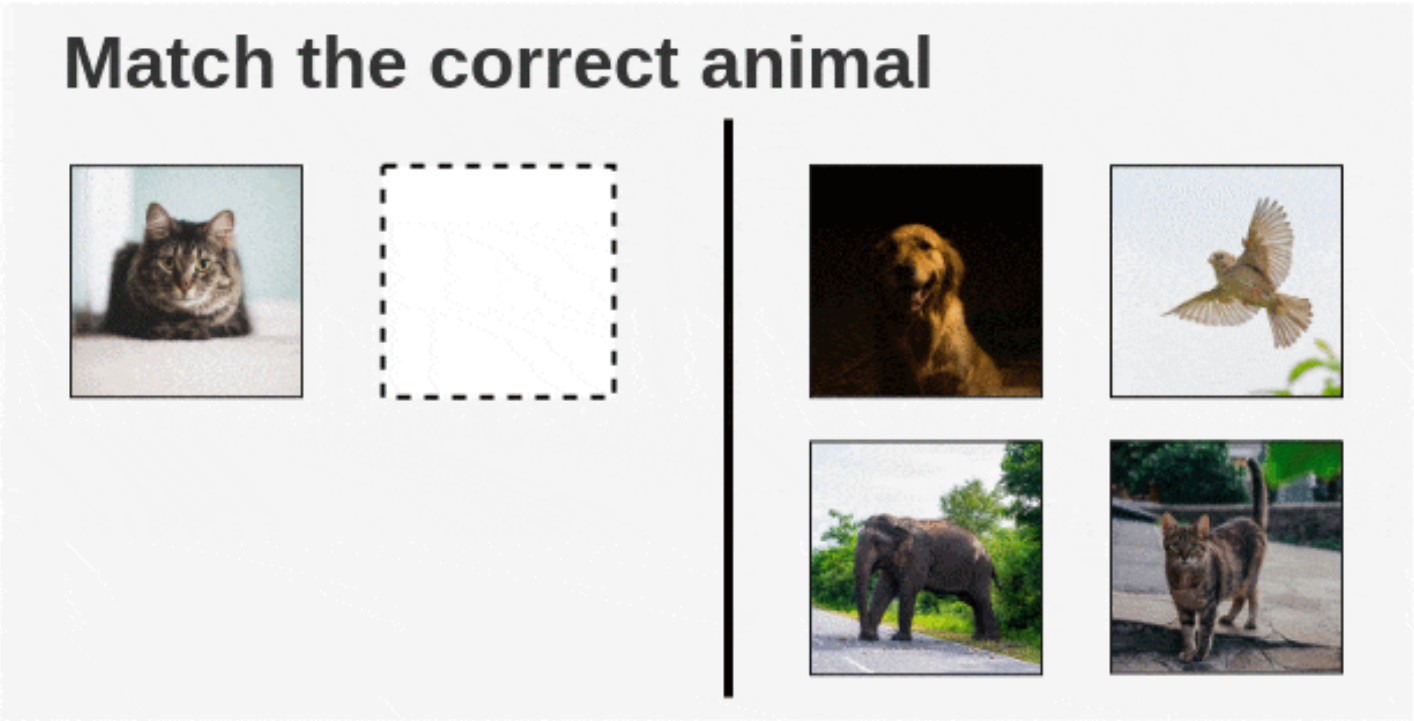
Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, Stephane Deny

