MADMO2

Self-Supervised overview

Тарас Хахулин
Skoltech, MIPT
Deep Learning Engineer, Samsung AI Center
Tg: @vitaminotar
https://github.com/khakhulin/
https://twitter.com/t_khakhulin
https://www.linkedin.com/in/taras-khakhulin/

Different types of learning

- Supervised Learning
- Semi-Supervised Learning
- Unsupervised Learning
- Self-Supervised

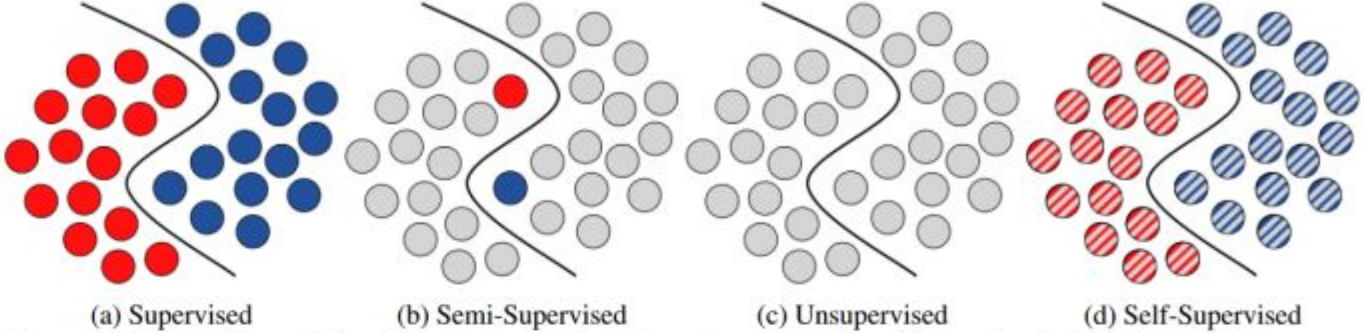
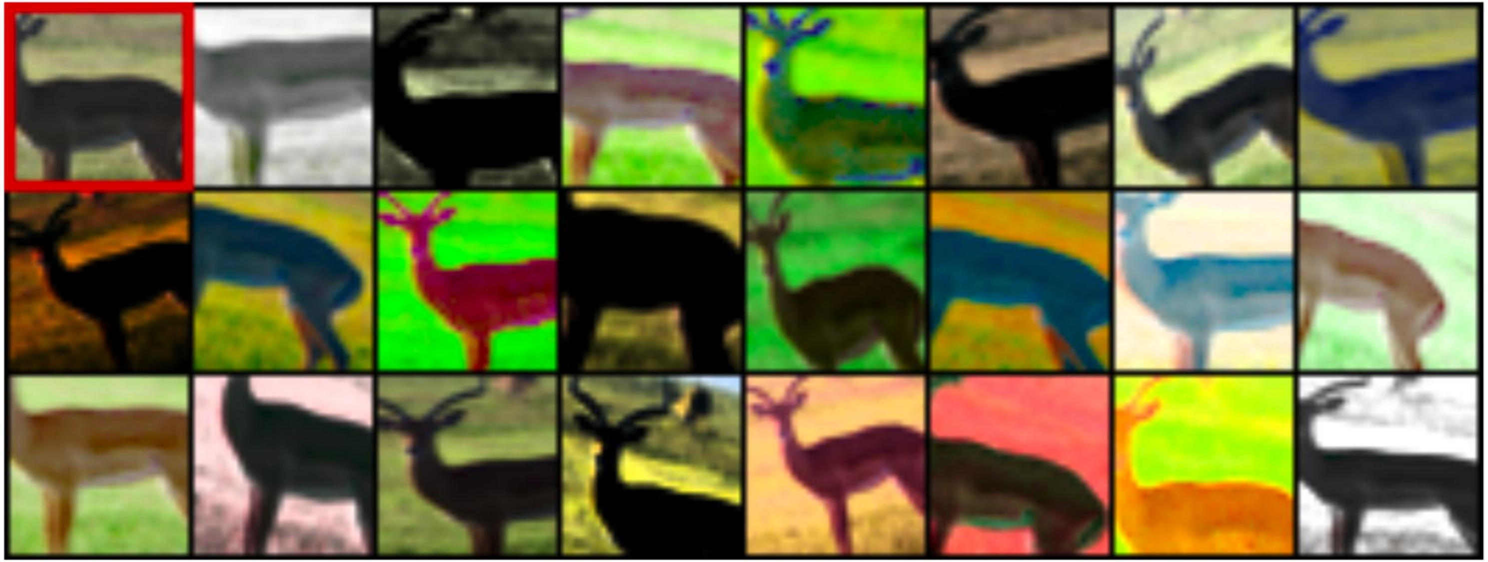


Figure 2: Illustrations of the four presented deep learning strategies - The red and dark blue circles represent labeled data points of different classes. The light grey circles represent unlabeled data points. The black lines define the underlying decision boundary between the classes. The striped circles represent datapoints which ignore and use the label information at different stages of the training process.

