Unsupervise\self-supervised learning

Self-supervised

Есть много неразмеченных данных

Задача: научить хорошие векторные представления об изображениях на неразмеченных данных

Генеративный подход

DC gan

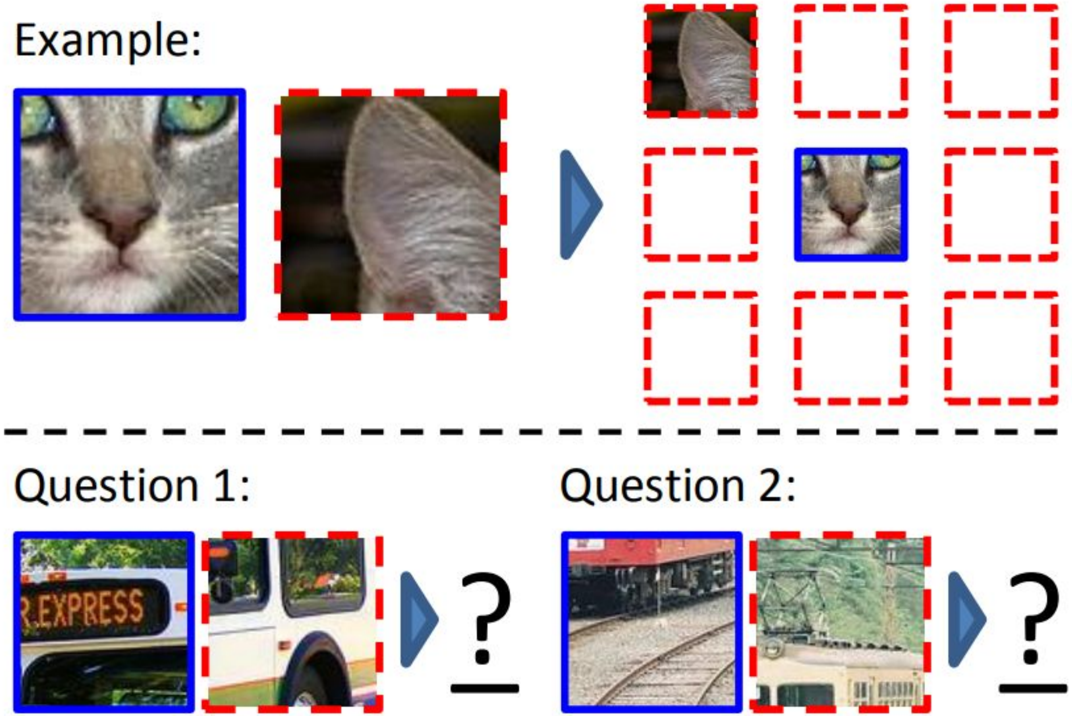
используем выходы из дискриминатора

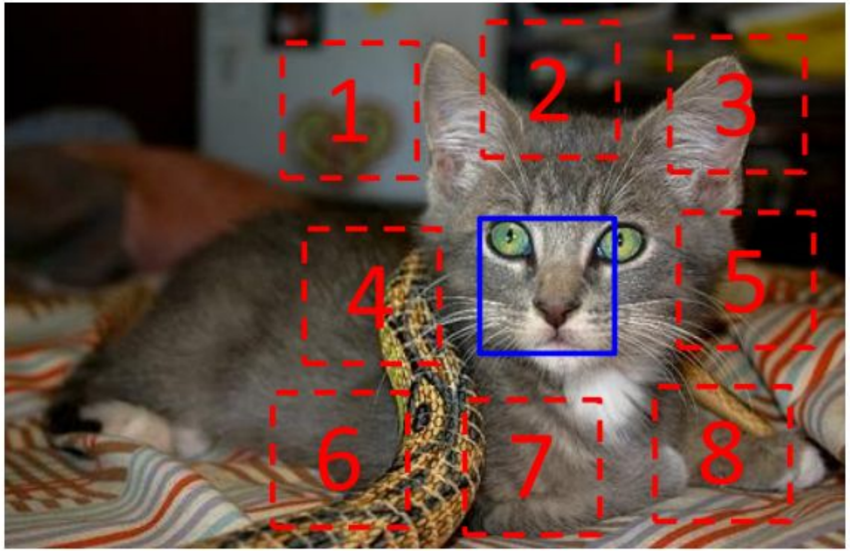
как семплируем фичи или DC gan?