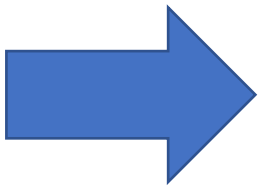
Facebook AI Researcch

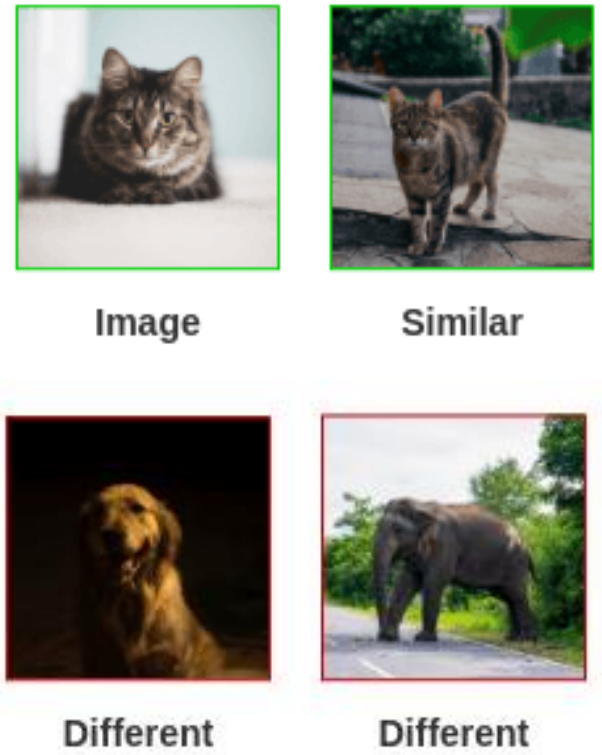
https://arxiv.org/pdf/2103.03230.pdf

발표자: 김진희

Contrastive learning?



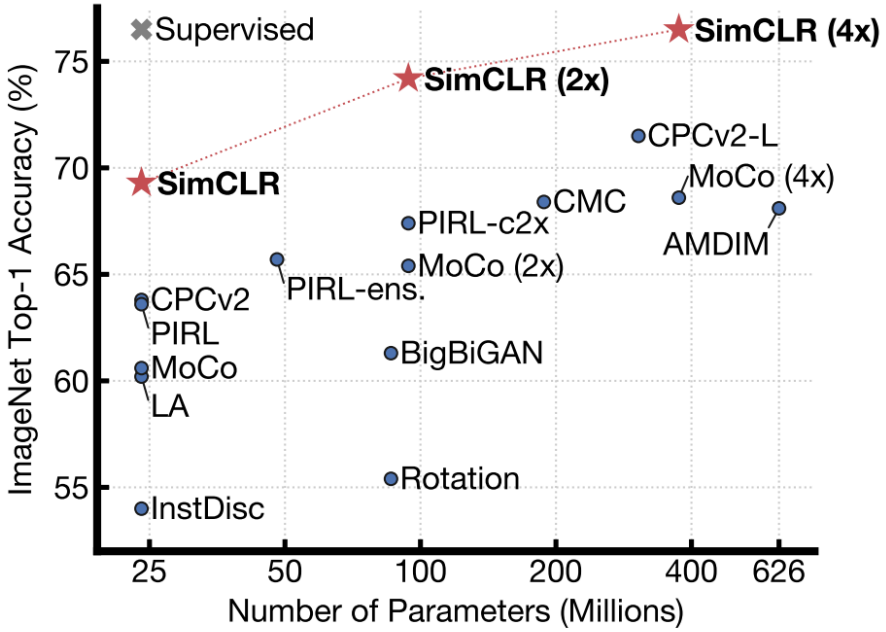




https://amitness.com/2020/03/illustrated-simclr/

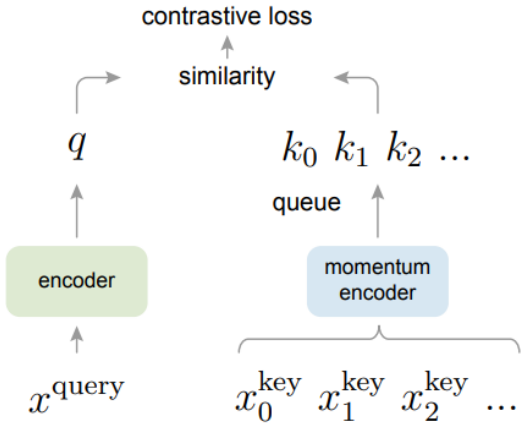
Contrastive learning?

- **(2019.11) Momentum Contrast for Unsupervised Visual Representation Learning. (MoCo)**
- **(2020.01) A Simple Framework for Contrastive Learning of Visual Representations (SimCLR)**
- **(2020.03) Improved Baselines with Momentum Contrastive Learning.(MoCo v2)**
- **(2020.06) Big Self-Supervised Models are Strong Semi-Supervised Learners.(SimCLR v2)**
- **(2020.06) Bootstrap your own latent: A new approach to self-supervised Learning (BYOL)**
- **(2020.06) Unsupervised Learning of Visual Features by Contrasting Cluster Assignments (SwAV)**
- **(2020.11) Exploring Simple Siamese Representation Learning (SimSiam)**
- **(2021.03) Barlow Twins: Self-Supervised Learning via Redundancy Reduction**



