An Introduction to Self-Supervised Learning

Yuan YAO

HKUST

Supervised Learning

- **Data**: (x, y)
  x is input, y is output/response (label)
- **Goal**: Learn a *function* to map x -> y
- **Examples**:
  - Classification,
  - regression,
  - object detection,
  - semantic segmentation,
  - image captioning, etc.

Cat

Unsupervised Learning

- **Data**: x
  Just input data, no output labels!
- **Goal**: Learn some underlying hidden *structure* of the data
- **Examples**:
  - Clustering,
  - dimensionality reduction (manifold learning),
  - Density (probability) estimation,
  - Generative models:
    - Autoencoder
    - GANs, etc.

Generative Models

Given training data, generate new samples from same distribution

Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

Want to learn $p_{model}(x)$ similar to $p_{data}(x)$

Today: Self-Supervised Learning

- **Data**: x
  - Just input data, no output labels!
  - Data labeling is expensive and thus high-quality labeled dataset is limited.
- Both **Self-Supervised Learning (SSL)** and Generative model learn good data representation from unlabelled dataset.
- Generative learning aims to model **data distribution** $p_{data}(x)$, or **generating** data.
- Self-supervised learning methods solve "***pretext***" tasks that produce **good features** for downstream tasks:
  - **Pretext task: (x_{-i}, x_i) or $(x, T(x))$**
  - Learning good representation makes it easier to transfer useful information to a variety of downstream tasks.
    - e.g. A downstream task has only a few examples.
    - e.g. Zero-shot transfer to new tasks.

Self-supervised pretext tasks

Example: learn to predict image transformations / complete corrupted images

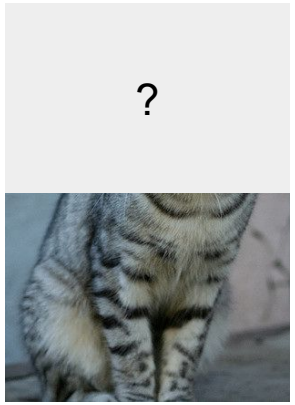
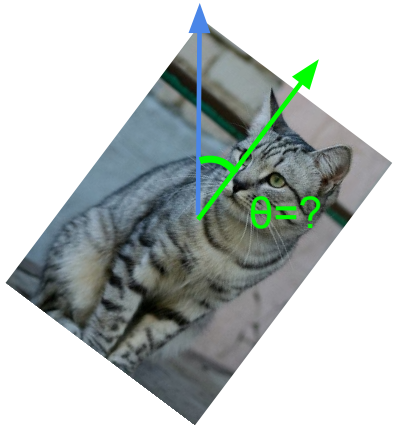


image completion



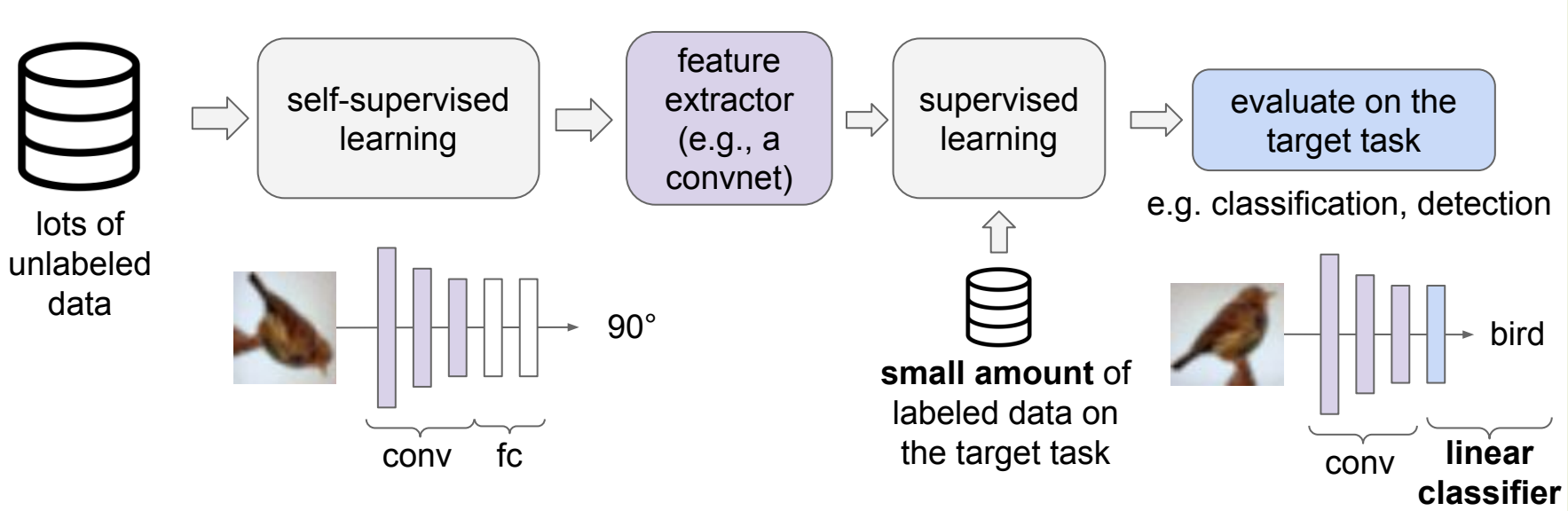
rotation prediction

“jigsaw puzzle”

colorization

- Solving the pretext tasks allow the model to learn good features.
- We can automatically generate labels for the pretext tasks.

Evaluation of self-supervised learning



1. Learn good feature extractors from self-supervised pretext tasks, e.g., predicting image rotations

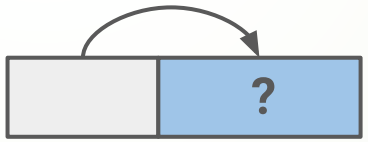
2. Attach a shallow network on the feature extractor; train the shallow network on the target task with small amount of labeled data

Methods of Self-Supervised Learning

- Self-prediction
- Contrastive learning

Self-Prediction

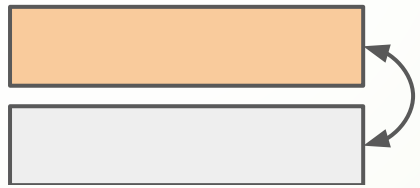
- **Self-prediction**: Given an individual data sample, the task is to predict one part of the sample given the other part.
- The part to be predicted pretends to be missing.



"Intra-sample" prediction: reconstruction

Contrastive Learning

- **Contrastive learning**: Given multiple data samples, the task is to predict the relationship among them.
- The multiple samples can be selected from the dataset based on some known logics (e.g. the order of words / sentences), or fabricated by altering the original version.

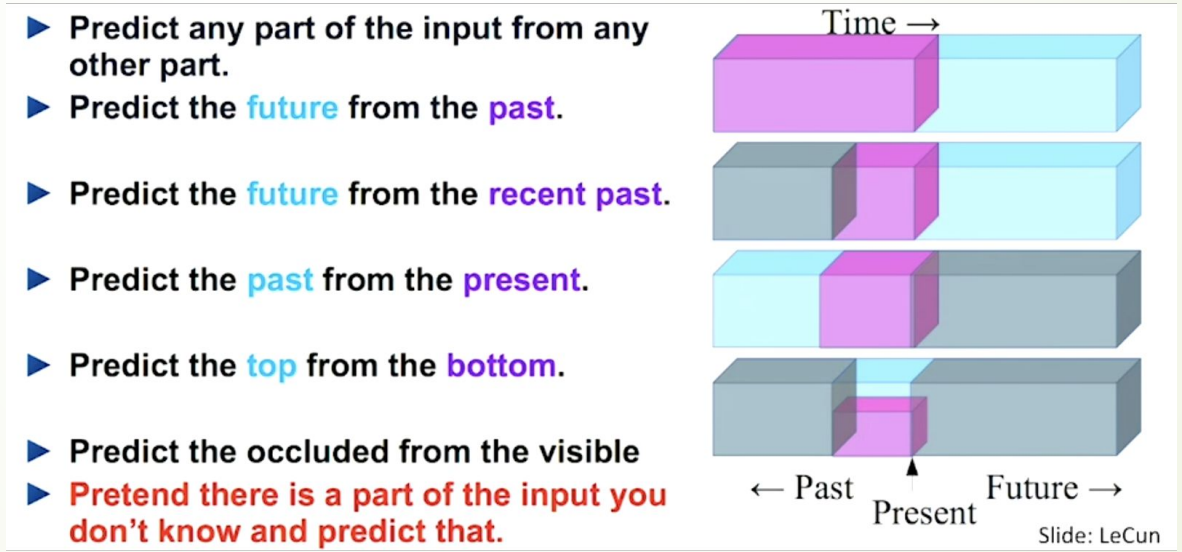


"Inter-sample" prediction: similar or not?

Self-prediction

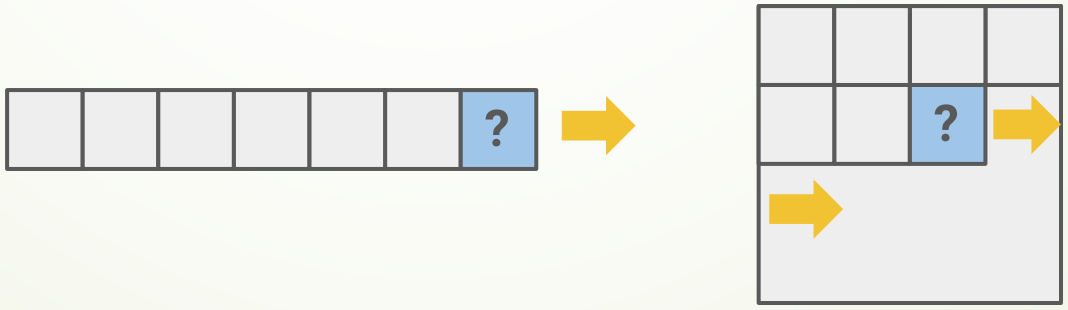
Self-Prediction

- Self-prediction construct prediction tasks within every individual data sample: to predict a part of the data from the rest while pretending we don't know that part.



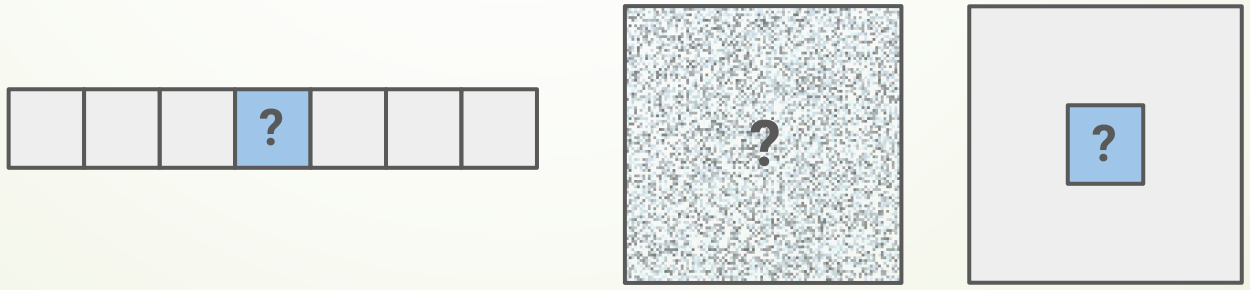
Self-Prediction: Autoregressive Generation

- The autoregressive model predicts future behavior based on past behavior. Any data that comes with an innate sequential order can be modeled with regression.
- *Examples*:
  - Audio (WaveNet, WaveRNN)
  - Autoregressive language modeling (GPT, XLNet)
  - Images in raster scan (PixelCNN, PixelRNN, iGPT)



Self-Prediction: Masked Generation

- We mask a random portion of information and pretend it is missing, irrespective of the natural sequence. The model learns to predict the missing portion given other unmasked information.
- *Examples*:
  - Masked language modeling (BERT)
  - Images with masked patch (denoising autoencoder, context autoencoder, colorization)



Example: Masked AutoEncoder (MAE)

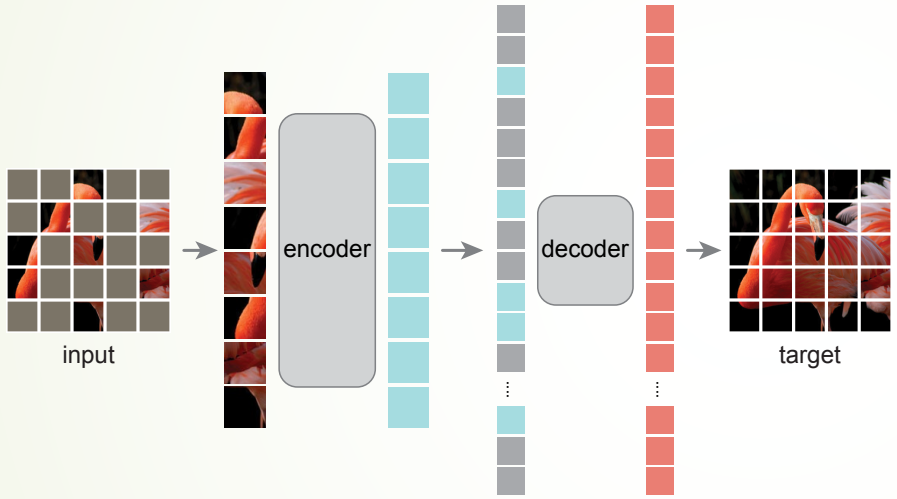


Figure 1. **Our MAE architecture**. During pre-training, a large random subset of image patches (*e.g.*, 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed 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 (full sets of patches) for recognition tasks.