Дискриминативный подходы

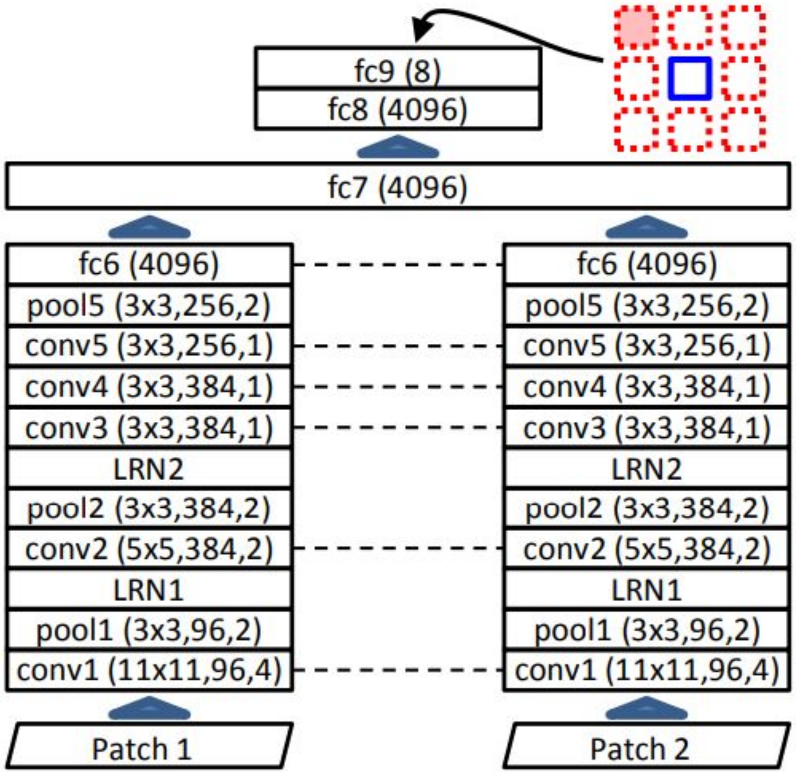
учим предсказывать контекст для изображения

https://arxiv.org/pdf/1505.05192.pdf


Example:
Question 1:
R.EXPRESS
?
Question 2:
?