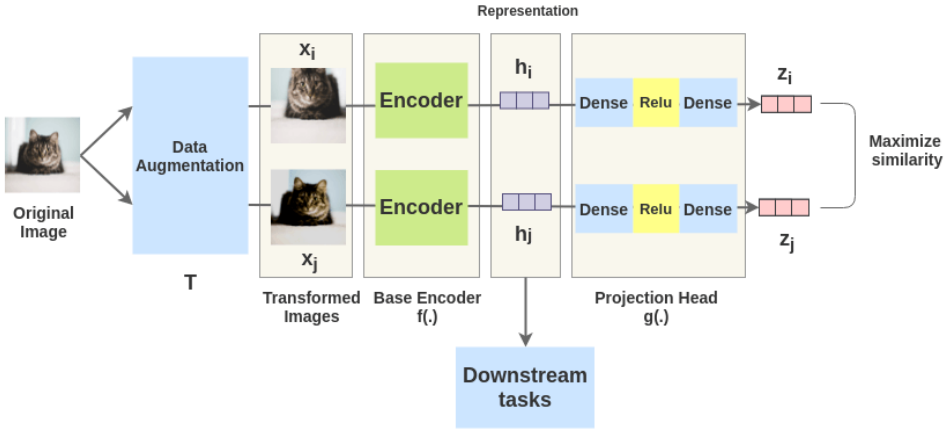
Method	Top-1	Top-5
Supervised	76.5	
MoCo	60.6	
PIRL	63.6	-
SimCLR	69.3	89.0
MoCo v2	71.1	90.1
SimSiam	71.3	-
SwAV (w/o multi-crop)	71.8	-
BYOL	74.3	91.6
SwAV	75.3	-
Barlow Twins (ours)	73.2	91.0