- A large random subset of image patches (75%) is masked out.
- Masked token is introduced after Encoder.

- Source: He et al. 2021. arXiv:2111.06377.

Example: Masked AutoEncoder (MAE)

[Source: He et al. 2021. arXiv:2111.06377. Masked Autoencoders Are Scalable Vision Learners]

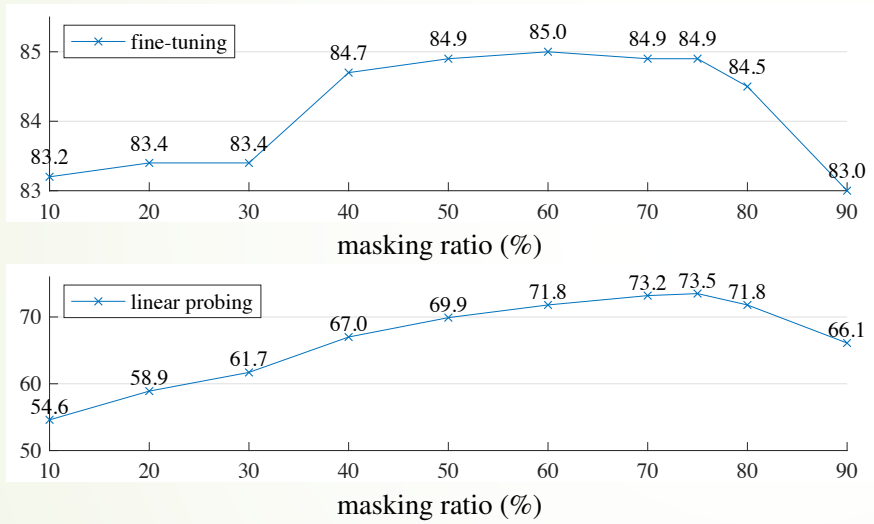
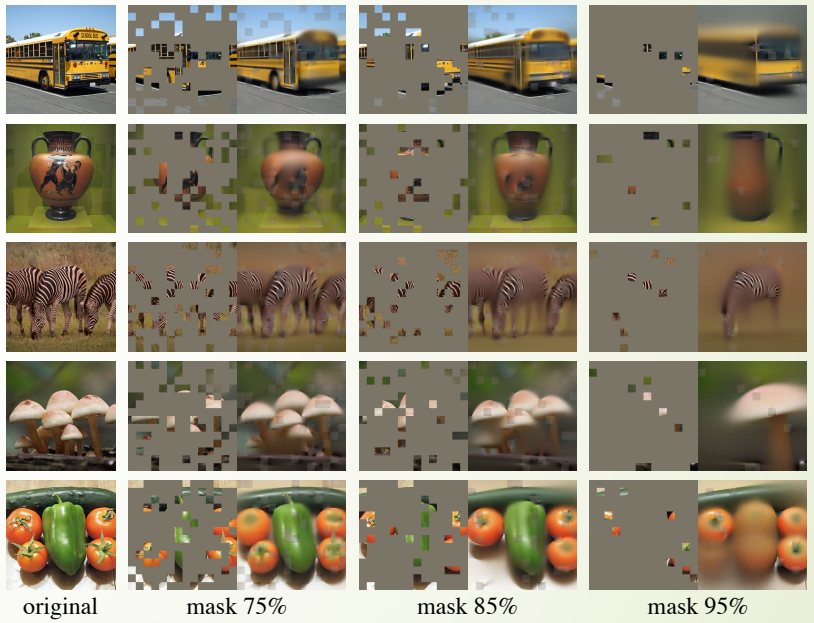


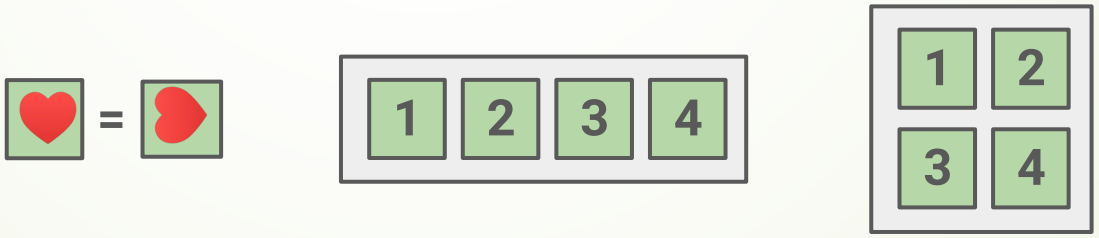
Figure 5. **Masking ratio**. A high masking ratio (75%) works well for both fine-tuning (top) and linear probing (bottom). The y-axes are ImageNet-1K validation accuracy (%) in all plots in this paper.

- High masking ratio (75%) works well.



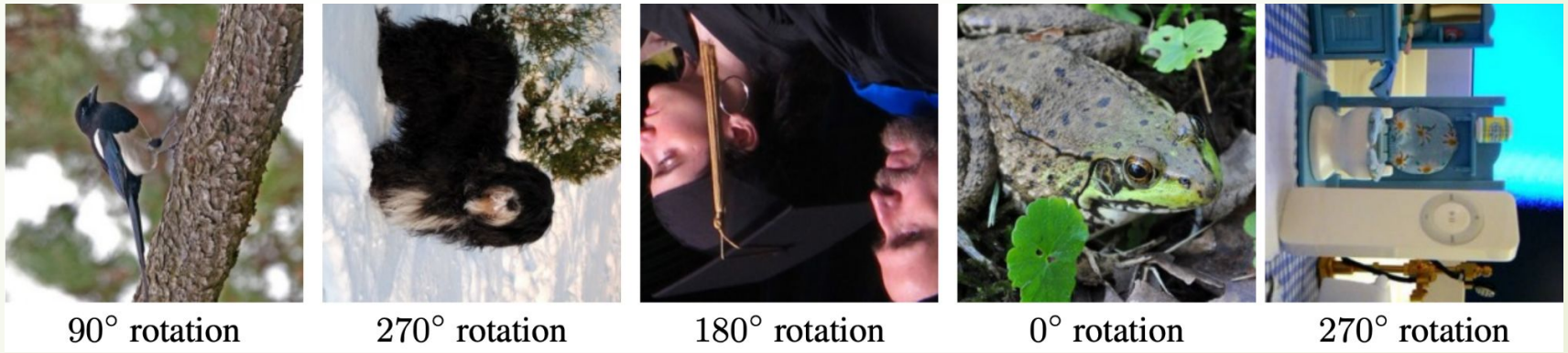
Self-Prediction: Innate Relationship Prediction

- Some transformation (e.g. rotation, inpainting, jigsaw puzzle, coloring) of one data sample should maintain the original information or follow the desired innate logic.
- *Examples*:
  - Order of image patches (e.g., relative position, jigsaw puzzle)
  - Image rotation
  - Counting features across patches



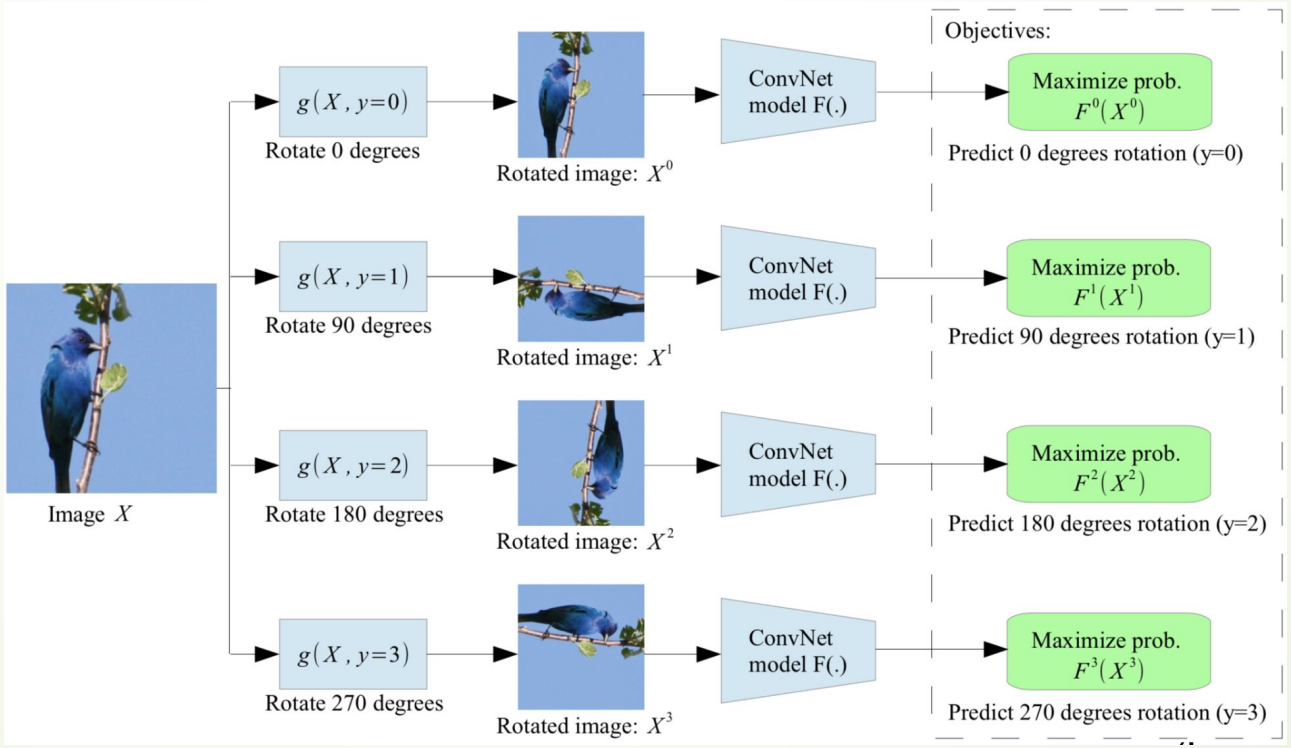
Pretext task: predict rotations

- **Hypothesis**: a model could recognize the correct rotation of an object only if it has the "visual commonsense" of what the object should look like unperturbed.

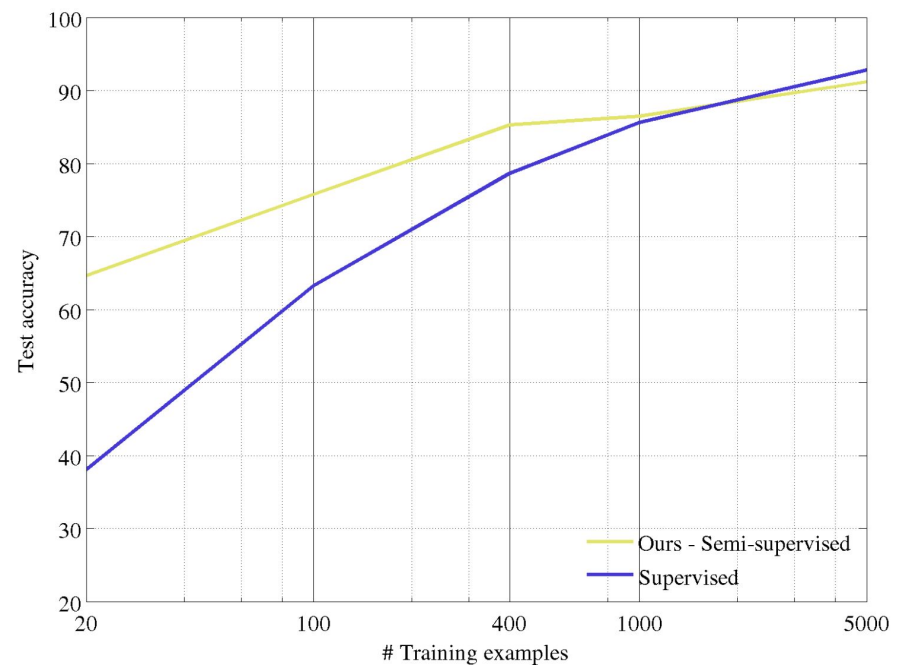


Rotation prediction

- Self-supervised learning by rotating the entire input images.
- The model learns to predict which rotation is applied (4-way classification)



Evaluation on semi-supervised learning



- Self-supervised learning on **CIFAR10** (entire training set).
- Freeze conv1 + conv2
  Learn **conv3 + linear** layers with subset of labeled CIFAR10 data (classification).