Img https://arxiv.org/abs/2002.08721

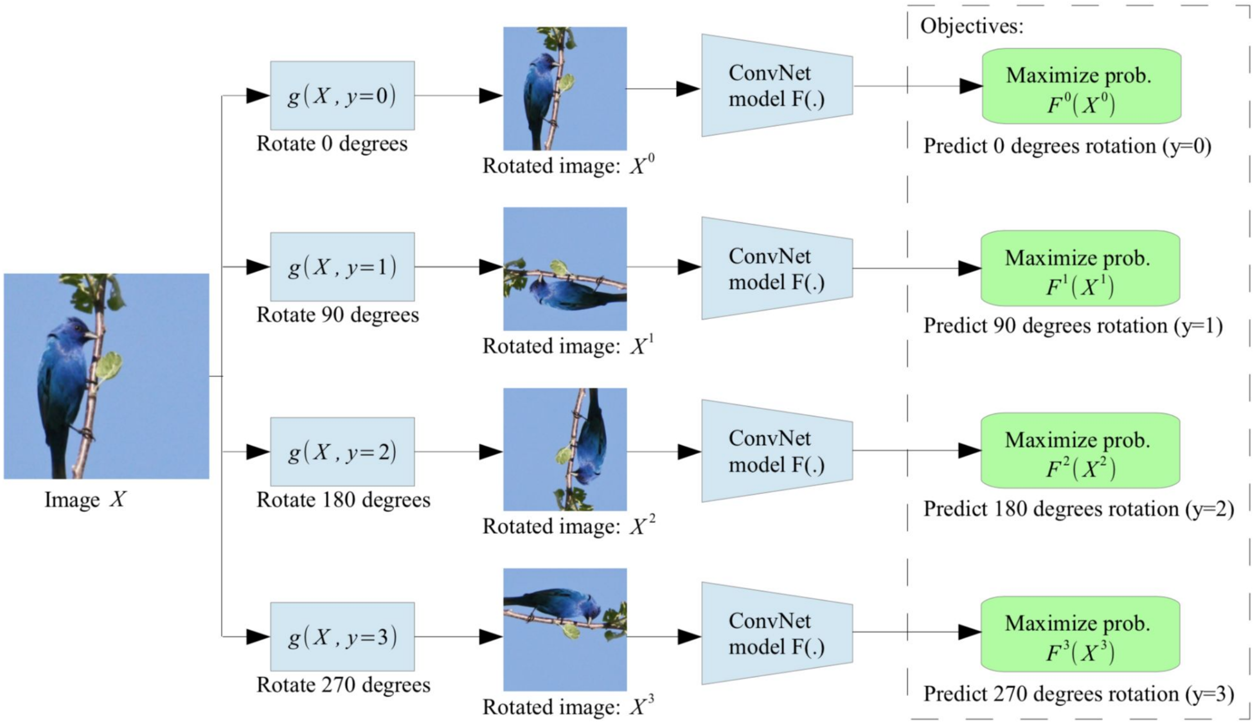
But how?