Introduction



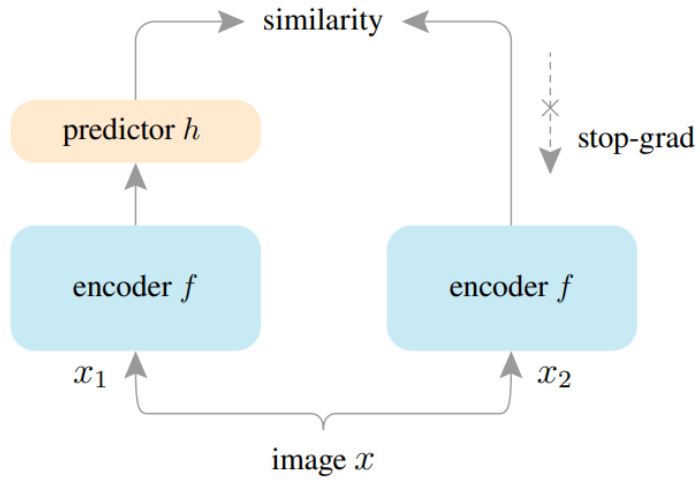
MoCo

Momentum encoder



SimCLR

Large batch size



SimSiam

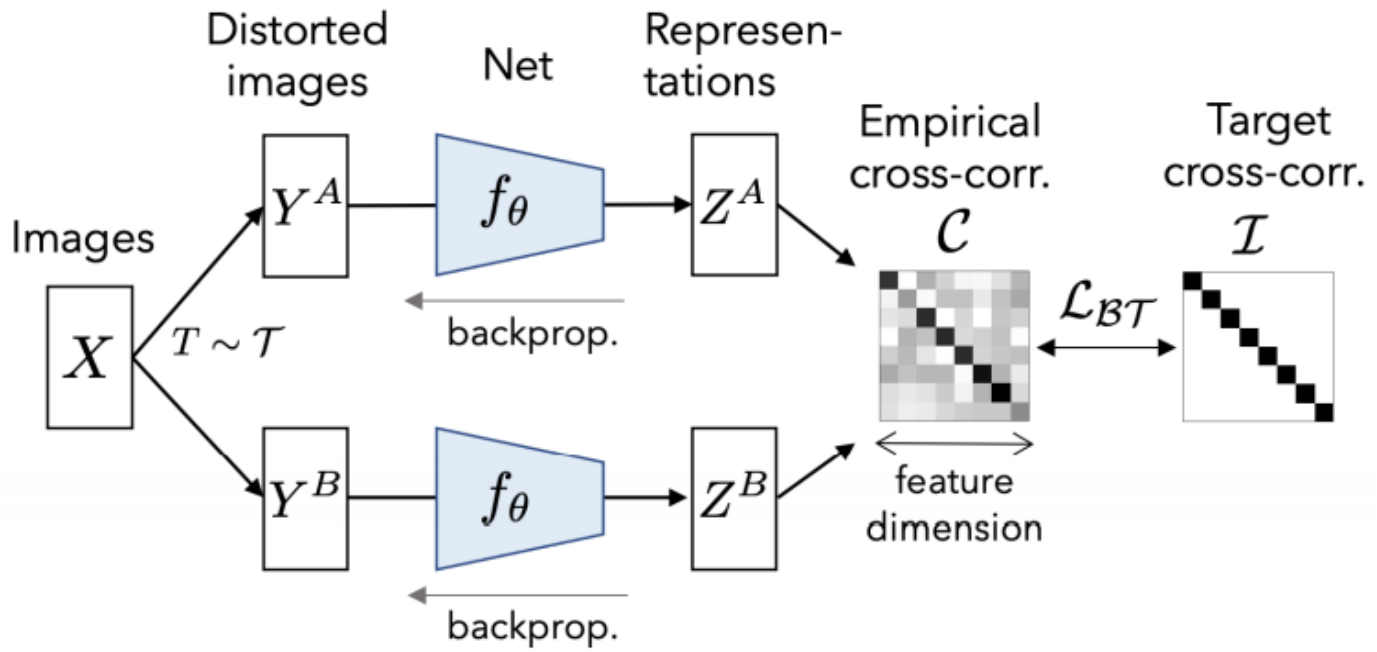
Stop gradient

SwAV Non-differentiable operators

BYOL Asymmetry between network architectures

- **BARLOW TWINS** **does not require** large batches nor asymmetry between the network twins such as a predictor network, gradient stopping, or a moving average on the weight updates

Method



$$\mathcal{L}_{\mathcal{BT}} \triangleq \underbrace{\sum_i (1 - \mathcal{C}_{ii})^2}_{\text{invariance term}} + \lambda \underbrace{\sum_i \sum_{j \neq i} \mathcal{C}_{ij}{}^2}_{\text{redundancy reduction term}}$$

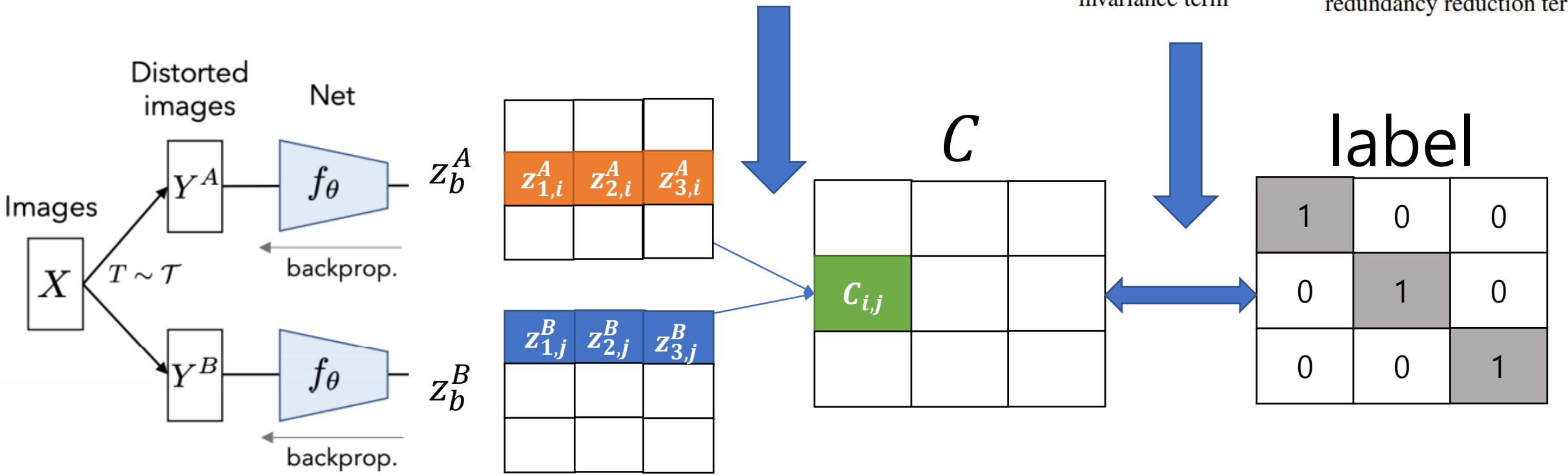
$$\mathcal{C}_{ij} \triangleq \frac{\sum_b z^A_{b,i} z^B_{b,j}}{\sqrt{\sum_b \left(z^A_{b,i}\right)^2} \sqrt{\sum_b \left(z^B_{b,j}\right)^2}}$$