- Gidaris, et al. ICLR 2018, arXiv:1803.07728

Evaluation of transfer learning

	Classification (%mAP)		Detection (%mAP)	Segmentation (%mIoU)
Trained layers	fc6-8	all	all	all
ImageNet labels	78.9	79.9	56.8	48.0
Random		53.3	43.4	19.8
Random rescaled Krähenbühl et al. (2015)	39.2	56.6	45.6	32.6
Egomotion (Agrawal et al., 2015)	31.0	54.2	43.9	
Context Encoders (Pathak et al., 2016b)	34.6	56.5	44.5	29.7
Tracking (Wang & Gupta, 2015)	55.6	63.1	47.4	
Context (Doersch et al., 2015)	55.1	65.3	51.1	
Colorization (Zhang et al., 2016a)	61.5	65.6	46.9	35.6
BIGAN (Donahue et al., 2016)	52.3	60.1	46.9	34.9
Jigsaw Puzzles (Noroozi & Favaro, 2016)	-	67.6	53.2	37.6
NAT (Bojanowski & Joulin, 2017)	56.7	65.3	49.4	
Split-Brain (Zhang et al., 2016b)	63.0	67.1	46.7	36.0
ColorProxy (Larsson et al., 2017)		65.9		38.4
Counting (Noroozi et al., 2017)	-	67.7	51.4	36.6
(Ours) RotNet	**70.87**	**72.97**	**54.4**	**39.1**

Pretrained with full ImageNet supervision

No pretraining

Self-supervised learning with rotation prediction

Self-supervised learning on **ImageNet** (entire training set) with AlexNet.

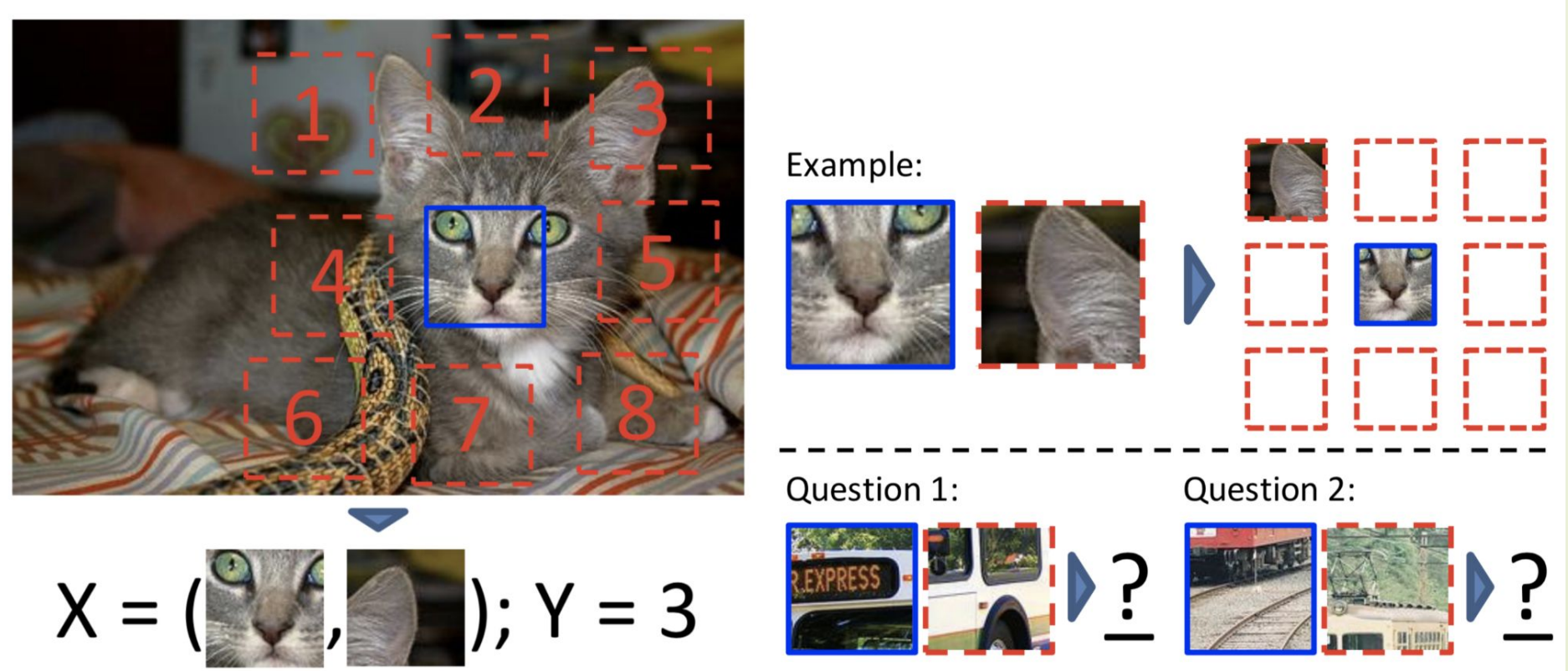
Finetune on labeled data from **Pascal VOC 2007**.

source: Gidaris et al. 2018

Visualize learned features

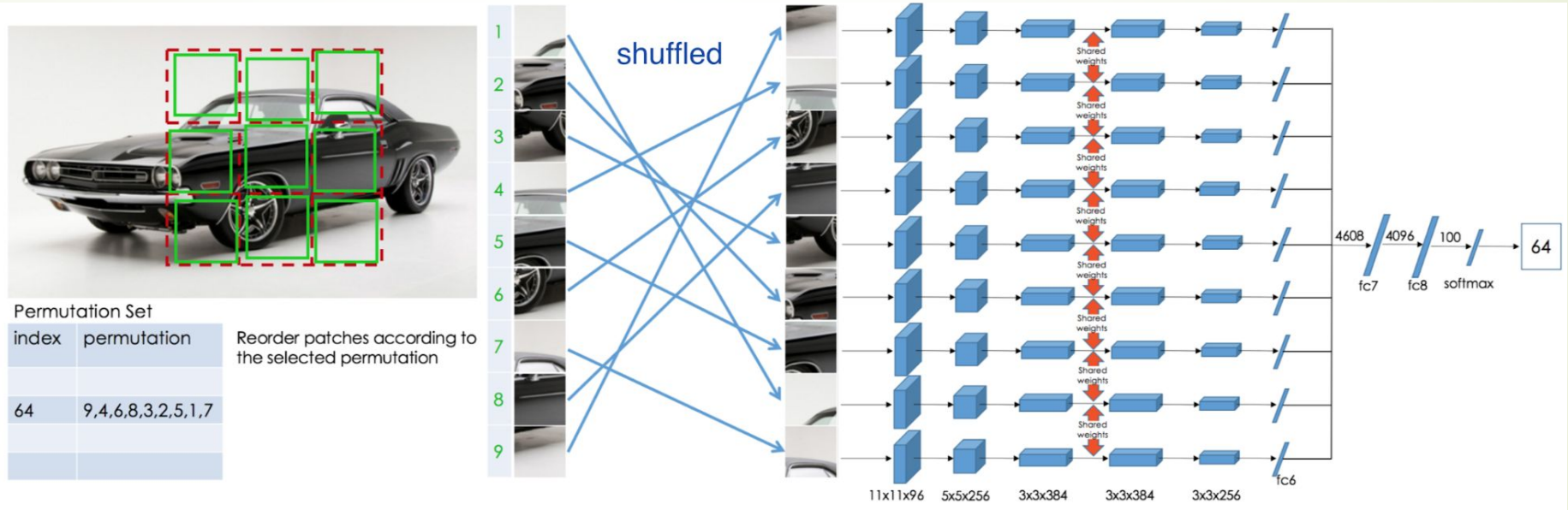
(a) **Attention maps of supervised model** (b) **Attention maps of our self-supervised model**

Pretext task: predict relative patch locations



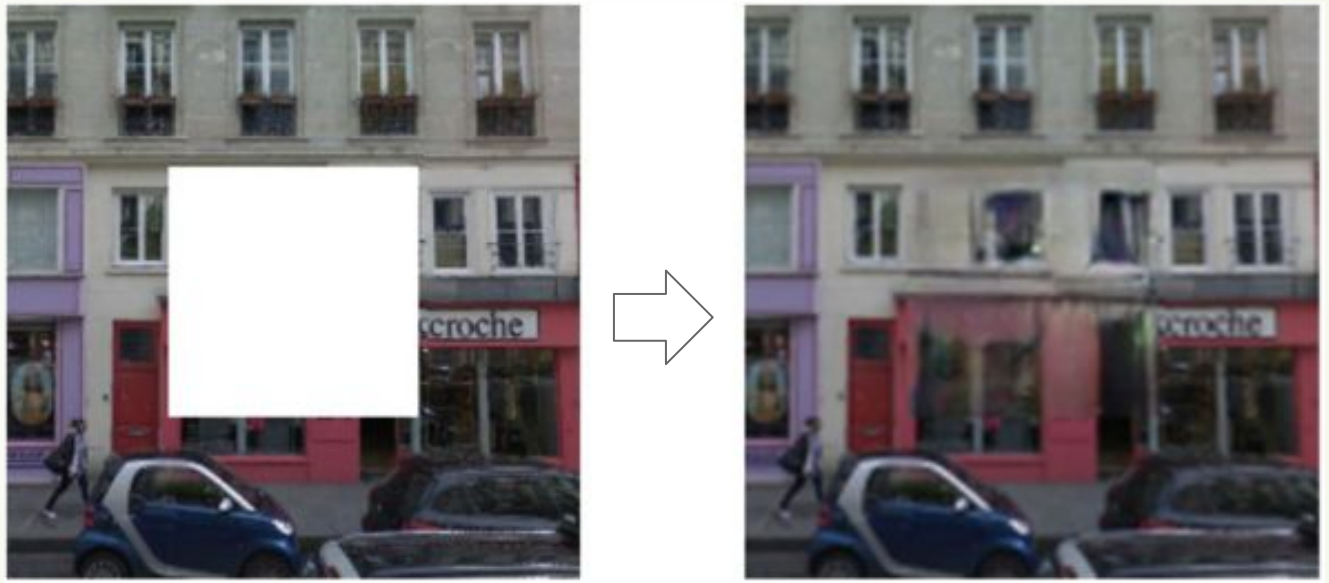
(Image source: Doersch et al., ICCV 2015)

Pretext task: solving "jigsaw puzzles"



(Image source: Noroozi & Favaro, 2016)

Pretext task: predict missing pixels (inpainting)



Context Encoders: Feature Learning by Inpainting (Pathak et al., 2016)

Learning to inpaint by reconstruction

- Learning to reconstruct the missing pixels

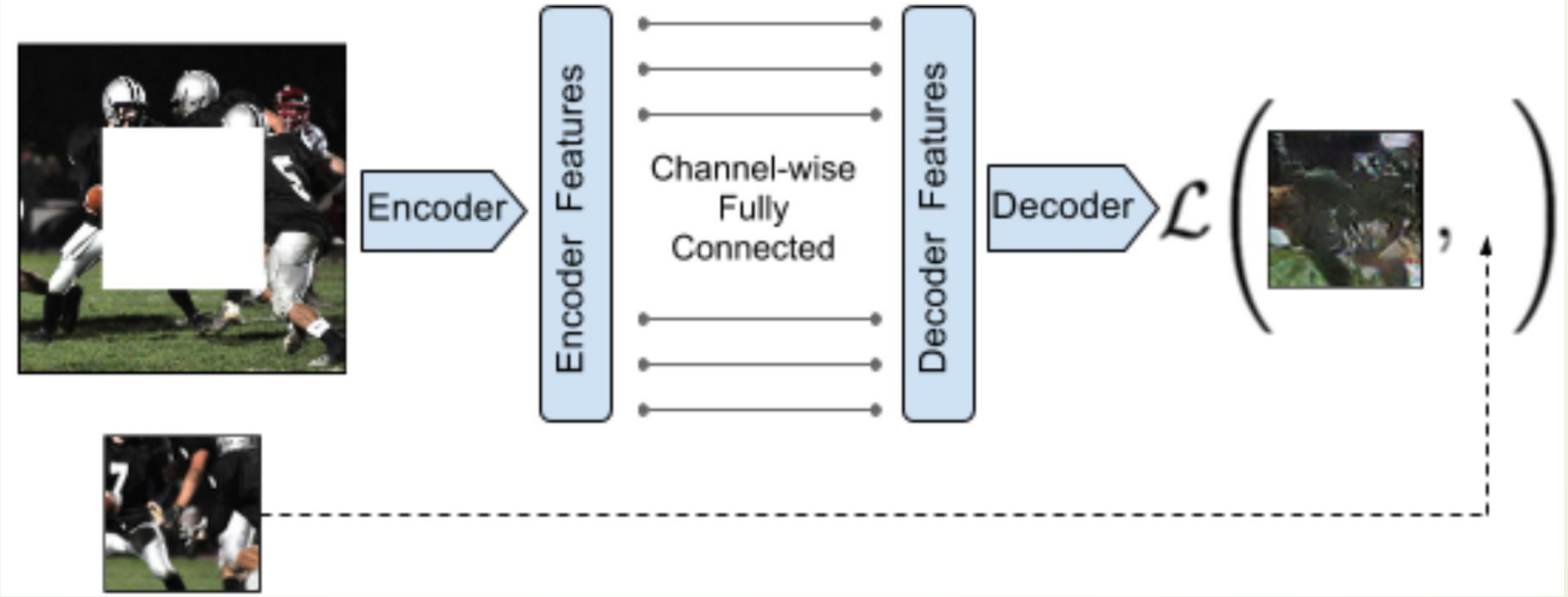


Image Source: Pathak et al., 2016

Learning to inpaint by reconstruction

- Loss = reconstruction + adversarial learning
- Adversarial loss between "real" images and *inpainted images*