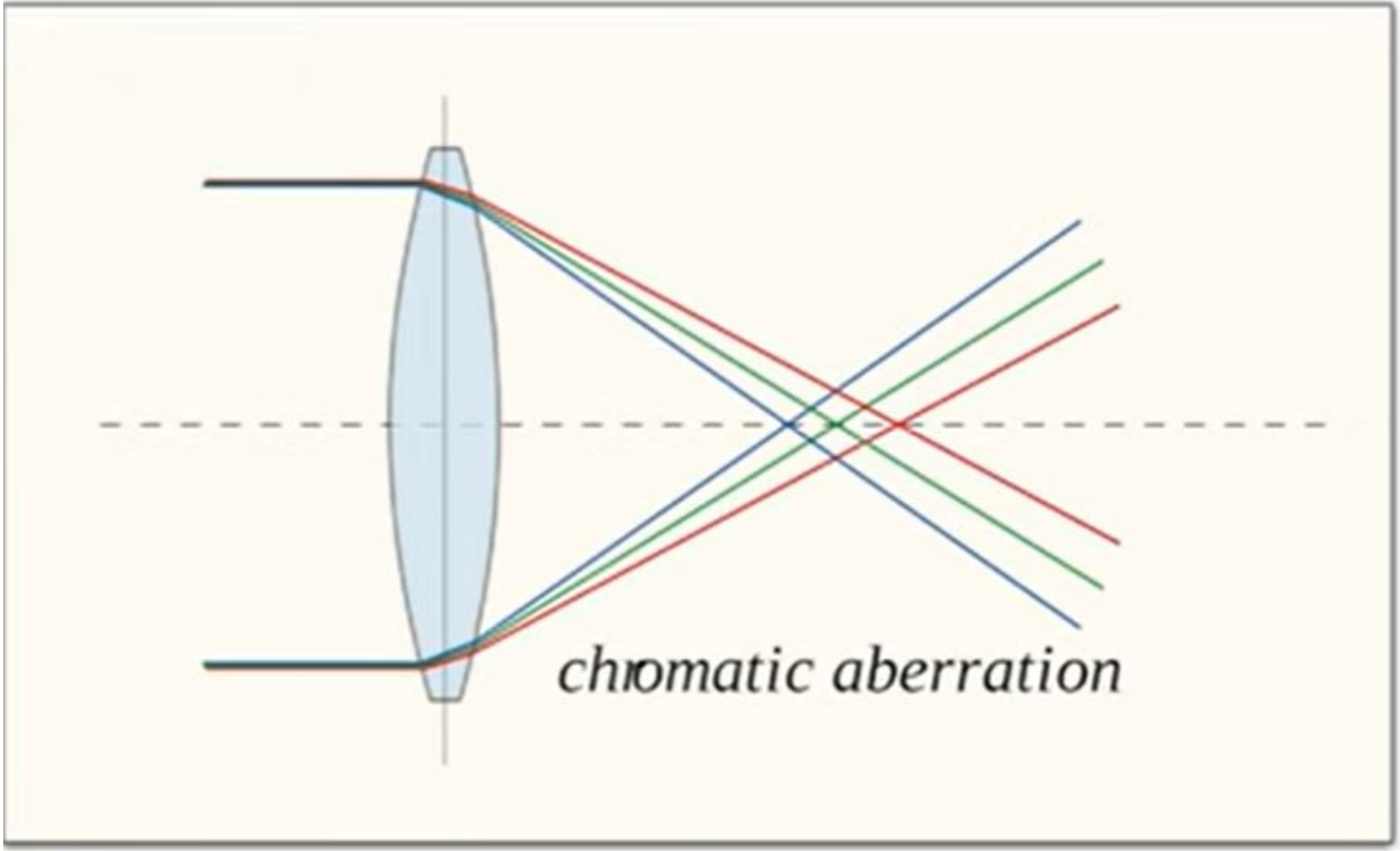
X = (, ); Y = 3

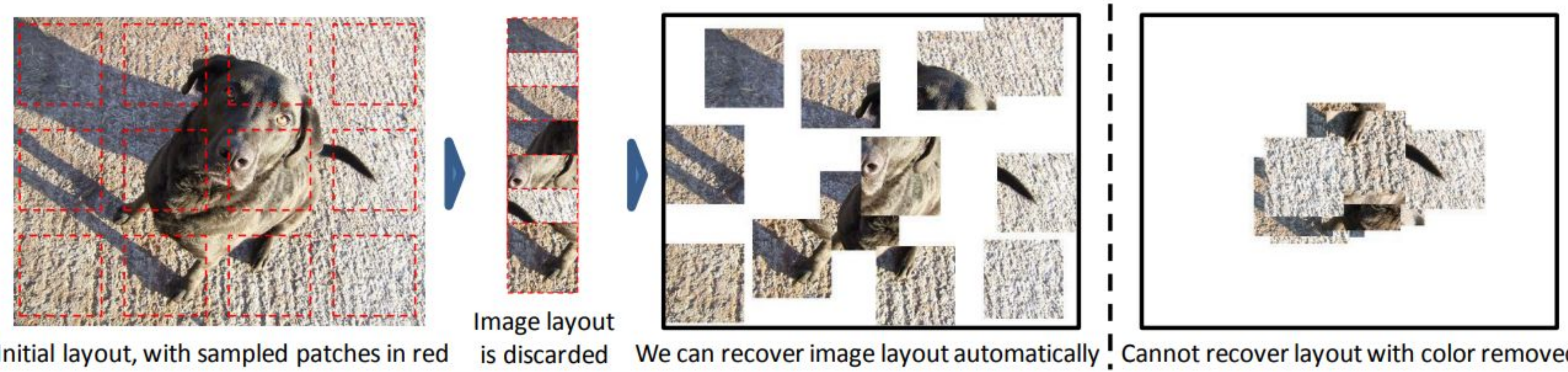


fc9 (8)
fc8 (4096)
fc7 (4096)
fc6 (4096)
pool5 (3x3,256,2)
conv5 (3x3,256,1)
conv4 (3x3,384,1)
conv3 (3x3,384,1)
LRN2
pool2 (3x3,384,2)
conv2 (5x5,384,2)
LRN1
pool1 (3x3,96,2)
conv1 (11x11,96,4)
Patch 1
fc6 (4096)
pool5 (3x3,256,2)
conv5 (3x3,256,1)
conv4 (3x3,384,1)
conv3 (3x3,384,1)
LRN2
pool2 (3x3,384,2)
conv2 (5x5,384,2)
LRN1
pool1 (3x3,96,2)
conv1 (11x11,96,4)
Patch 2

сеть легко запоминает тривиальное решение задачи

- задаем расстояние между патчами
- сдвиг центра патчи
- color projection или dropping