1. Data augmentation $X \rightarrow Y^A, Y^B$
2. Forward propagation + normalization along batch axis $\mathrm{Z^A} = Norm(f_\theta(Y^A)),\ \mathrm{Z^B} = Norm(f_\theta(Y^B))$
3. Computation of cross-correlation matrix C
4. Loss calculation and backpropagation

Method

$$\mathcal{C}_{ij} \triangleq \frac{\sum_b z^A_{b,i} z^B_{b,j}}{\sqrt{\sum_b \left(z^A_{b,i}\right)^2} \sqrt{\sum_b \left(z^B_{b,j}\right)^2}}$$

$$\mathcal{L}_{\mathcal{BT}} \triangleq \underbrace{\sum_i (1 - \mathcal{C}_{ii})^2}_{\text{invariance term}} + \lambda \underbrace{\sum_i \sum_{j \neq i} \mathcal{C}_{ij}{}^2}_{\text{redundancy reduction term}}$$

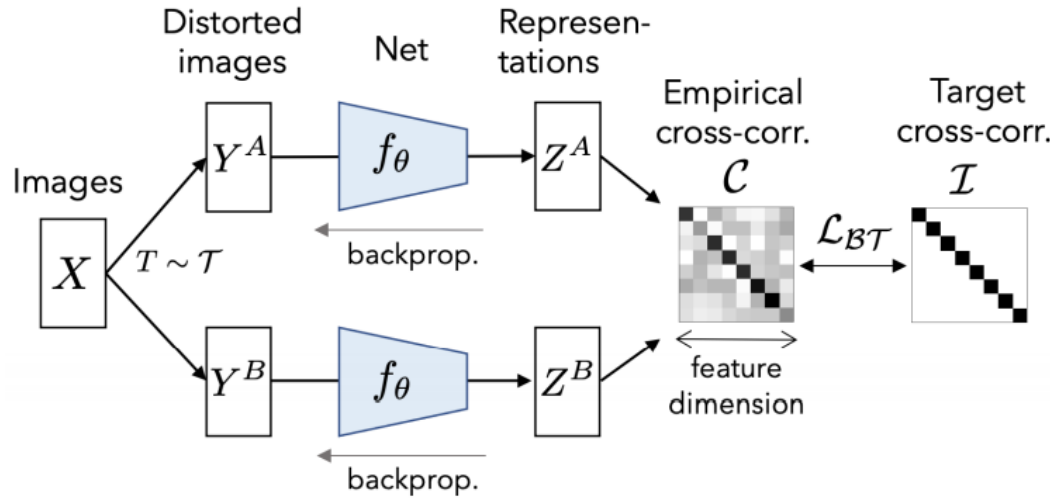


The representation vectors of distorted versions of a sample to be similar, while minimizing the redundancy between the components of these vectors.