$$L(x) = L_{recon}(x) + L_{adv}(x)$$

$$L_{recon}(x) = ||M * (x - F_\theta((1 - M) * x))||_2^2$$

$$L_{adv} = \max_D \mathbb{E}[\log(D(x))] + \log(1 - D(F((1 - M) * x)))]$$

Inpainting evaluation

Input (context) reconstruction adversarial recon + adv

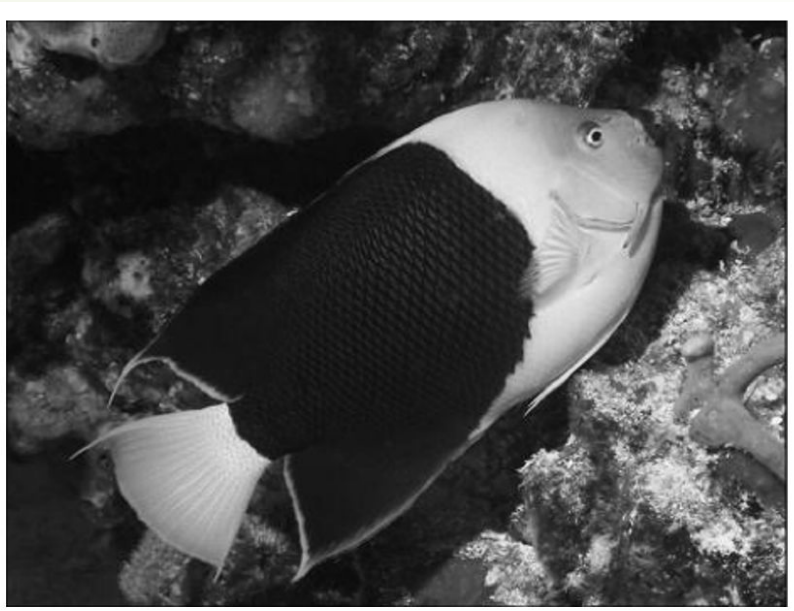
Transfer features to classification

- Self-supervised learning on ImageNet training set, transfer to classification (Pascal VOC 2007), detection (Pascal VOC 2007), and semantic segmentation (Pascal VOC 2012)

Pretraining Method	Supervision	Pretraining time	Classification	Detection	Segmentation
ImageNet [26]	1000 class labels	3 days	78.2%	56.8%	48.0%
Random Gaussian	initialization	< 1 minute	53.3%	43.4%	19.8%
Autoencoder	-	14 hours	53.8%	41.9%	25.2%
Agrawal *et al.* [1]	egomotion	10 hours	52.9%	41.8%	-
Wang *et al.* [39]	motion	1 week	58.7%	47.4%	-
Doersch *et al.* [7]	relative context	4 weeks	55.3%	46.6%	-
Ours	context	14 hours	56.5%	44.5%	30.0%

Source: Pathak et al., 2016

Pretext task: image coloring

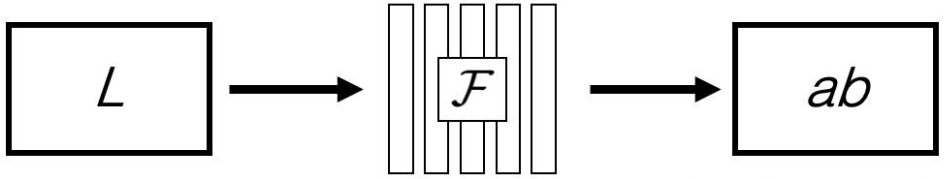


Grayscale image: *L* channel

$\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$

Color information: *ab* channels

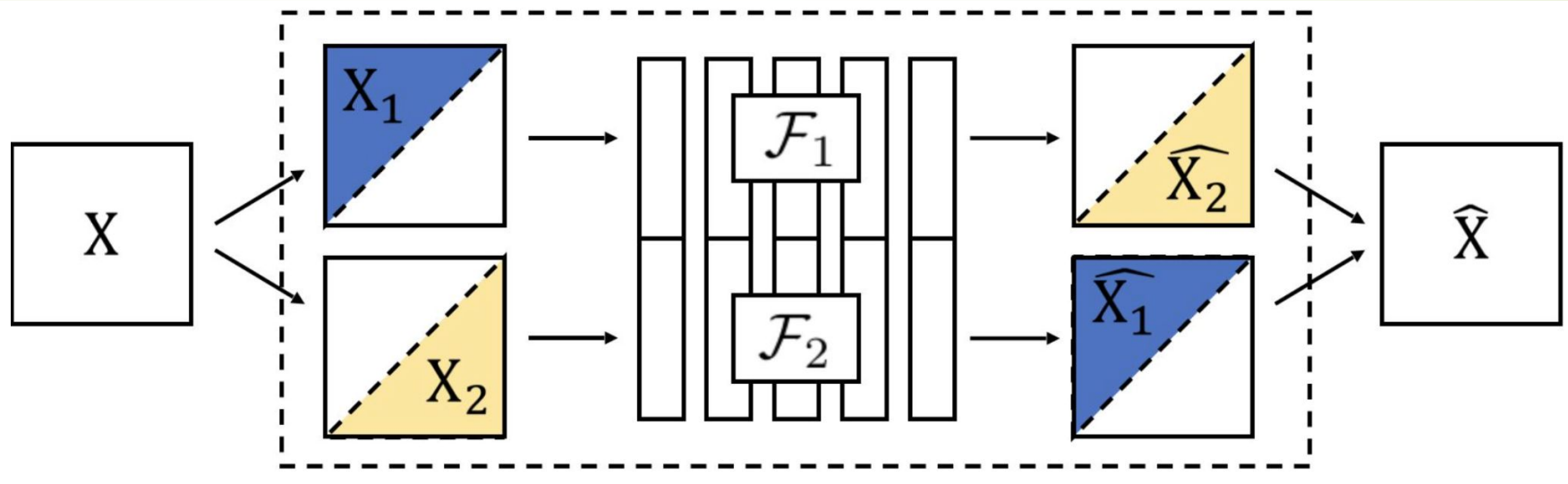
$\hat{\mathbf{Y}} \in \mathbb{R}^{H \times W \times 2}$



Source: Richard Zhang / Phillip Isola

Learning features from colorization: Split-brain Autoencoder

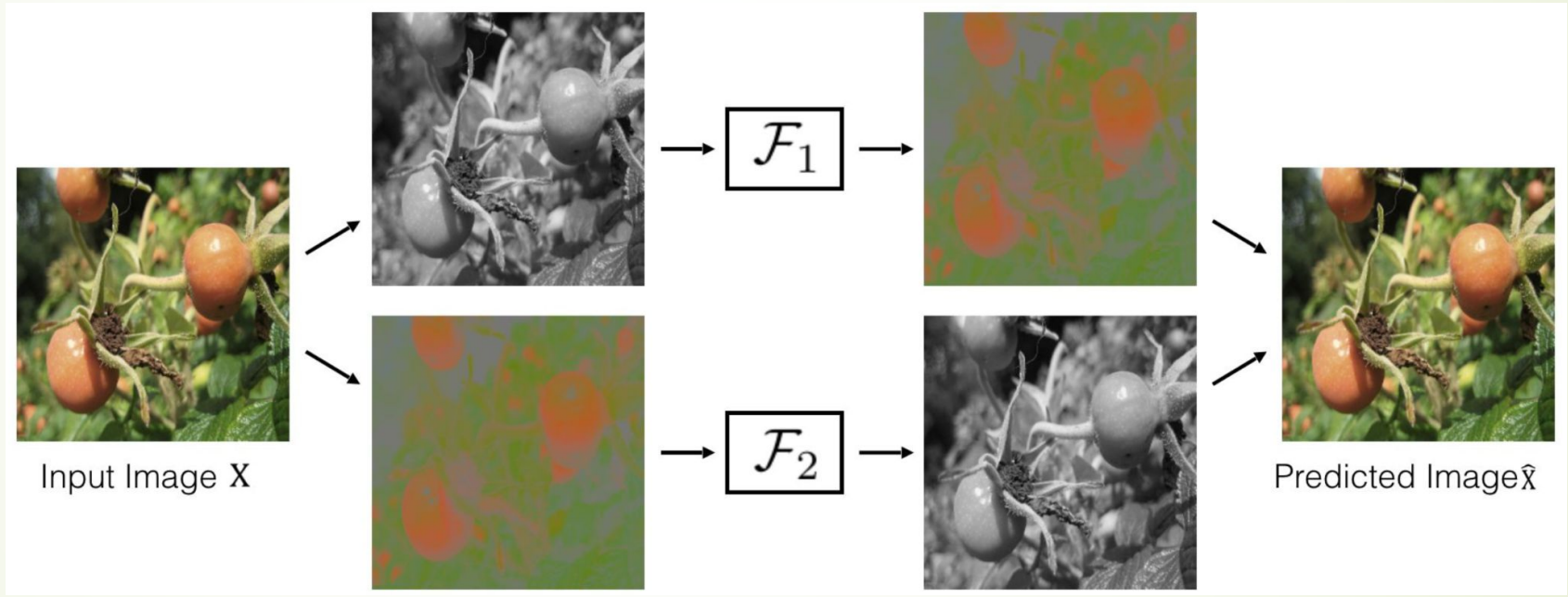
- Cross-channel prediction



Source: Richard Zhang / Phillip Isola

Learning features from colorization: Split-brain Autoencoder

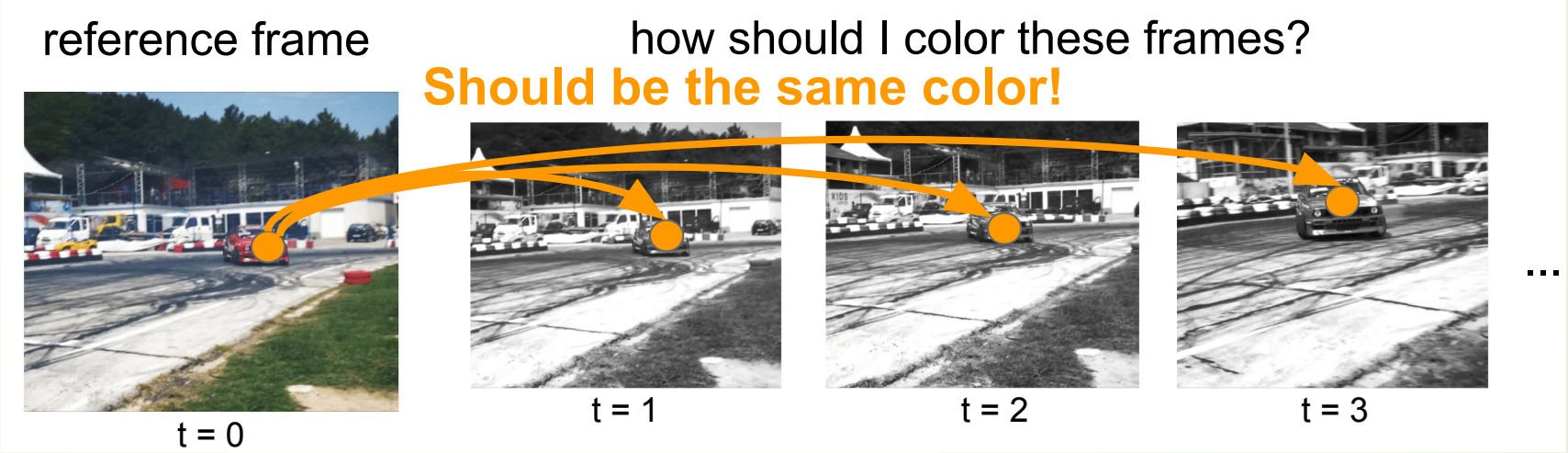
- Example:



Source: Richard Zhang / Phillip Isola

Pretext task: video coloring

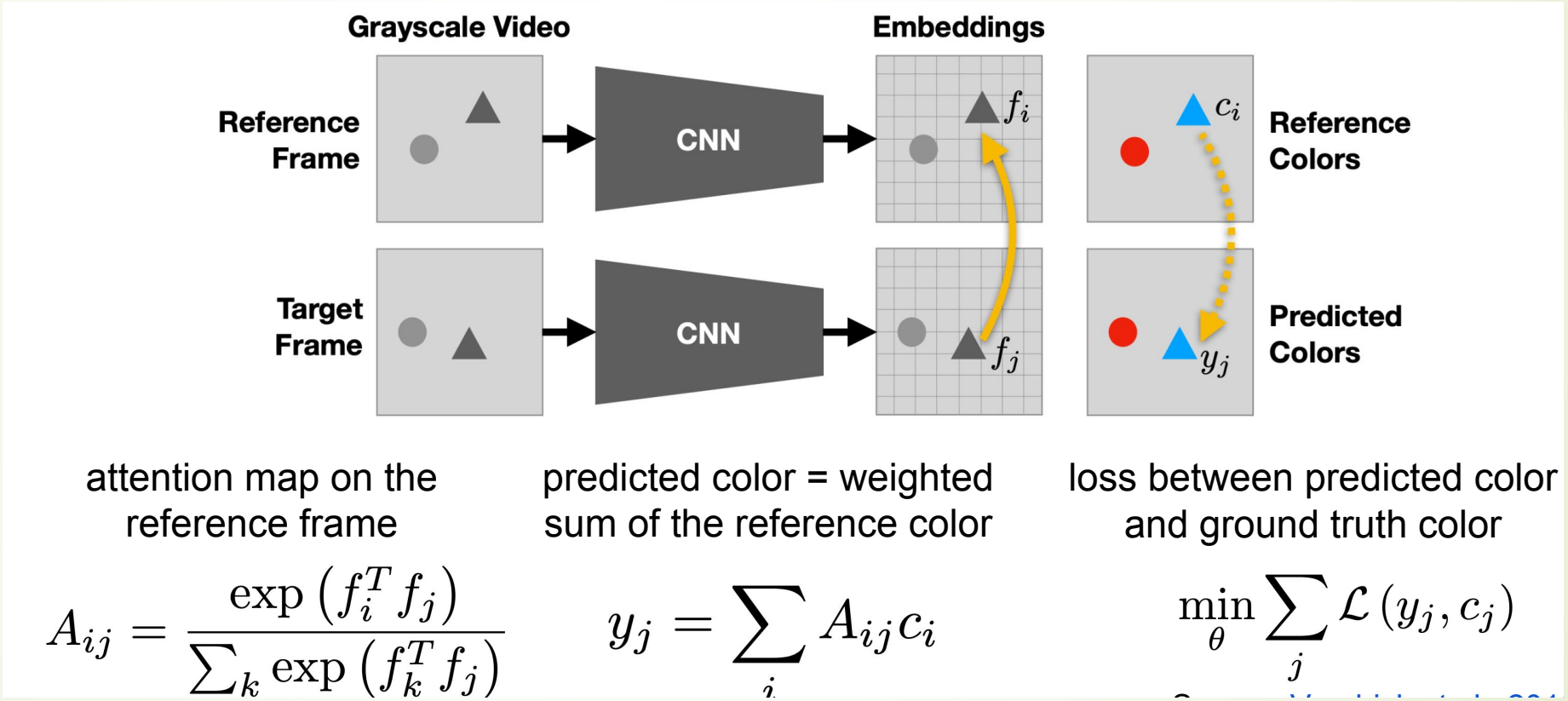
- Idea: model the temporal coherence of colors in videos
- Hypothesis: learning to color video frames should allow model to learn to track regions or objects without labels!



Vondrick et al., 2018

Pretext task: video coloring

- Idea: model the temporal coherence of colors in videos



attention map on the reference frame

$$A_{ij} = \frac{\exp\left(f_i^T f_j\right)}{\sum_k \exp\left(f_k^T f_j\right)}$$

predicted color = weighted sum of the reference color

$$y_j = \sum_i A_{ij} c_i$$

loss between predicted color and ground truth color

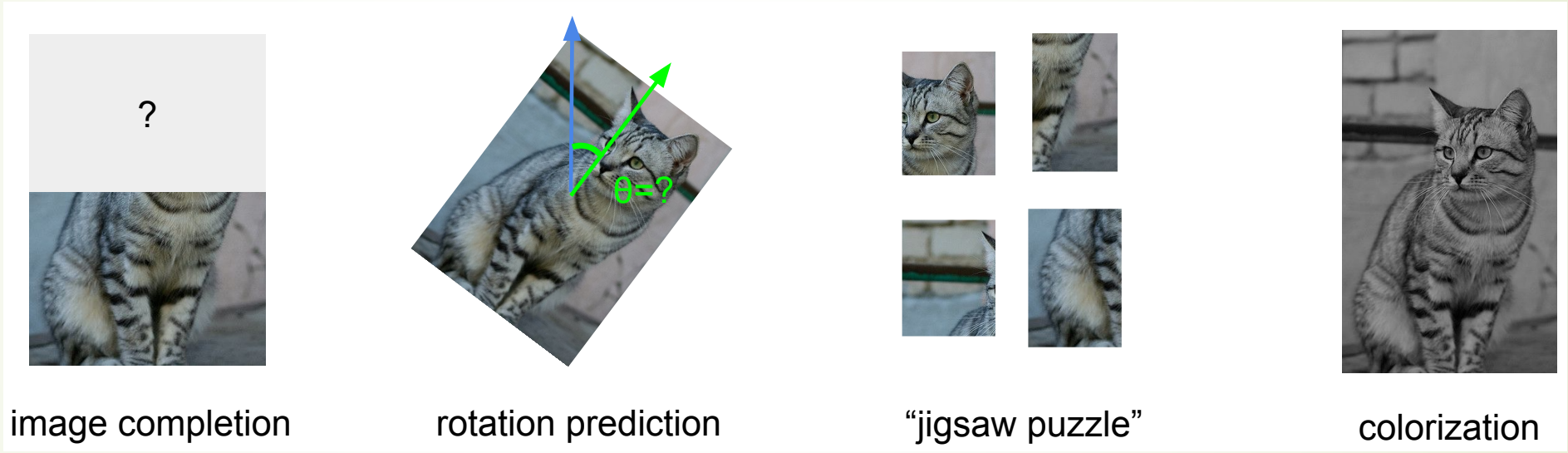
$$\min_\theta \sum_j \mathcal{L}\left(y_j, c_j\right)$$

Vondrick et al., 2018

Summary

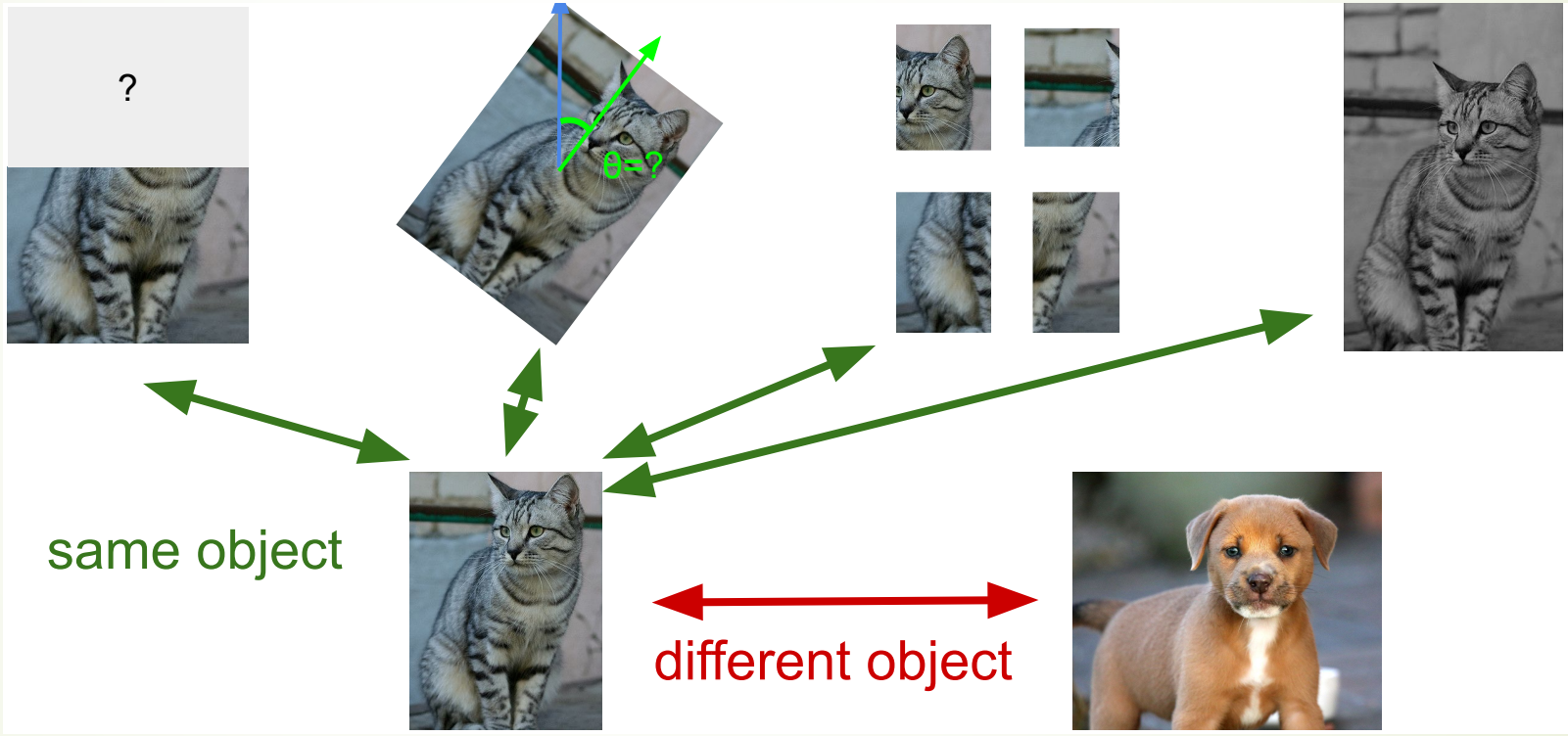
- Pretext tasks focus on "visual common sense", e.g., predict rotations, inpainting, rearrangement, and colorization.
- The models are forced learn good features about natural images, e.g., semantic representation of an object category, in order to solve the pretext tasks.
- We don't care about the performance of these pretext tasks, but rather how useful the learned features are for downstream tasks (classification, detection, segmentation).
- *Problems:*
  - *1) coming up with individual pretext tasks is tedious, and*
  - *2) the learned representations may not be general.*

Pretext tasks from image transformations

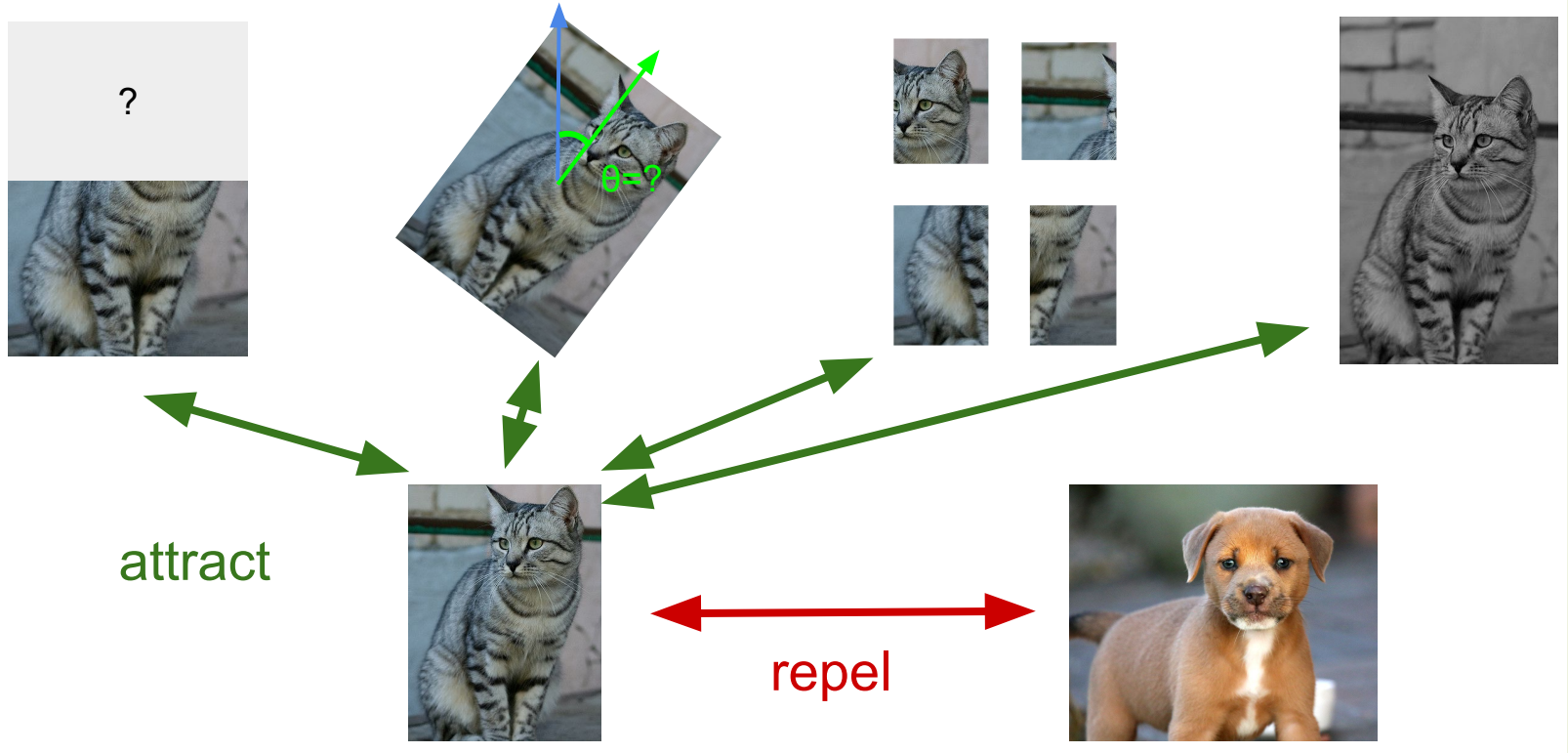


- Learned representations may be tied to a specific pretext task!
- Can we come up with a more general pretext task?