chromatic aberration

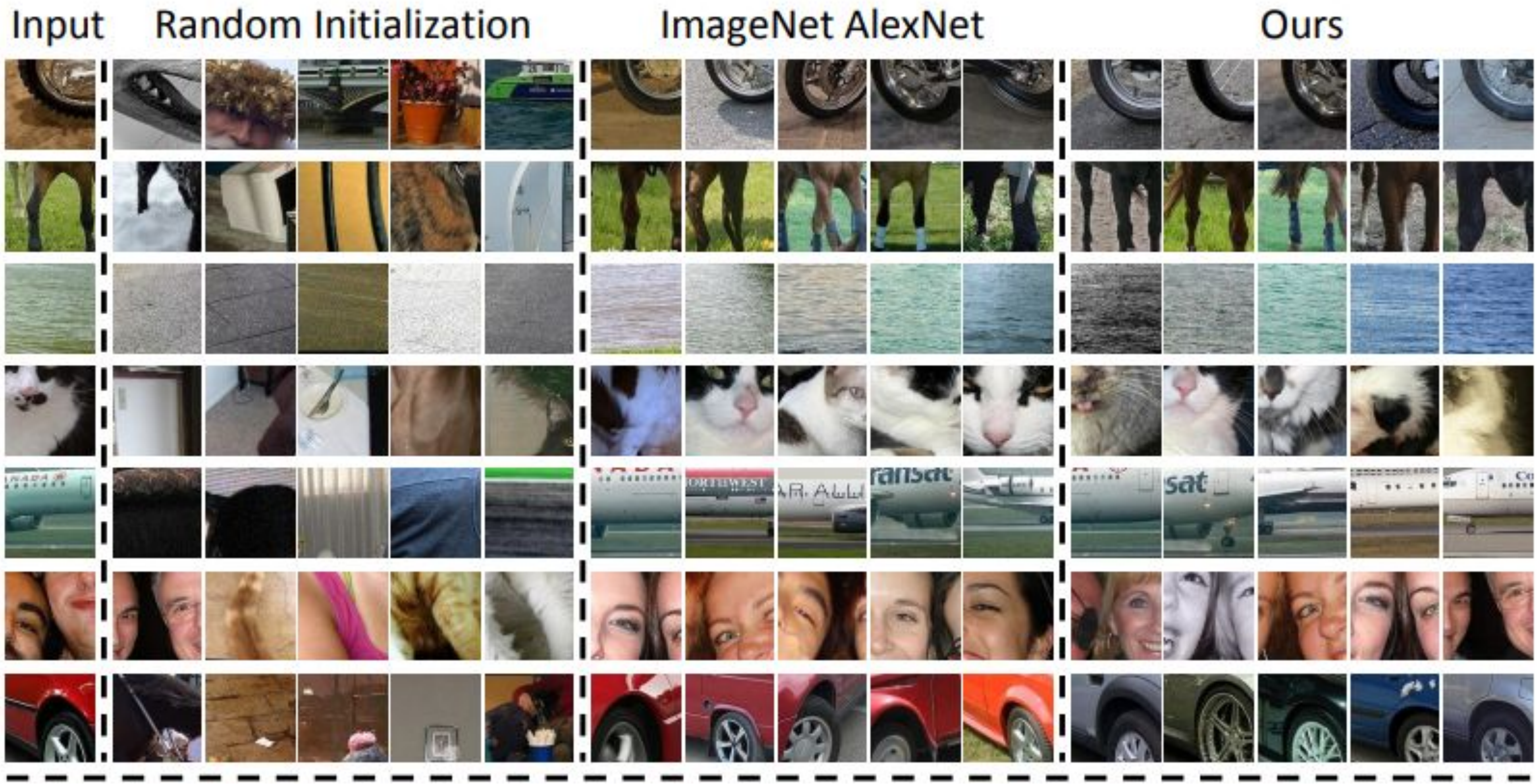


Initial layout, with sampled patches in red

Image layout is discarded

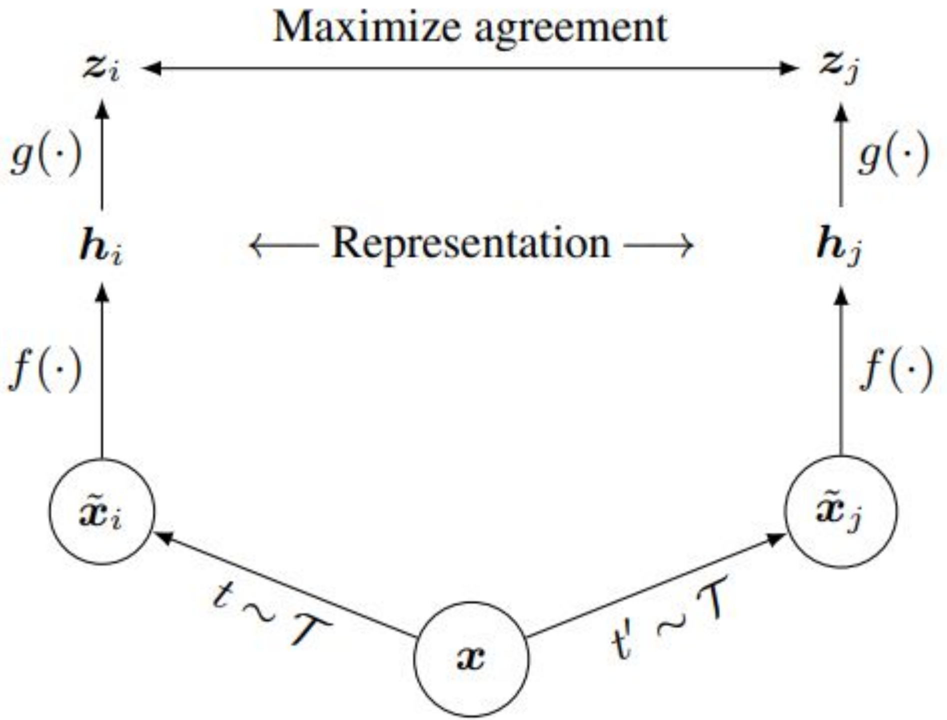
We can recover image layout automatically