Method

Algorithm 1 PyTorch-style pseudocode for Barlow Twins.

```
# f: encoder network
# lambda: weight on the off-diagonal terms
# N: batch size
# D: dimensionality of the representation
#
# mm: matrix-matrix multiplication
# off_diagonal: off-diagonal elements of a matrix
# eye: identity matrix

for x in loader: # load a batch with N samples
    # two randomly augmented versions of x
    y_a, y_b = augment(x)

    # compute representations
    z_a = f(y_a) # NxD
    z_b = f(y_b) # NxD

    # normalize repr. along the batch dimension
    z_a_norm = (z_a - z_a.mean(0)) / z_a.std(0) # NxD
    z_b_norm = (z_b - z_b.mean(0)) / z_b.std(0) # NxD

    # cross-correlation matrix
    c = mm(z_a_norm.T, z_b_norm) / N # DxD

    # loss
    c_diff = (c - eye(D)).pow(2) # DxD
    # multiply off-diagonal elems of c_diff by lambda
    off_diagonal(c_diff).mul_(lambda)
    loss = c_diff.sum()

    # optimization step
    loss.backward()
    optimizer.step()
```



$$\mathcal{L}_{\mathcal{BT}} \triangleq \underbrace{\sum_i (1 - \mathcal{C}_{ii})^2}_{\text{invariance term}} + \lambda \underbrace{\sum_i \sum_{j \neq i} \mathcal{C}_{ij}^2}_{\text{redundancy reduction term}}$$

$$\mathcal{C}_{ij} \triangleq \frac{\sum_b z^A_{b,i} z^B_{b,j}}{\sqrt{\sum_b \left(z^A_{b,i}\right)^2} \sqrt{\sum_b \left(z^B_{b,j}\right)^2}}$$

Results

Self-supervised learning / Semi-supervised learning

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.

Method	Top-1	Top-5
Supervised	76.5	
MoCo	60.6	
PIRL	63.6	-
SimCLR	69.3	89.0
MoCo v2	71.1	90.1
SimSiam	71.3	-
SwAV (w/o multi-crop)	71.8	-
BYOL	74.3	91.6
SwAV	75.3	-
Barlow Twins (ours)	73.2	91.0

Table 2. **Semi-supervised learning on ImageNet** using 1% and 10% training examples. Results for the supervised method are from (Zhai et al., 2019). Best results are in **bold**.

Method	Top-1		Top-5	
	1%	10%	1%	10%
Supervised	25.4	56.4	48.4	80.4
PIRL	-	-	57.2	83.8
SimCLR	48.3	65.6	75.5	87.8
BYOL	53.2	68.8	78.4	89.0
SwAV	53.9	**70.2**	78.5	**89.9**
Barlow Twins (ours)	**55.0**	69.7	**79.2**	89.3

Results

Transfer learning

Table 3. **Transfer learning: image classification.** We benchmark learned representations on the image classification task by training linear classifiers on fixed features. We report top-1 accuracy on Places-205 and iNat18 datasets, and classification mAP on VOC07. Top-3 best self-supervised methods are underlined.

Method	Places-205	VOC07	iNat18
Supervised	53.2	87.5	46.7
SimCLR	52.5	85.5	37.2
MoCo-v2	51.8	<u>86.4</u>	38.6
SwAV (w/o multi-crop)	52.8	<u>86.4</u>	39.5
SwAV	<u>56.7</u>	<u>88.9</u>	<u>48.6</u>
BYOL	<u>54.0</u>	<u>86.6</u>	<u>47.6</u>
BARLOW TWINS (ours)	<u>54.1</u>	86.2	<u>46.5</u>

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× schedule. Best results are in **bold**.

Method	VOC07+12 det			COCO det			COCO instance seg		
	AP_{all}	AP_{50}	AP_{75}	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
Sup.	53.5	81.3	58.8	38.2	58.2	41.2	33.3	54.7	35.2
MoCo-v2	**57.4**	82.5	**64.0**	**39.3**	58.9	**42.5**	**34.4**	55.8	36.5
SwAV	56.1	**82.6**	62.7	38.4	58.6	41.3	33.8	55.2	35.9
SimSiam	57	82.4	63.7	39.2	**59.3**	42.1	**34.4**	**56.0**	**36.7**
BT (ours)	56.8	**82.6**	63.4	39.2	59.0	**42.5**	34.3	**56.0**	36.5

Results

Ablation study

Loss function

Table 5. **Loss function explorations**. We ablate the invariance and redundancy terms in our proposed loss and observe that both terms are necessary for good performance. We also experiment with different normalization schemes and a cross-entropy loss and observe reduced performance.