Exemplar based



Dosovitskiy et al., 2015

Image rotation



Gidaris et al. 2018

Patch order: relative position

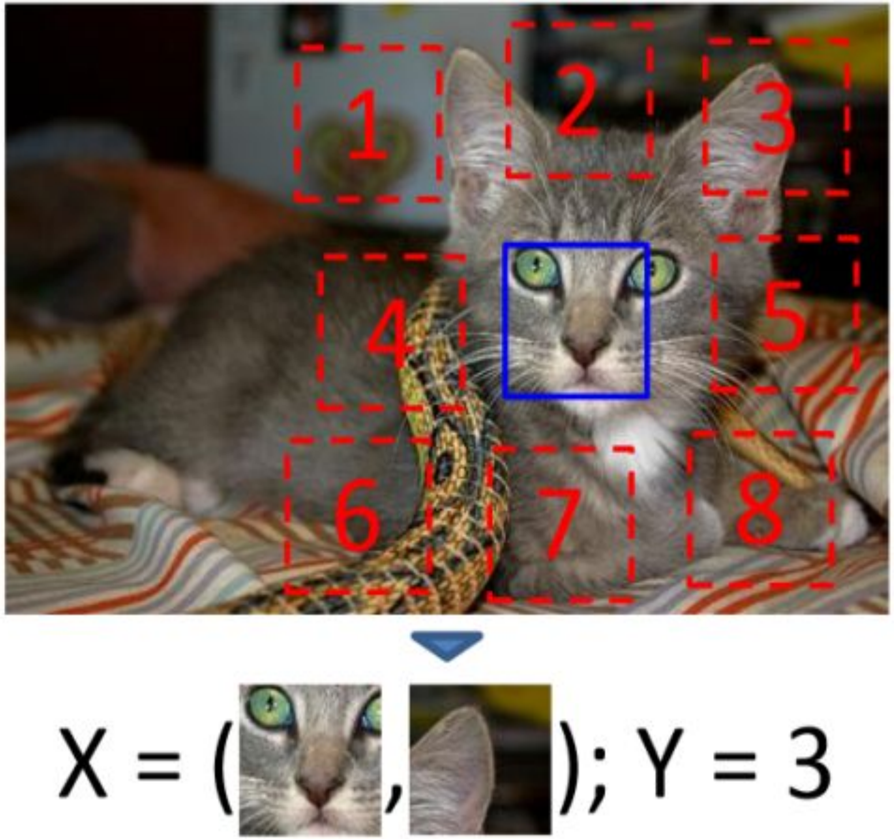
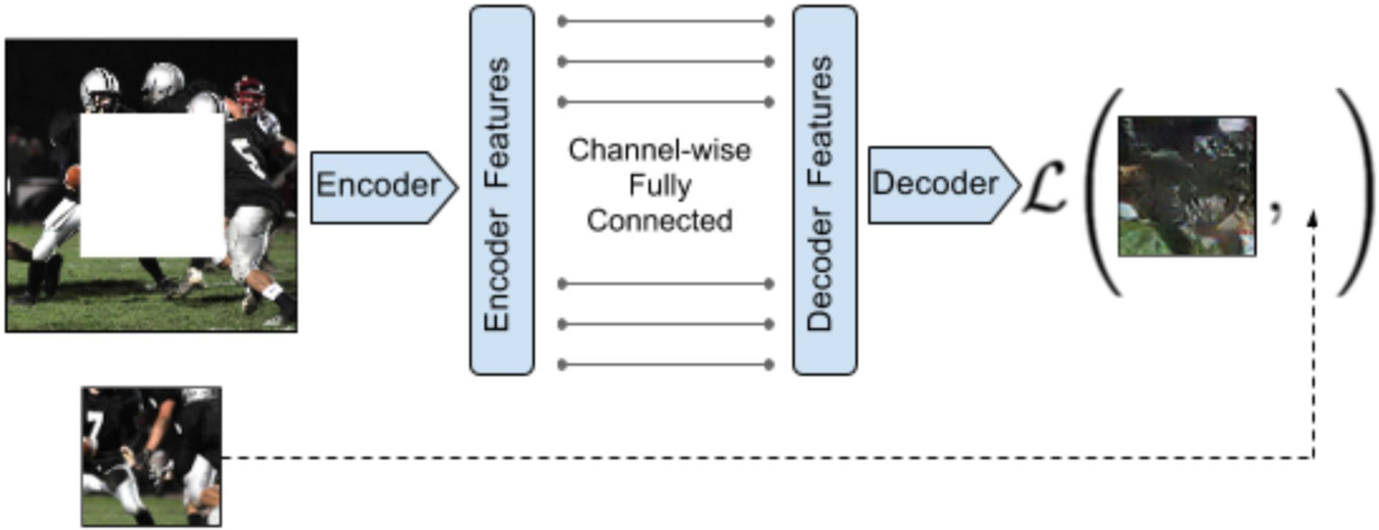


Figure 2. The algorithm receives two patches in one of these eight possible spatial arrangements, without any context, and must then classify which configuration was sampled.

Doersch et al. (2015)