Cannot recover layout with color removed

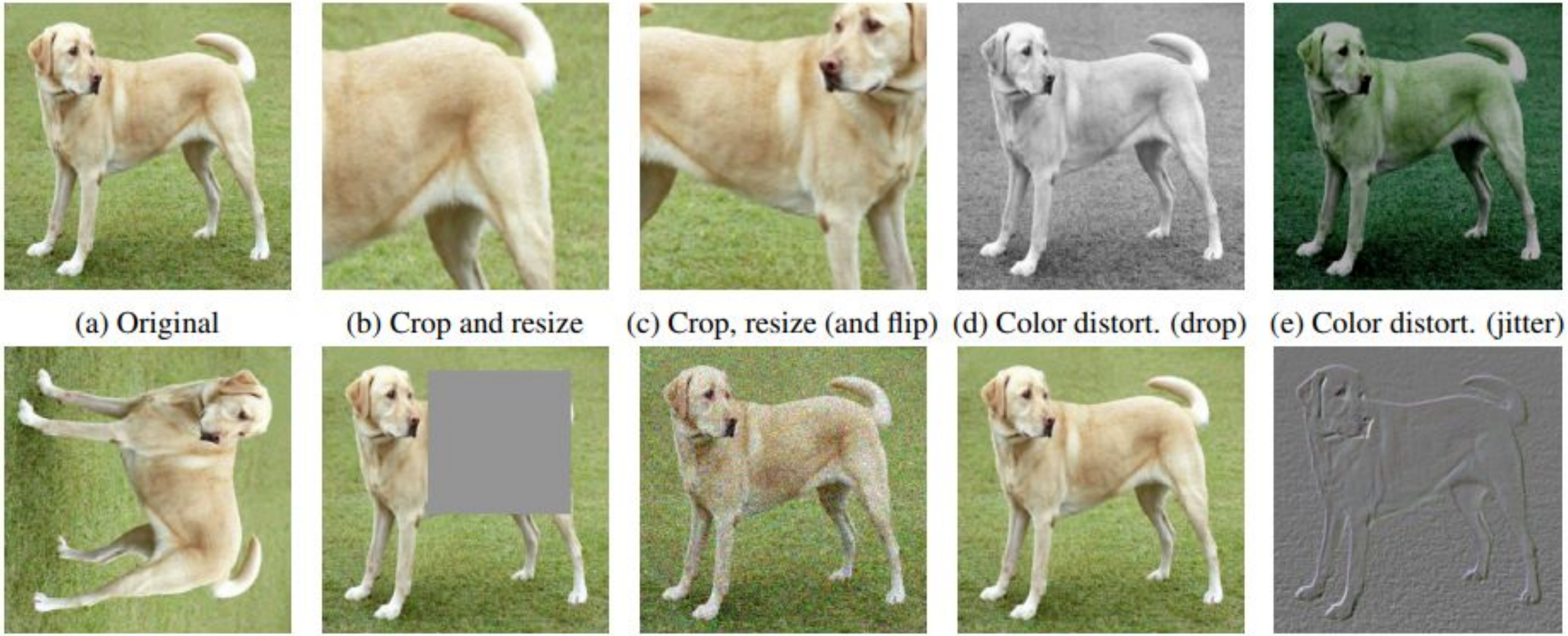

Input
Random Initialization
ImageNet AlexNet
Ours

SimCLR

$$z_i = g(h_i) = W^{(2)}\sigma(W^{(1)}h_i)$$



SimCLR: аугментации



(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)

SimCLR: обучение

A *contrastive loss function* defined for a contrastive prediction task. Given a set $\{\tilde{\boldsymbol{x}}_k\}$ including a positive pair of examples $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_j$, the *contrastive prediction task* aims to identify $\tilde{\boldsymbol{x}}_j$ in $\{\tilde{\boldsymbol{x}}_k\}_{k\neq i}$ for a given $\tilde{\boldsymbol{x}}_i$.

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

$$\mathrm{sim}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^\top \boldsymbol{v}/\|\boldsymbol{u}\|\|\boldsymbol{v}\|$$

Algorithm 1 SimCLR's main learning algorithm.

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

SimCLR: результаты

Method	Architecture	Label fraction 1%	Label fraction 10%
		Top 5	
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**

Table 7. ImageNet accuracy of models trained with few labels.

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**

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