A more general pretext task?



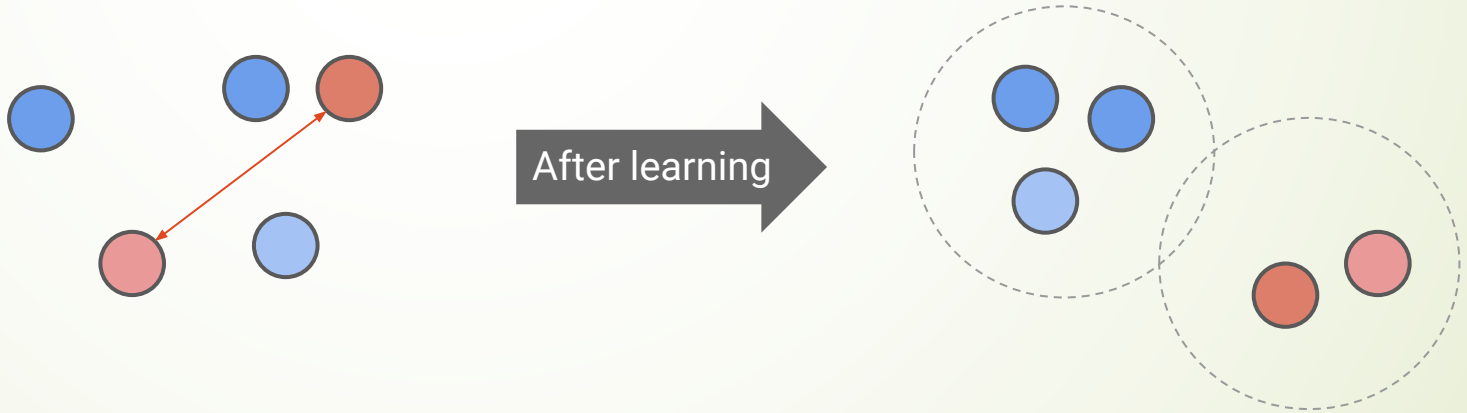
Contrastive Learning



Contrastive learning

Contrastive Learning

- The goal of contrastive representation learning is to learn such an embedding space in which *similar* sample pairs stay *close* to each other while *dissimilar* ones are *far apart*.



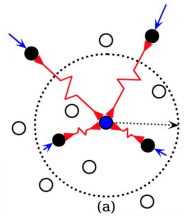
Early work on metric and contrastive learning

Metric learning (Xing et al. 2002)

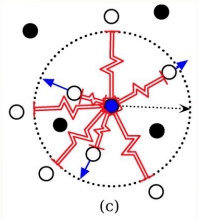
$$d_A(x, y) = ||x - y||_A = \sqrt{(x - y)^T A (x - y)}$$

Contrastive Loss (Chopra & Hadsell et al. 2005)

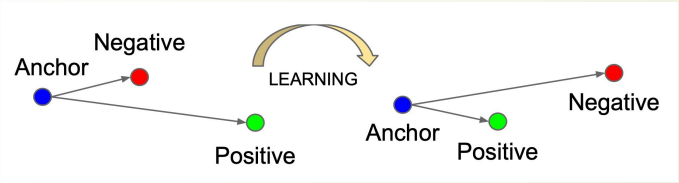
i. If $Y_{ij} = 0$, then update W to decrease $D_W = \|G_W(\vec{X}_i) - G_W(\vec{X}_j)\|_2$



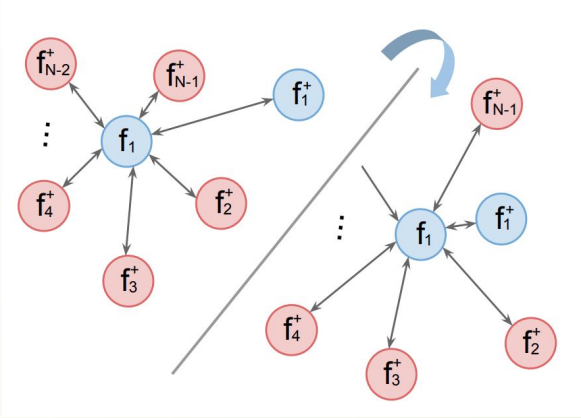
ii. If $Y_{ij} = 1$, then update W to increase $D_W = \|G_W(\vec{X}_i) - G_W(\vec{X}_j)\|_2$



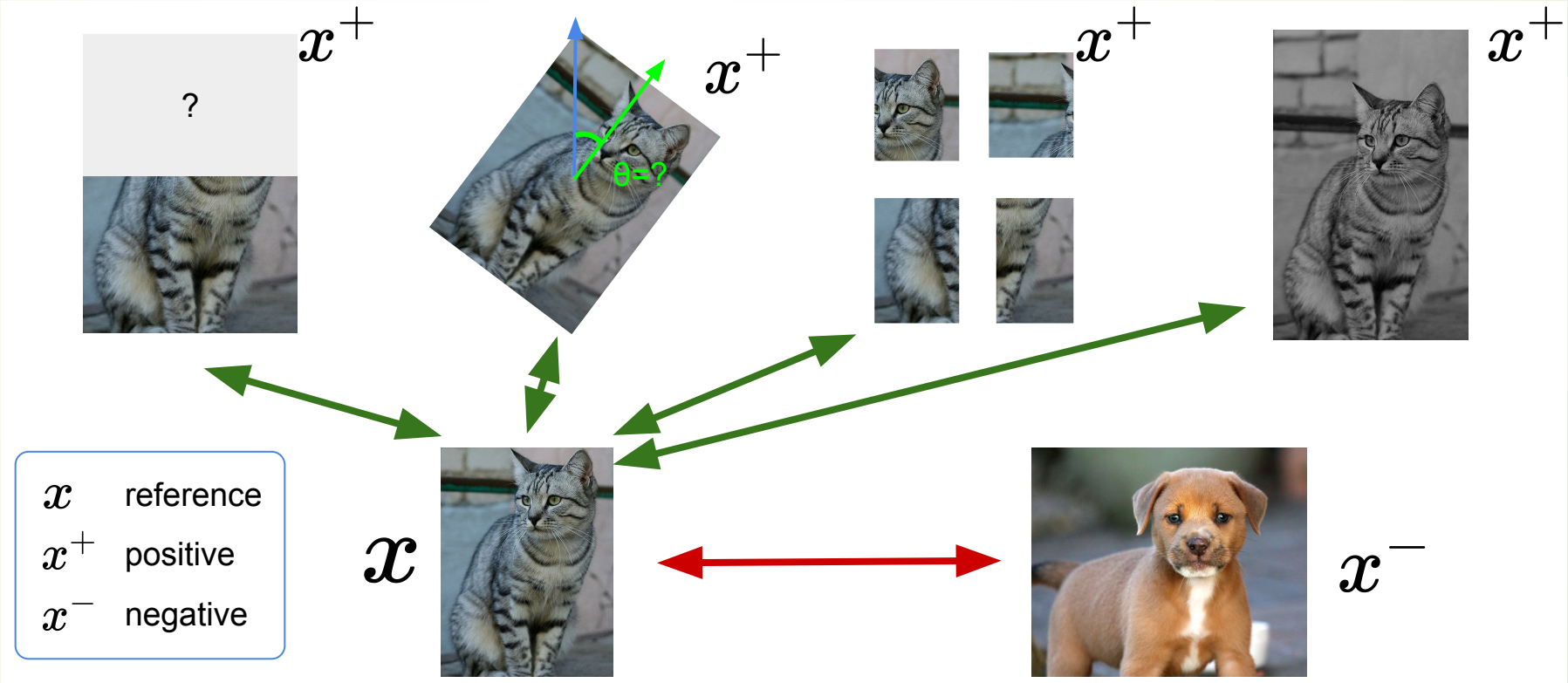
Triplet loss (Schroff et al. 2015)



N-pair loss (Sohn 2016)



Contrastive Learning of Images



A Modern Formulation

- *For* x: reference sample; x+ positive sample; x- negative sample

$$\text{score}(f(x), f(x^+)) >> \text{score}(f(x), f(x^-))$$

- Given a chosen score function, we aim to learn an **encoder function** *f* that yields high score for positive pairs (x, x+) and low scores for negative pairs (x, x-).

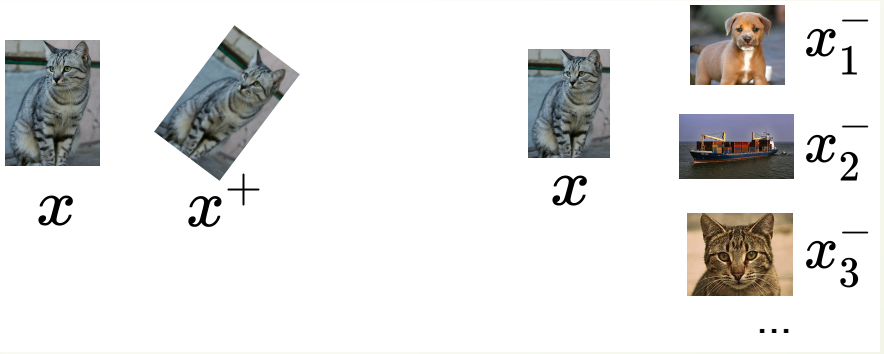
Contrastive loss function

- Given 1 positive sample and N-1 negative samples, cross-entropy loss:

$$L = -\mathbb{E}_X \left[\log \frac{\exp(s(f(x), f(x^+))}{\exp(s(f(x), f(x^+)) + \sum_{j=1}^{N-1} \exp(s(f(x), f(x_j^-))} \right]$$

score for the positive pair

score for the N-1 negative pairs



InfoNCE loss

Loss function given 1 positive sample and N - 1 negative samples:

$$L = -\mathbb{E}_X \left[\log \frac{\exp(s(f(x), f(x^+))}{\exp(s(f(x), f(x^+)) + \sum_{j=1}^{N-1} \exp(s(f(x), f(x_j^-))} \right]$$

Commonly known as the InfoNCE loss (van den Oord et al., 2018)

A *lower bound* on the mutual information between *f(x)* and $f(x^+)$

$$MI[f(x), f(x^+)] - \log(N) \geq -L$$

The larger the negative sample size (*N*), the tighter the bound

Detailed derivation: Poole et al., 2019

Detailed derivation: Poole et al., 2019

SimCLR: A Simple Framework for Contrastive Learning

- Cosine similarity as the score function:

$$s_{i,j} = \frac{z_i^T z_j}{||z_i|| \, ||z_j||}$$

- Generate positive samples through data augmentation:
  - random cropping, random color distortion, and random blur.
- Base encoder ***f*** and projection head ***g*** (discarded after training) are trained with contrastive loss

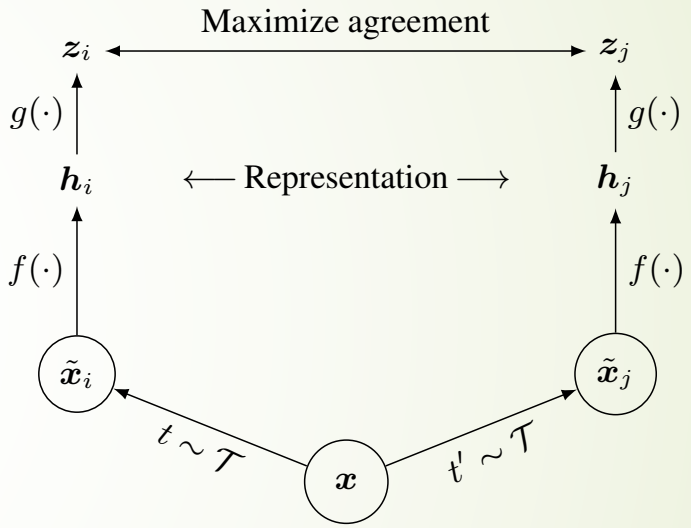
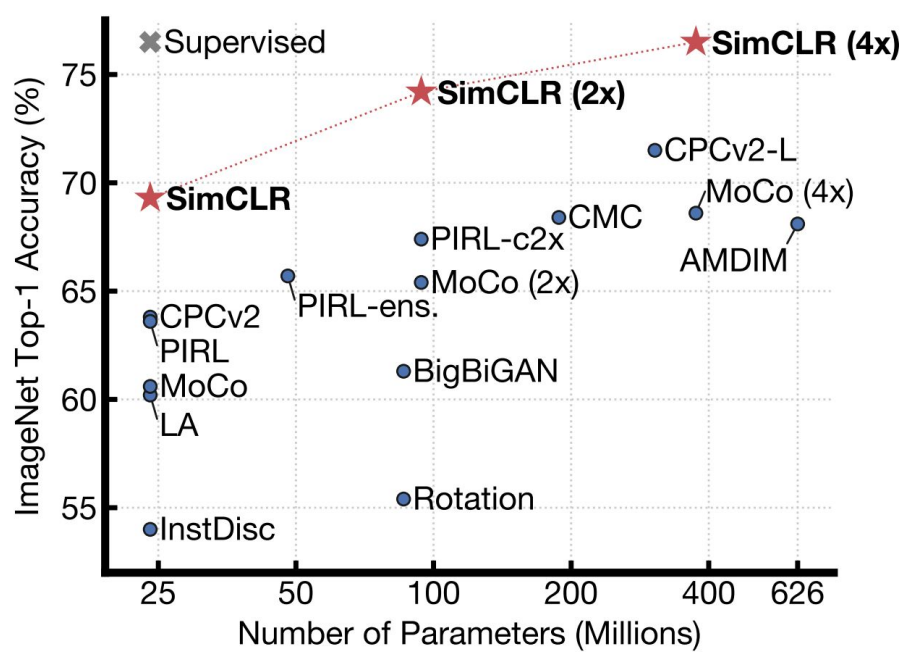


Figure 2. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder network $f(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement using a contrastive loss. After training is completed, we throw away the projection head $g(\cdot)$ and use encoder $f(\cdot)$ and representation $\boldsymbol{h}$ for downstream tasks.