Context Encoder

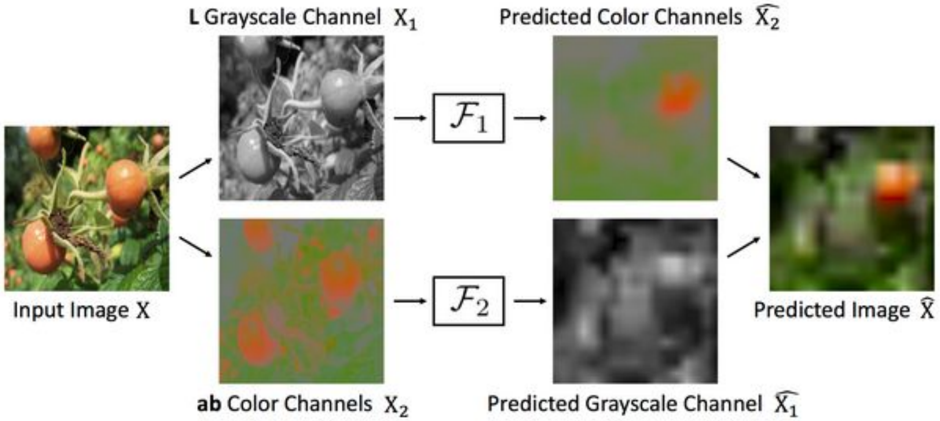


Pathak, et al., 2016

Colorization

DeOldify

Colorization

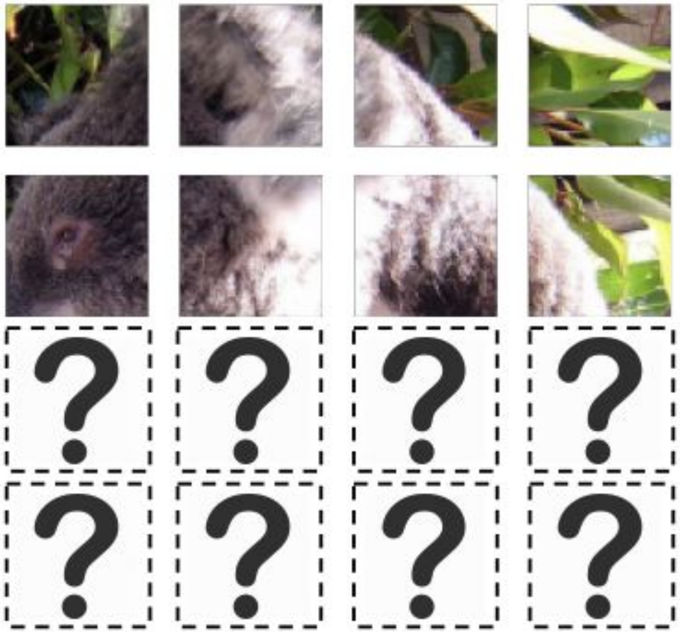


DeOldify

[Zheng et al.](#)