Loss function	Top-1	Top-5
Baseline	71.4	90.2
Only invariance term (on-diag term)	57.3	80.5
Only red. red. term (off-diag term)	0.1	0.5
Normalization along feature dim.	69.8	88.8
No BN in MLP	71.2	89.7
No BN in MLP + no Normalization	53.4	76.7
Cross-entropy with temp.	63.3	85.7

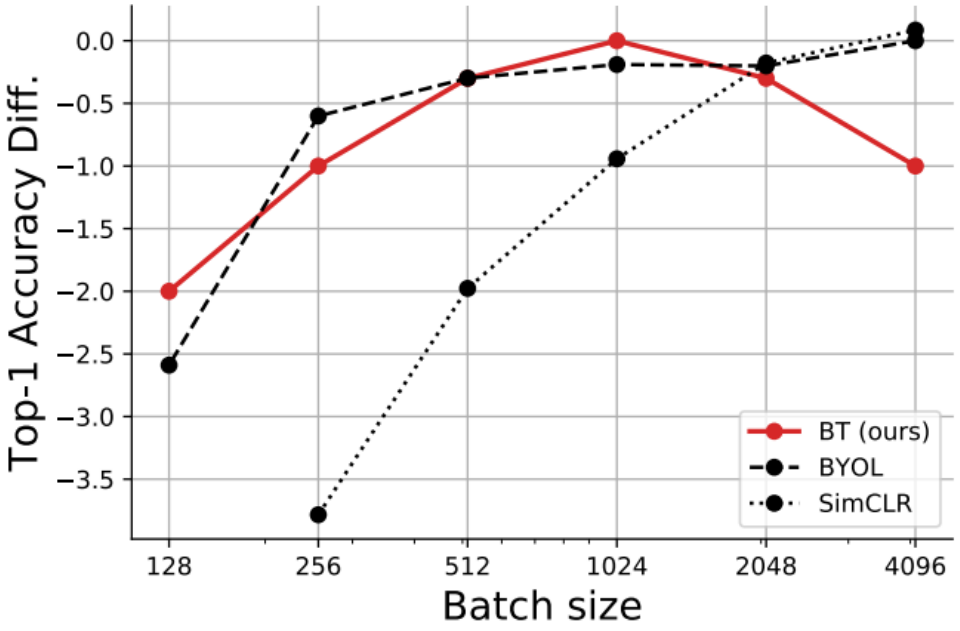
$$\mathcal{L}_{\mathcal{BT}} \triangleq \underbrace{\sum_i (1 - \mathcal{C}_{ii})^2}_{\text{invariance term}} + \lambda \underbrace{\sum_i \sum_{j \neq i} \mathcal{C}_{ij}{}^2}_{\text{redundancy reduction term}}$$

Cross-entropy with temp. $= -\log \sum_i \exp(C_{ii}/\tau)$
$+\lambda \sum_i \sum_{j \neq i} \exp(\max(C_{ij}, 0)/\tau)$

