Self Supervised Learning

Naeemullah Khan
naeemullah.khan@kaust.edu.sa

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

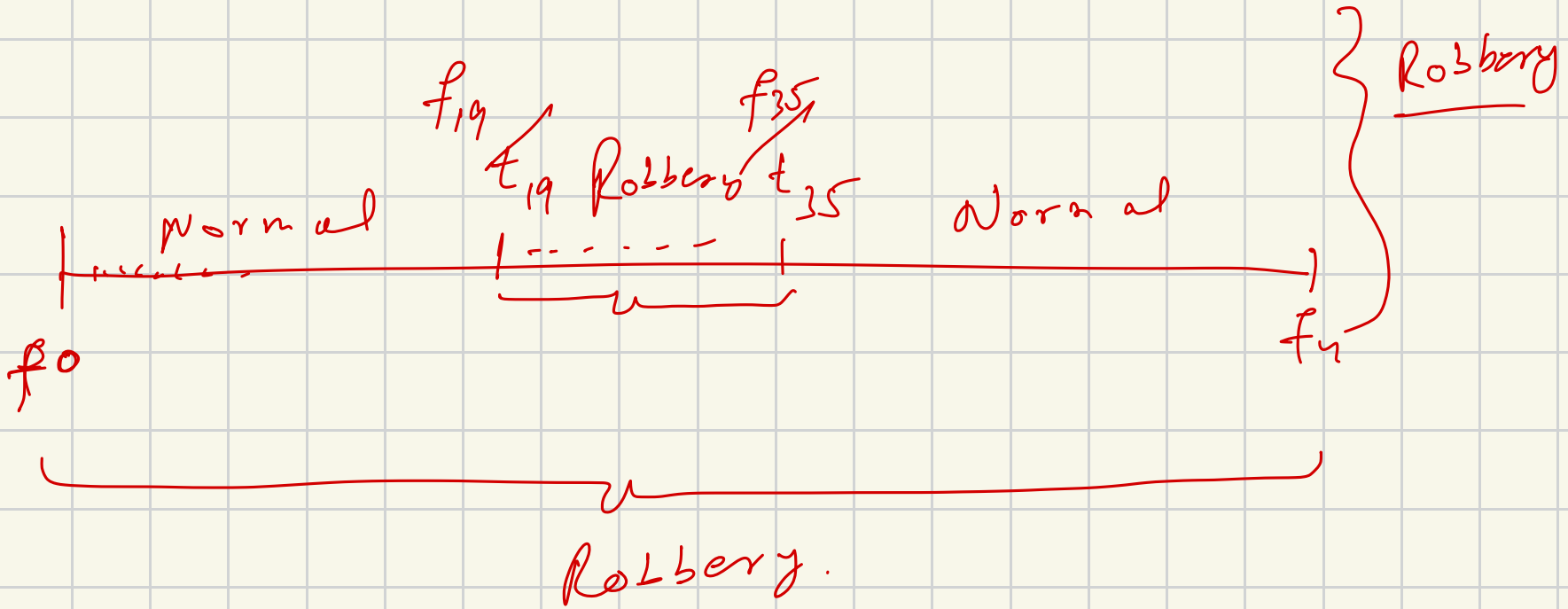
KAUST Academy
King Abdullah University of Science and Technology