Contrastive Learning

Contrastive Prediction Coding

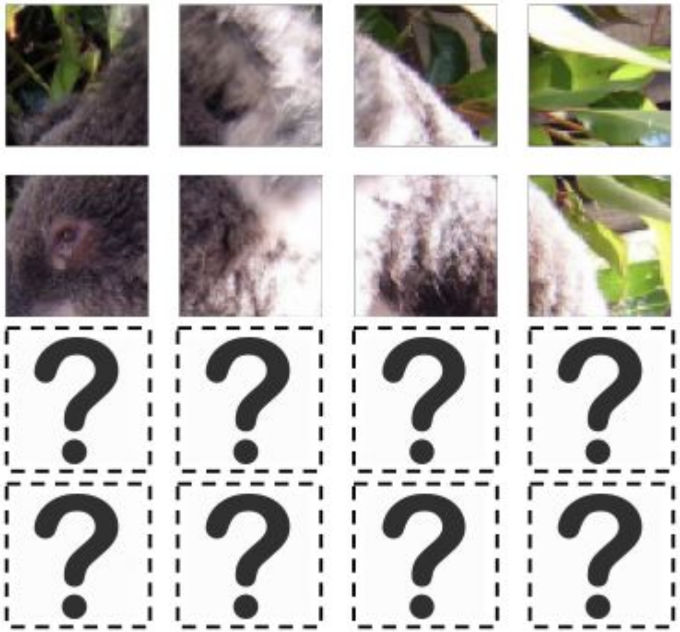


Predict “future” patches

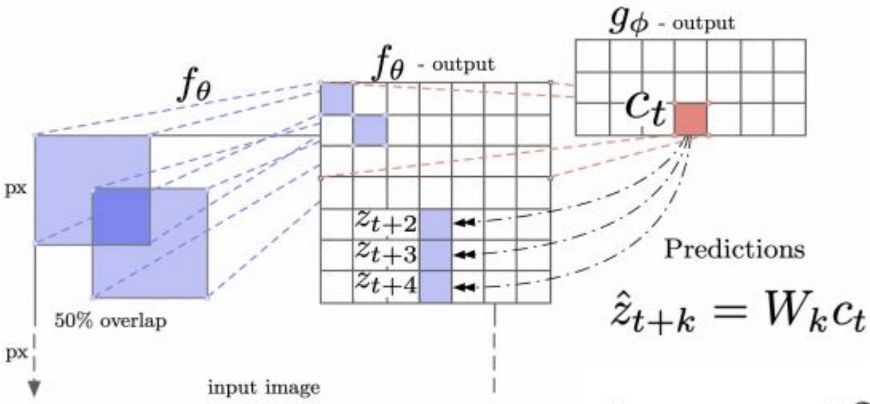
MSE doesn’t solve our problems

https://arxiv.org/abs/1905.09272

Contrastive Prediction Coding



Predict “future” patches



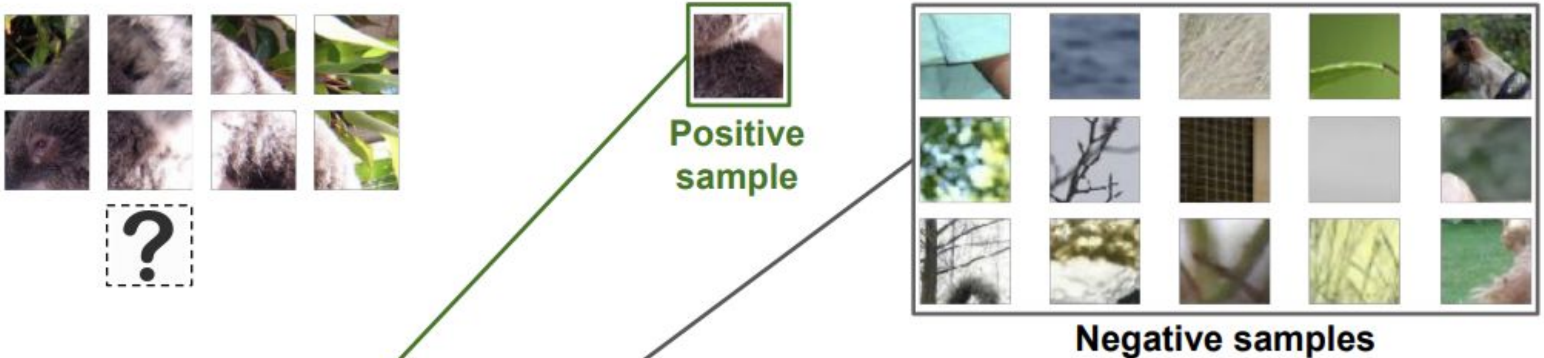
$$\hat{z}_{t+k} = W_k c_t$$

$$\|\hat{z}_t - z_t\|^2$$

https://arxiv.org/abs/1905.09272

Contrastive loss

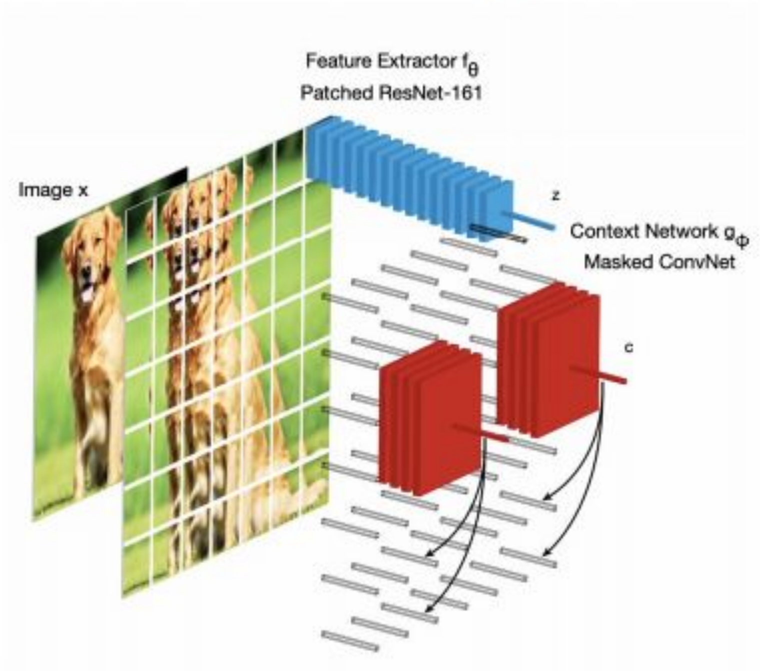
- MSE loss $\|\hat{z}_t - z_t\|^2$ would lead to a constant prediction.
- Generative models are hard to train.
- Instead, use Noise Contrastive Estimation loss:

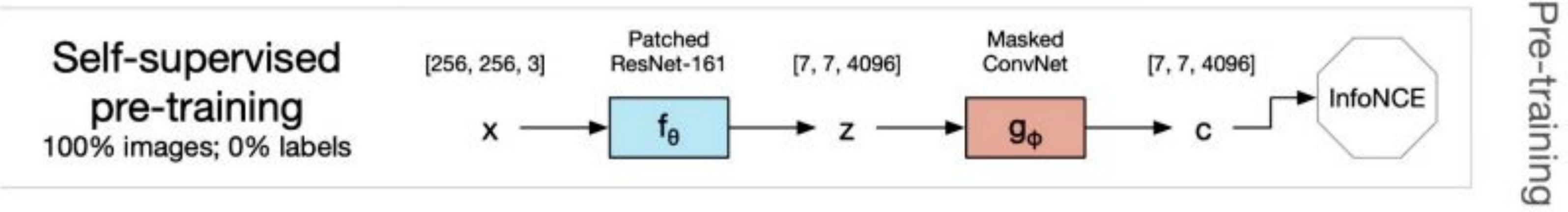


$$\log p(z_t \mid \hat{z}_t, \{z_l\}) = \log\left(\frac{\exp(\hat{z}_t^T z_t)}{\exp(\hat{z}_t^T z_t) + \sum_l \exp(\hat{z}_t^T z_l)}\right) \rightarrow \max$$