Source: Chen et al., ICML 2020

Training linear classifier on SimCLR features



- Train feature encoder on **ImageNet** (entire training set) using SimCLR.
- Freeze feature encoder, train a linear classifier on top with labeled data.

- Chen et al. ICML 2020

Semi-supervised learning on SimCLR features

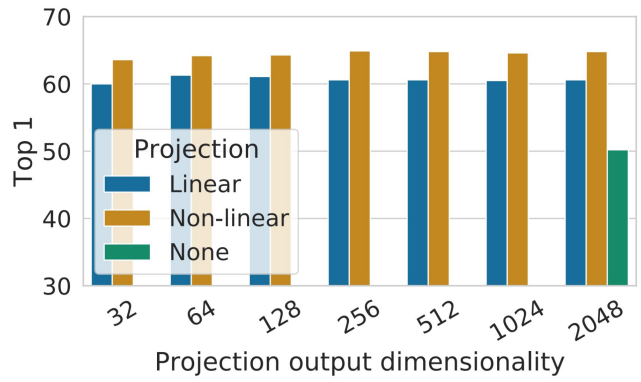
Method	Architecture	Label fraction 1%	Label fraction 10%
		Top 5	Top 5
Supervised baseline	ResNet-50	48.4	80.4
Methods using other label-propagation:			
Pseudo-label	ResNet-50	51.6	82.4
VAT+Entropy Min.	ResNet-50	47.0	83.4
UDA (w. RandAug)	ResNet-50	-	88.5
FixMatch (w. RandAug)	ResNet-50	-	89.1
S4L (Rot+VAT+En. 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**

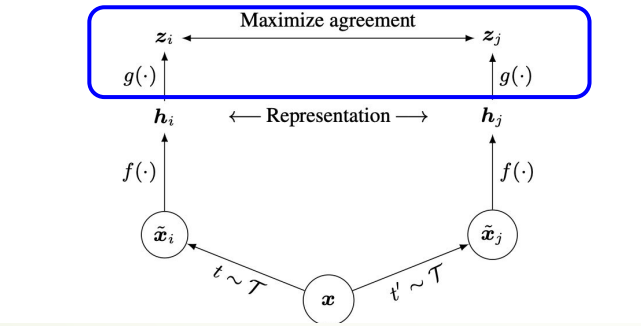
Table 7. ImageNet accuracy of models trained with few labels.

- Train feature encoder on **ImageNet** (entire training set) using SimCLR.
- **Finetune** the encoder with 1% / 10% of labeled data on ImageNet.

- Chen et al. ICML 2020

SimCLR: projection head





- Linear / non-linear projection heads improve representation learning.
- A possible explanation:
  - contrastive learning objective may discard useful information for downstream tasks
  - representation space **z** is trained to be invariant to data transformation.
  - by leveraging the projection head **g(·)**, more information can be preserved in the **h** representation space

- Chen et al. ICML 2020

SimCLR needs large batch size!

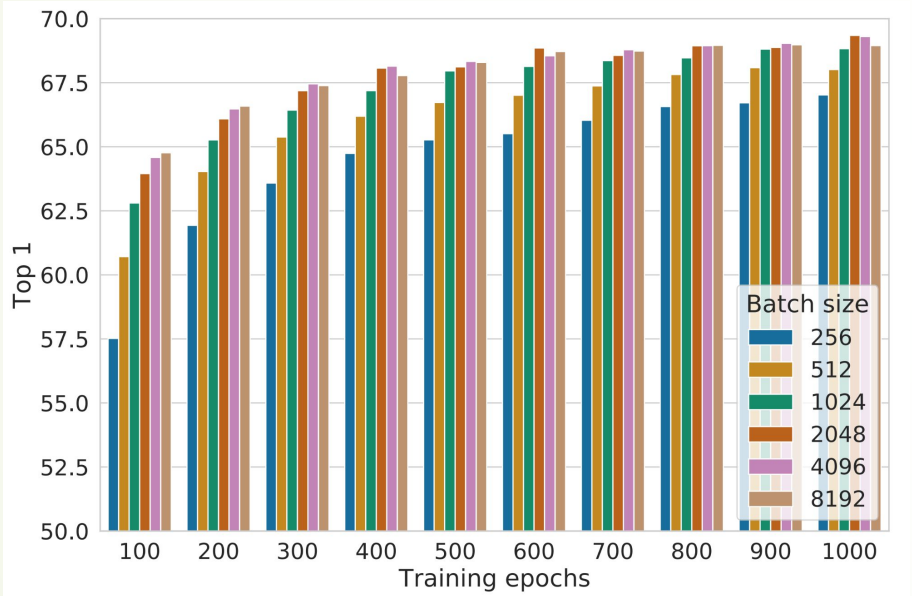


Figure 9. Linear evaluation models (ResNet-50) trained with different batch size and epochs. Each bar is a single run from scratch.[10]

- Large training batch size is crucial for SimCLR!
- Large batch size causes large memory footprint during backpropagation: requires distributed training on TPUs (ImageNet experiments)

- Chen et al. ICML 2020

Momentum Contrastive Learning (MoCo)

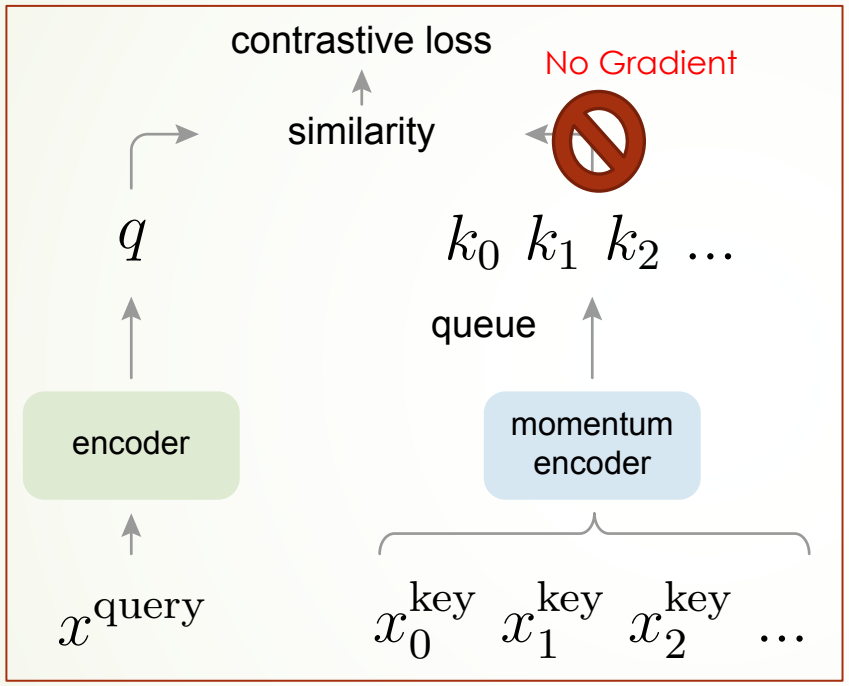


Figure 1. Momentum Contrast (MoCo) trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. The dictionary keys $\{k_0, k_1, k_2, ...\}$ are defined on-the-fly by a set of data samples. The dictionary is built as a queue, with the current mini-batch enqueued and the oldest mini-batch dequeued, decoupling it from the mini-batch size. The keys are encoded by a slowly progressing encoder, driven by a momentum update with the query encoder. This method enables a large and consistent dictionary for learning visual representations.

- **Key differences to SimCLR:**
  - Keep a running queue of keys (negative samples).
  - Compute gradients and update the encoder only through the queries.
  - Decouple min-batch size with the number of keys: can support a large number of negative samples.
  - The key encoder is slowly progressing through the momentum update rules:

$$\theta_{\mathrm{k}} \leftarrow m\theta_{\mathrm{k}} + (1-m)\theta_{\mathrm{q}}$$

He et al. CVPR 2020

MoCo v2

- A hybrid of ideas from SimCLR and MoCo:
  - From SimCLR: non-linear projection head and strong data augmentation.
  - From MoCo: momentum-updated queues that allow training on a large number of negative samples (no TPU required!).

Improved Baselines with Momentum Contrastive Learning

Xinlei Chen Haoqi Fan Ross Girshick Kaiming He
Facebook AI Research (FAIR)

Source: Chen et al., 2020

MoCo v2 vs. SimCLR

case	unsup. pre-train MLP	aug+	cos	epochs	batch	ImageNet acc.
MoCo v1 [6]				200	256	60.6
SimCLR [2]	✓	✓	✓	200	256	61.9
SimCLR [2]	✓	✓	✓	200	8192	66.6
MoCo v2	✓	✓	✓	200	256	**67.5**
*results of **longer** unsupervised training follow:*						
SimCLR [2]	✓	✓	✓	1000	4096	69.3
MoCo v2	✓	✓	✓	800	256	**71.1**

Table 2. **MoCo *vs.* SimCLR**: ImageNet linear classifier accuracy (**ResNet-50, 1-crop 224×224**), trained on features from unsupervised pre-training. "aug+" in SimCLR includes blur and stronger color distortion. SimCLR ablations are from Fig. 9 in [2] (we thank the authors for providing the numerical results).

mechanism	batch	memory / GPU	time / 200-ep.
MoCo	256	**5.0G**	**53 hrs**
end-to-end	256	7.4G	65 hrs
end-to-end	4096	93.0G†	n/a

Table 3. **Memory and time cost** in 8 V100 16G GPUs, implemented in PyTorch. †: based on our estimation.

- Non-linear projection head and strong data augmentation are crucial for contrastive learning.
- Decoupling mini-batch size with negative sample size allows MoCo-V2 to outperform SimCLR with smaller batch size (256 vs. 8192).
- … all with much smaller memory footprint! ("end-to-end" means SimCLR here)

- Chen et al. 2020

MoCo v3: the State of the Art

An Empirical Study of Training Self-Supervised Vision Transformers

Xinlei Chen* Saining Xie* Kaiming He
Facebook AI Research (FAIR)

Code: https://github.com/facebookresearch/moco-v3

Abstract

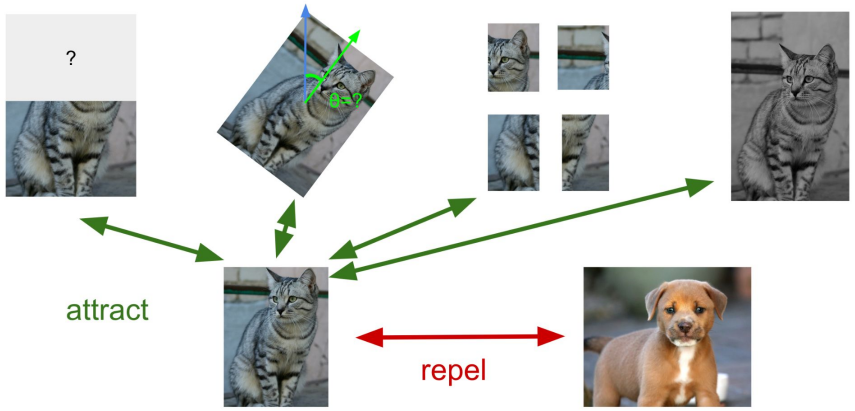
This paper does not describe a novel method. Instead, it studies a straightforward, incremental, yet must-know baseline given the recent progress in computer vision: self-supervised learning for Vision Transformers (ViT). While the training recipes for standard convolutional networks have been highly mature and robust, the recipes for ViT are yet to be built, especially in the self-supervised scenarios where training becomes more challenging. In this work, we go back to basics and investigate the effects of several fundamental components for training self-supervised ViT. We observe that instability is a major issue that degrades accuracy, and it can be hidden by apparently good results. We reveal that these results are indeed partial failure, and they can be improved when training is made more stable. We benchmark ViT results in MoCo v3 and several other self-supervised frameworks, with ablations in various aspects. We discuss the currently positive evidence as well as challenges and open questions. We hope that this work will provide useful data points and experience for future research.