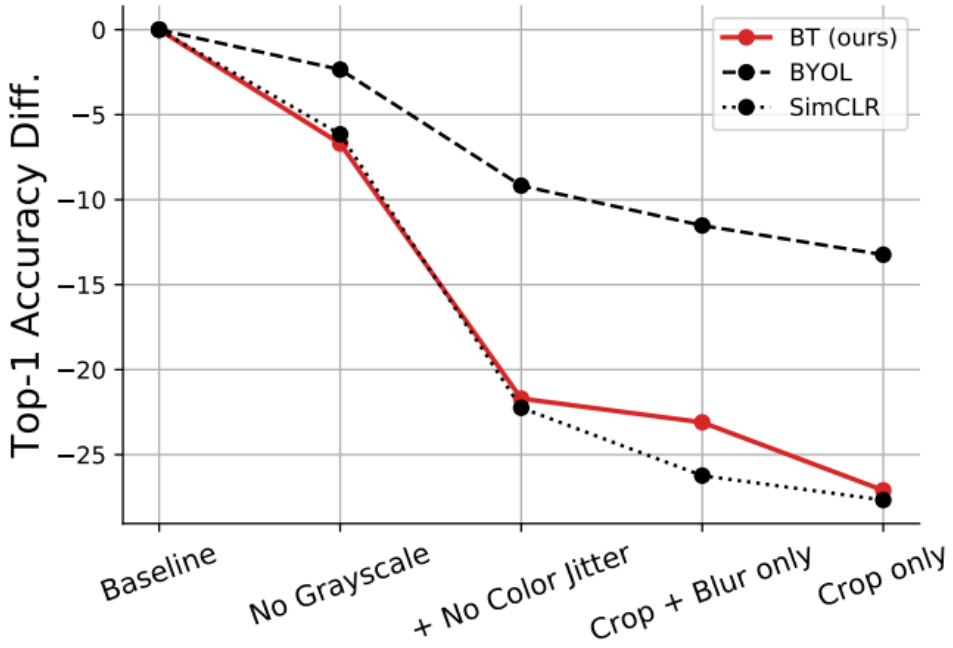
Results

Ablation study

Batch size



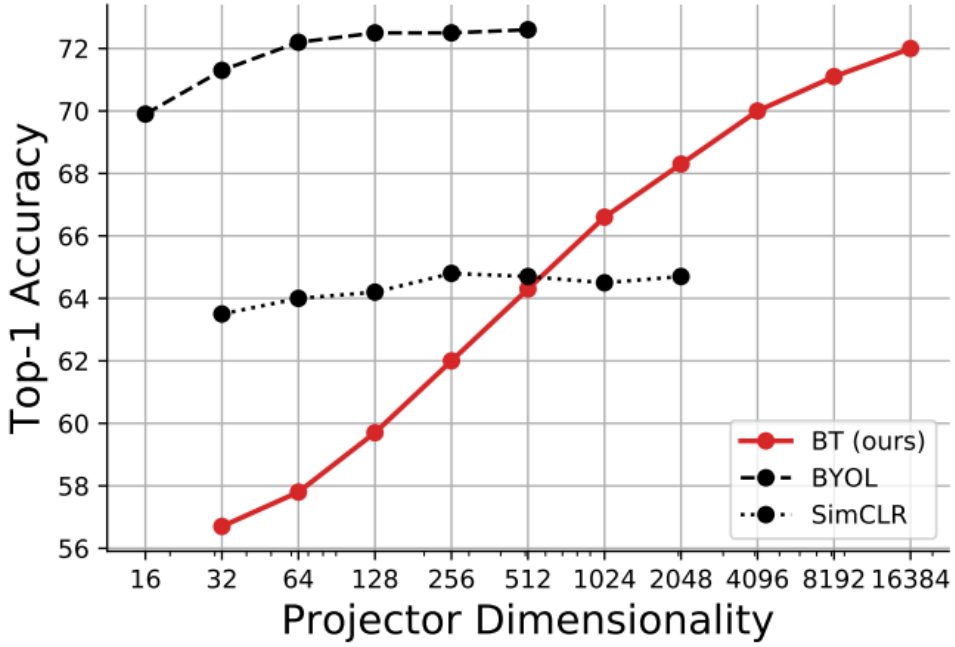
Removing Augmentations



Results

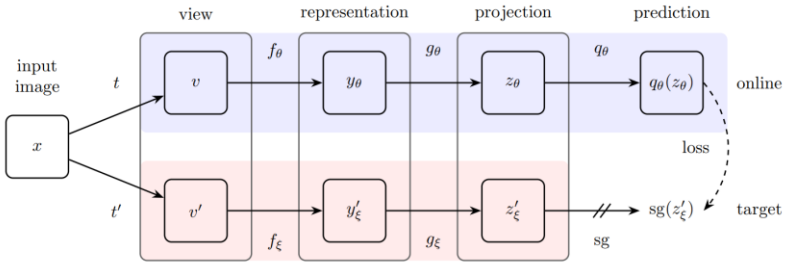
Ablation study

Projector dimensionality

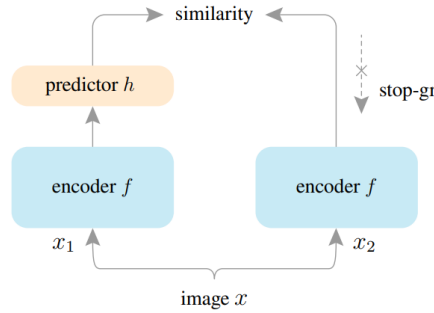


Asymmetric settings

case	stop-gradient	predictor	Top-1	Top-5
Baseline	-	-	71.4	90.2
(a)	✓	-	70.5	89.0
(b)	-	✓	70.2	89.0
(c)	✓	✓	61.3	83.5



BYOL



SimSiam