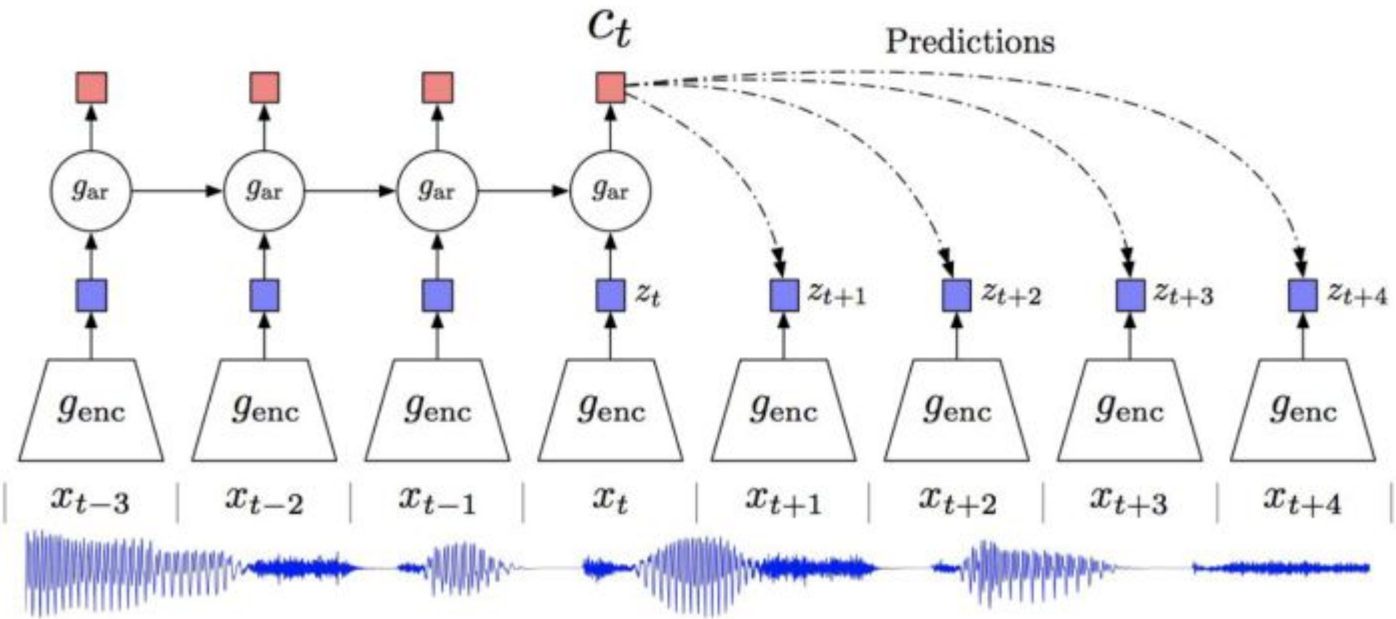
Predicted vector

Contrastive Prediction Coding





Original Prediction Coding



Paper [Oord]

Evaluation

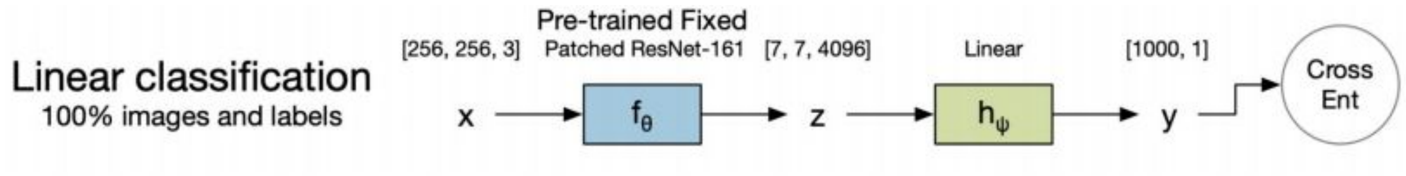


Table 1: Linear classification accuracy, and comparison to other self-supervised methods. In all cases the feature extractor is optimized in an unsupervised manner (using one of the methods listed below), and a linear classifier is trained on top using all labels in the ImageNet dataset.