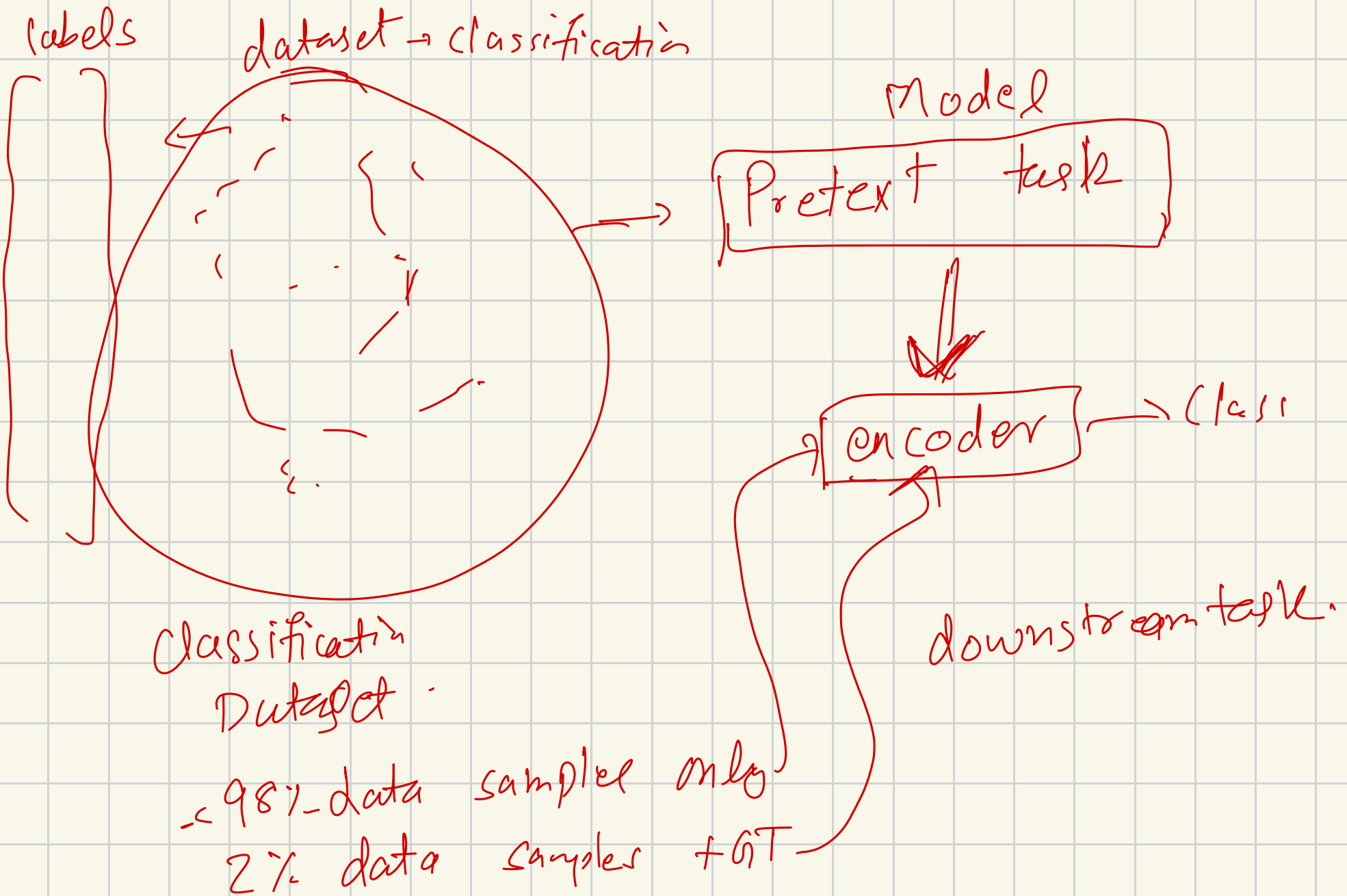
June 12, 2025

Table of Contents

Table of Contents (cont.)

semi-supervised learning.




labels
dataset → classification
model
Pretext task
encoder
Classi
classification
Dataset.
- 98% data sampled ml
2% data samples + GT
downstream task.

Motivation for Self-Supervised Learning

I am playing football

1. **Imagine you have tons of data, but very few labels.**
   Annotating data is expensive and slow, but unlabeled data is everywhere!

Motivation for Self-Supervised Learning

1. **Imagine you have tons of data, but very few labels.**
   Annotating data is expensive and slow, but unlabeled data is everywhere!
2. **What if you could learn useful features from all that unlabeled data?**
   Self-supervised learning lets us pre-train models to extract general, reusable representations, reducing the need for labeled data later.

Motivation for Self-Supervised Learning

1. **Imagine you have tons of data, but very few labels.**
   Annotating data is expensive and slow, but unlabeled data is everywhere!
2. **What if you could learn useful features from all that unlabeled data?**
   Self-supervised learning lets us pre-train models to extract general, reusable representations, reducing the need for labeled data later.
3. **Think beyond images:**
   These techniques work not just for vision, but also for language, audio, and time-series data, enabling unified approaches across domains.

Motivation for Self-Supervised Learning

1. **Imagine you have tons of data, but very few labels.**
   Annotating data is expensive and slow, but unlabeled data is everywhere!
2. **What if you could learn useful features from all that unlabeled data?**
   Self-supervised learning lets us pre-train models to extract general, reusable representations, reducing the need for labeled data later.
3. **Think beyond images:**
   These techniques work not just for vision, but also for language, audio, and time-series data, enabling unified approaches across domains.
4. **Inspired by how humans learn:**
   We learn patterns from the world around us without explicit labels—self-supervised learning mimics this natural process.