framework	model	params	acc. (%)
linear probing:			
iGPT [9]	iGPT-L	1362M	69.0
iGPT [9]	iGPT-XL	6801M	72.0
MoCo v3	ViT-B	86M	76.7
MoCo v3	ViT-L	304M	77.6
MoCo v3	ViT-H	632M	78.1
MoCo v3	ViT-BN-H	632M	79.1
MoCo v3	ViT-BN-L/7	304M	**81.0**
end-to-end fine-tuning:			
masked patch pred. [16]	ViT-B	86M	79.9†
MoCo v3	ViT-B	86M	83.2
MoCo v3	ViT-L	304M	**84.1**

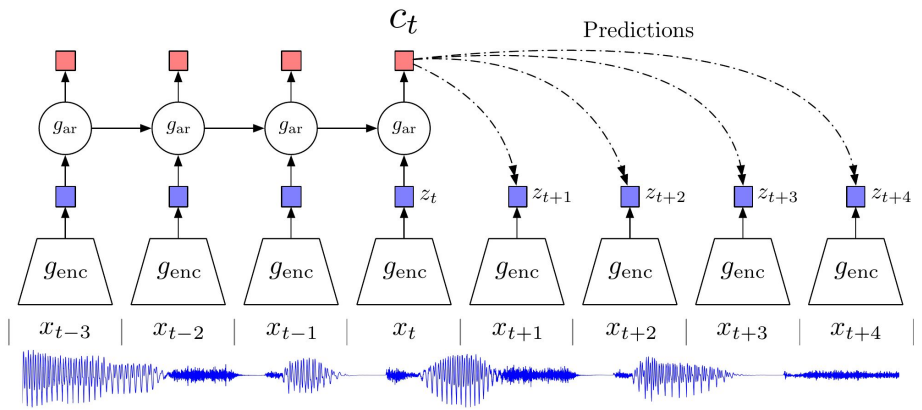
Table 1. **State-of-the-art Self-supervised Transformers** in ImageNet classification, evaluated by linear probing (top panel) or end-to-end fine-tuning (bottom panel). Both iGPT [9] and masked patch prediction [16] belong to the masked auto-encoding paradigm. MoCo v3 is a contrastive learning method that compares two (224×224) crops. ViT-B, -L, -H are the Vision Transformers proposed in [16]. ViT-BN is modified with BatchNorm, and "/7" denotes a patch size of 7×7. †: pre-trained in JFT-300M.

Chen et al. ICCV 2021

From Instance to Sequence …



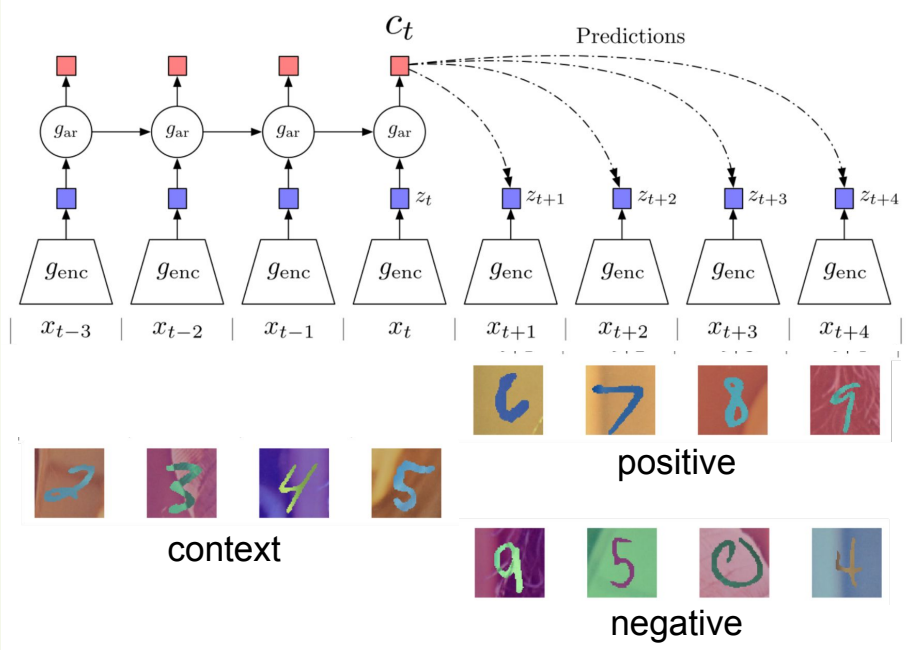
Instance-level contrastive learning:
contrastive learning based on
positive & negative instances.
Examples: SimCLR, MoCo



Source: van den Oord et al., 2018

Sequence-level contrastive learning:
contrastive learning based on
sequential / temporal orders.
Example: **Contrastive Predictive Coding (CPC)**

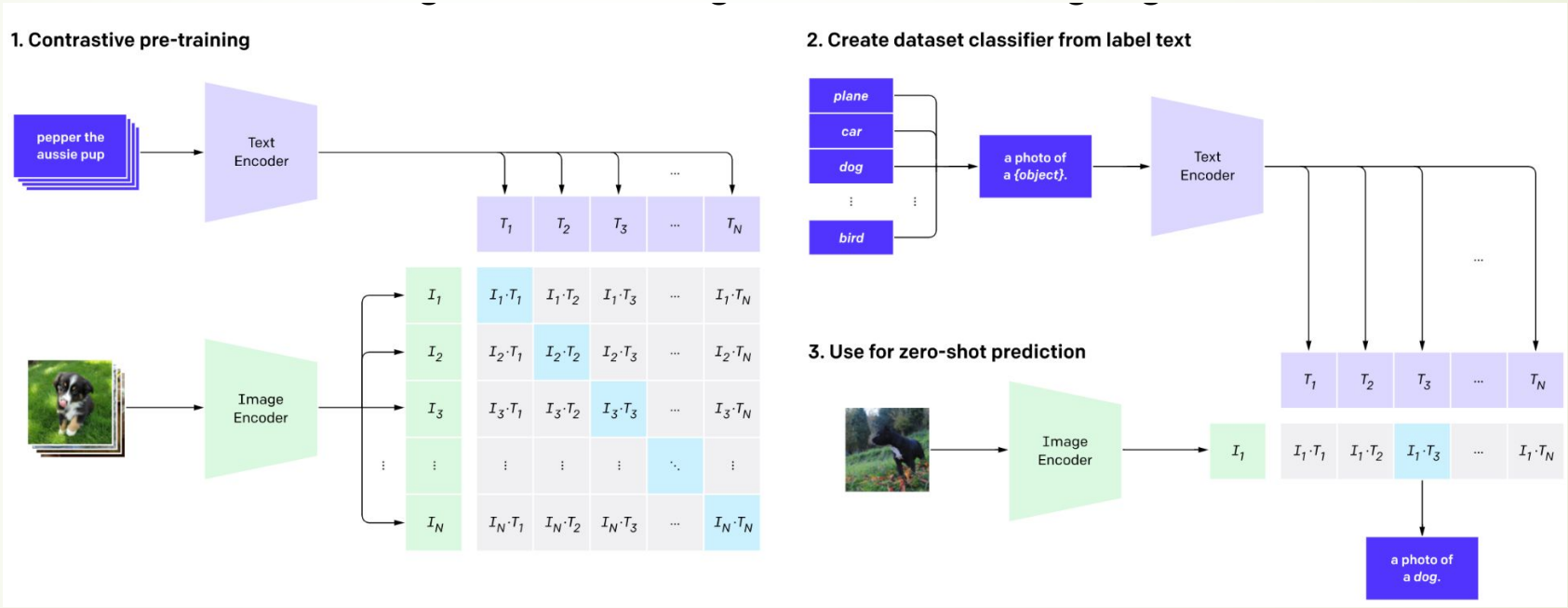
Contrastive Predictive Coding (CPC)



- Contrastive: contrast between "right" and "wrong" sequences using contrastive learning.
- Predictive: the model has to predict future patterns given the current context.
- Coding: the model learns useful feature vectors, or "code", for downstream tasks, similar to other self-supervised methods.
- Can be applied to a variety of learning problems, but not as effective in learning image representations compared to instance-level methods.

- Source: van den Oord et al. 2018 arXiv:1807.03748

Contrastive Language–Image Pre-training (CLIP)



Contrastive learning between image and natural language sentences, enables zero-shot learning.

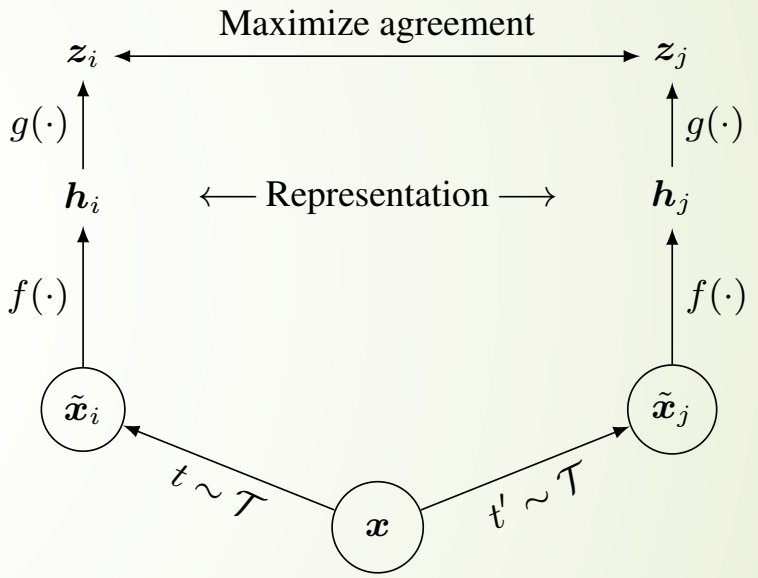
Radford et al., 2021

A Short History of Contrastive Representation Learning

- Common loss functions:
  - Contrastive loss (Chopra et al. 2005)
  - Triplet loss (Schroff et al. 2015; FaceNet)
  - Lifted structured loss (Song et al. 2015)
  - Multi-class n-pair loss (Sohn 2016)
  - Noise contrastive estimation ("NCE"; Gutmann & Hyvarinen 2010)
  - InfoNCE (van den Oord, et al. 2018)
  - Soft-nearest neighbors loss (Salakhutdinov & Hinton 2007, Frosst et al. 2019)

Summary: SimCLR

- **SimCLR**: a simple framework for contrastive representation learning
- **Key ideas**: non-linear projection head to allow flexible representation learning
- Simple to implement, effective in learning visual representation
- Requires large training batch size to be effective; large memory footprint



Source: Chen et al., 2020

Summary: MoCo

- **MoCo** (v1, v2): contrastive learning using momentum sample encoder
- Decouples negative sample size from minibatch size; allows large batch training without TPU
- MoCo-v2 combines the key ideas from SimCLR, i.e., nonlinear projection head, strong data augmentation, with momentum contrastive learning

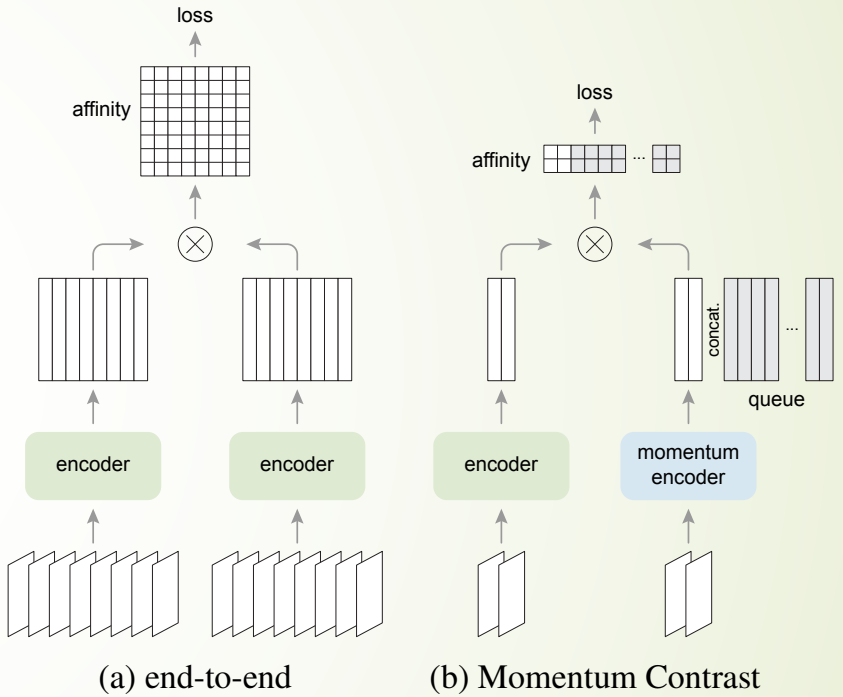


Figure 1. A **batching** perspective of two optimization mechanisms for contrastive learning. Images are encoded into a representation space, in which pairwise affinities are computed.

He et al., 2020; Chen et al., 2020

Thank you!