Method	**Architecture**	**Parameters (M)**	**Top-1**	**Top-5**
Methods using ResNet-50:				
Local Aggregation [66]	ResNet-50	24	60.2	-
Momentum Contrast [25]	ResNet-50	24	60.6	-
CPC v2	ResNet-50	24	**63.8**	**85.3**
Methods using different architectures:				
Multi-task [13]	ResNet-101	28	-	69.3
Rotation [32]	RevNet-50 ×4	86	55.4	-
CPC v1 [58]	ResNet-101	28	48.7	73.6
BigBiGAN [15]	RevNet-50 ×4	86	61.3	81.9
AMDIM [5]	Custom-103	626	68.1	-
CMC [57]	ResNet-50 ×2	188	68.4	88.2
Momentum Contrast [25]	ResNet-50 ×4	375	68.6	-
CPC v2	ResNet-161	305	**71.5**	**90.1**

CPC Loss

$$-\mathbf{E}_X \left[\log \frac{\exp(f(x)^{\mathrm{T}} f(x^+))}{\exp(f(x)^{\mathrm{T}} f(x^+)) + \sum_{j=1}^{N-1} \exp(f(x)^{\mathrm{T}} f(x_j))} \right]$$

x – выбранный патч
x+ – похожий на него,
xj – непохожий
f( ) – кодировщик объекта (представление его в виде вещественного вектора).

CPC Loss

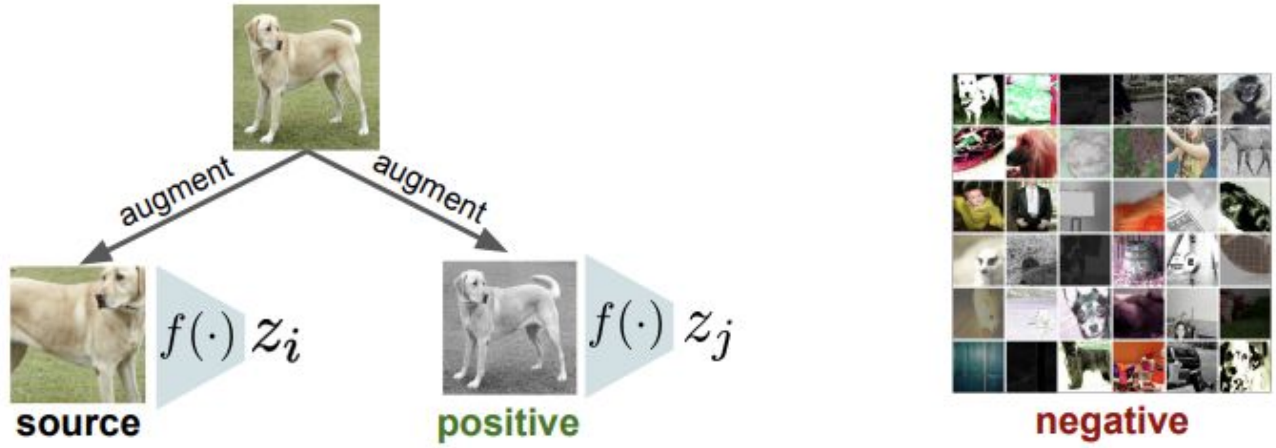
$$L_t = -\log\left(\frac{\exp(\hat{z}_t^T z_t)}{\exp(\hat{z}_t^T z_t) + \sum_l \exp(\hat{z}_t^T z_l)}\right)$$

source **positive** **negative**

- **source** --- predicted embedding for a patch
- **positive** --- true embedding
- **negative** --- embeddings of random patches

Other (simple) ways to sample source, positive and negatives?

A Simple Framework for Contrastive Learning (SimCLR)



$$L_{i,j} = -\log\left(\frac{\exp(z_i^T z_j)}{\exp(z_i^T z_j) + \sum_l \exp(z_i^T z_l)}\right)$$

[SimCLR]

SimCLR

- Given an image find its augmentation among other images.
- Use the rest of batch as negative examples instead of a memory bank (need large batches, N=2-4K)

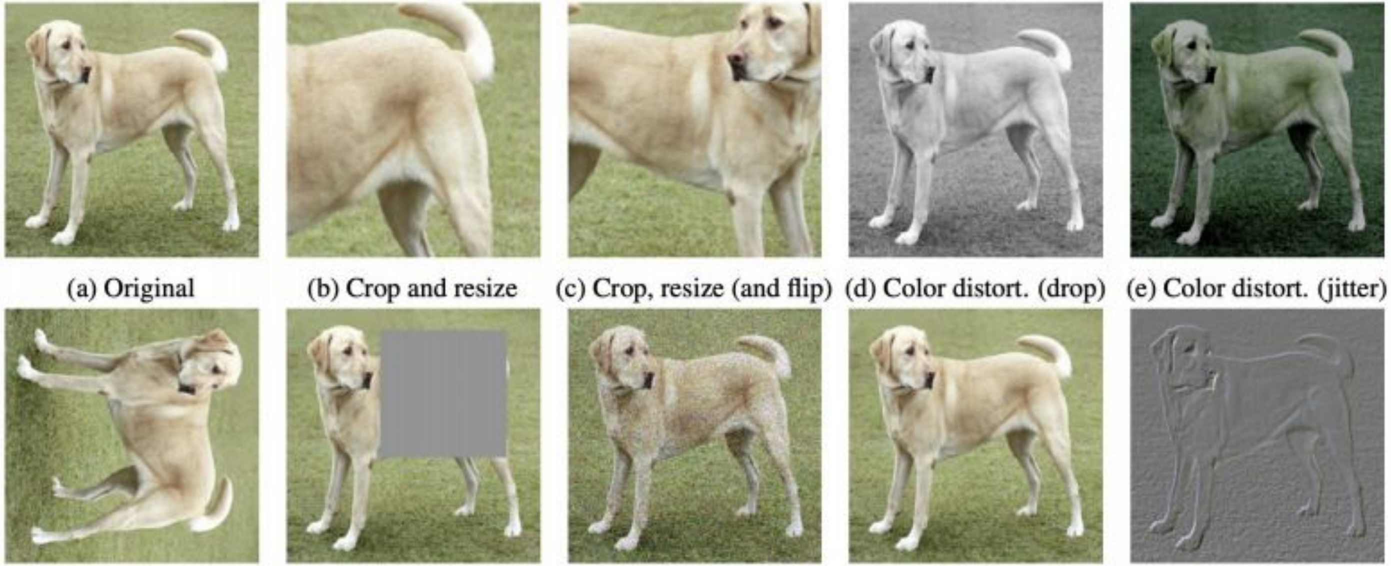
Batch of size N:

2N loss terms:

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

[SimCLR]

Data augmentation strategy



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

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

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)

Method	Architecture	Param.	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**

Table 6. ImageNet accuracies of linear classifiers trained on representations learned with different self-supervised methods.

SimCLR

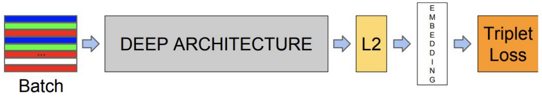


Figure 2. **Model structure.** Our network consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training.

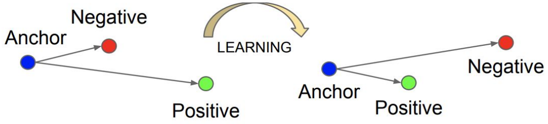


Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.

Additional Metric Learning

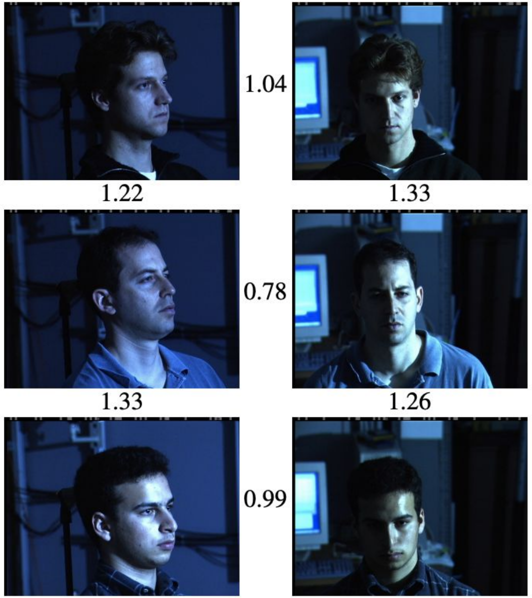


Figure 1. **Illumination and Pose invariance.** Pose and illumination have been a long standing problem in face recognition. This figure shows the output distances of FaceNet between pairs of faces of the same and a different person in different pose and illumination combinations. A distance of 0.0 means the faces are identical, 4.0 corresponds to the opposite spectrum, two different identities. You can see that a threshold of 1.1 would classify every pair correctly.

FaceNet, 15

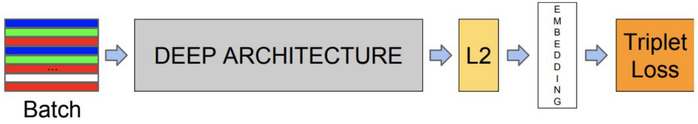


Figure 2. **Model structure.** Our network consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training.

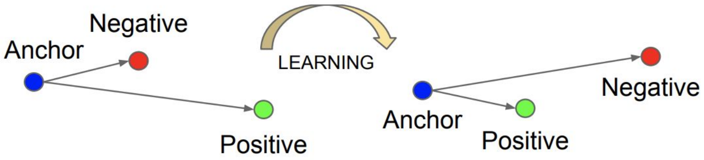


Figure 3. The **Triplet Loss** minimizes the distance between an *anchor* and a *positive*, both of which have the same identity, and maximizes the distance between the *anchor* and a *negative* of a different identity.