Motivation for Self-Supervised Learning

1. **Imagine you have tons of data, but very few labels.**
   Annotating data is expensive and slow, but unlabeled data is everywhere!
2. **What if you could learn useful features from all that unlabeled data?**
   Self-supervised learning lets us pre-train models to extract general, reusable representations, reducing the need for labeled data later.
3. **Think beyond images:**
   These techniques work not just for vision, but also for language, audio, and time-series data, enabling unified approaches across domains.
4. **Inspired by how humans learn:**
   We learn patterns from the world around us without explicit labels—self-supervised learning mimics this natural process.
5. **A journey of methods:**
   The field has evolved from generative models (like VAEs and GANs), to contrastive and predictive coding, and now to hybrid approaches combining reconstruction and discrimination.

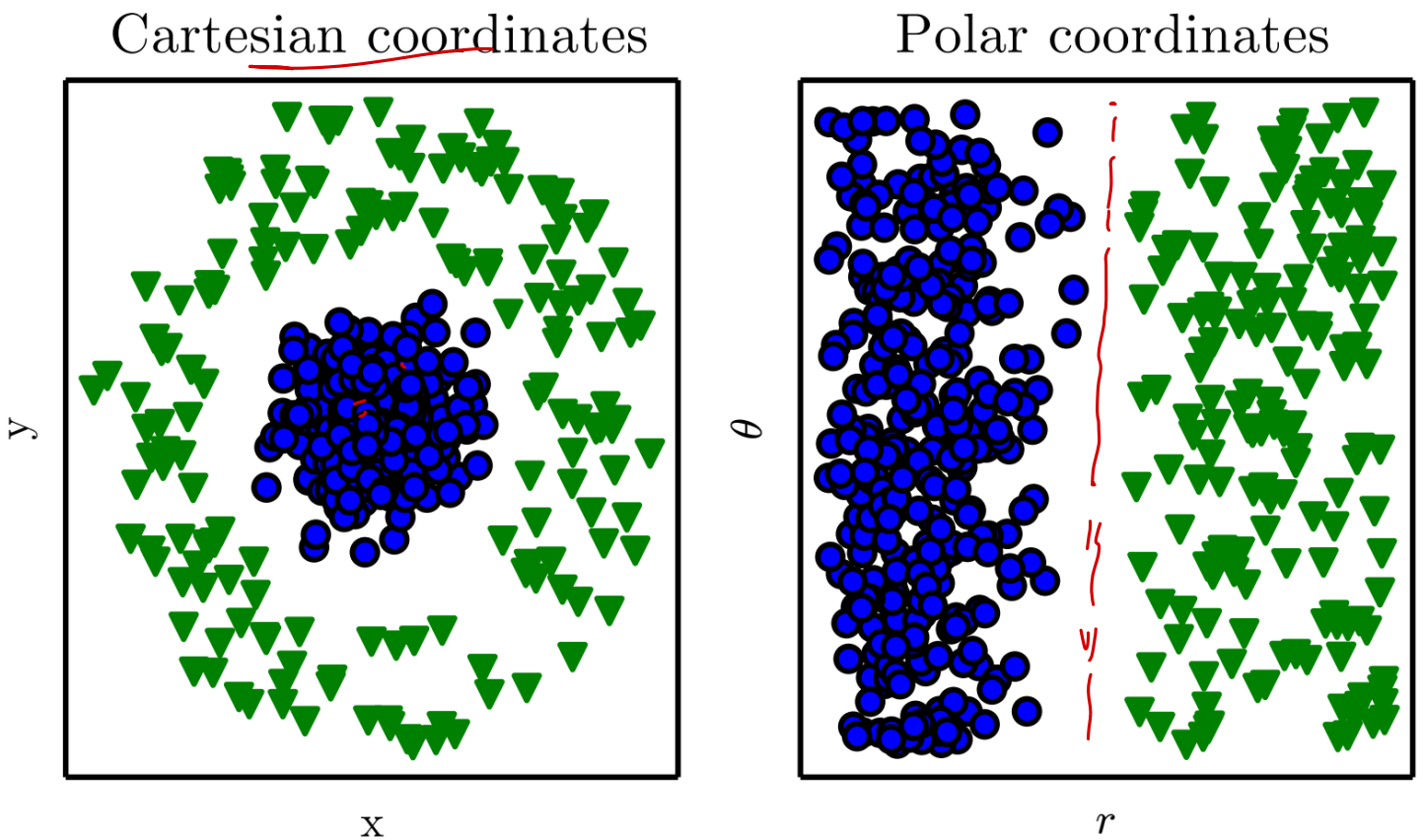
Learning Outcomes

By the end of this section, you should be able to:

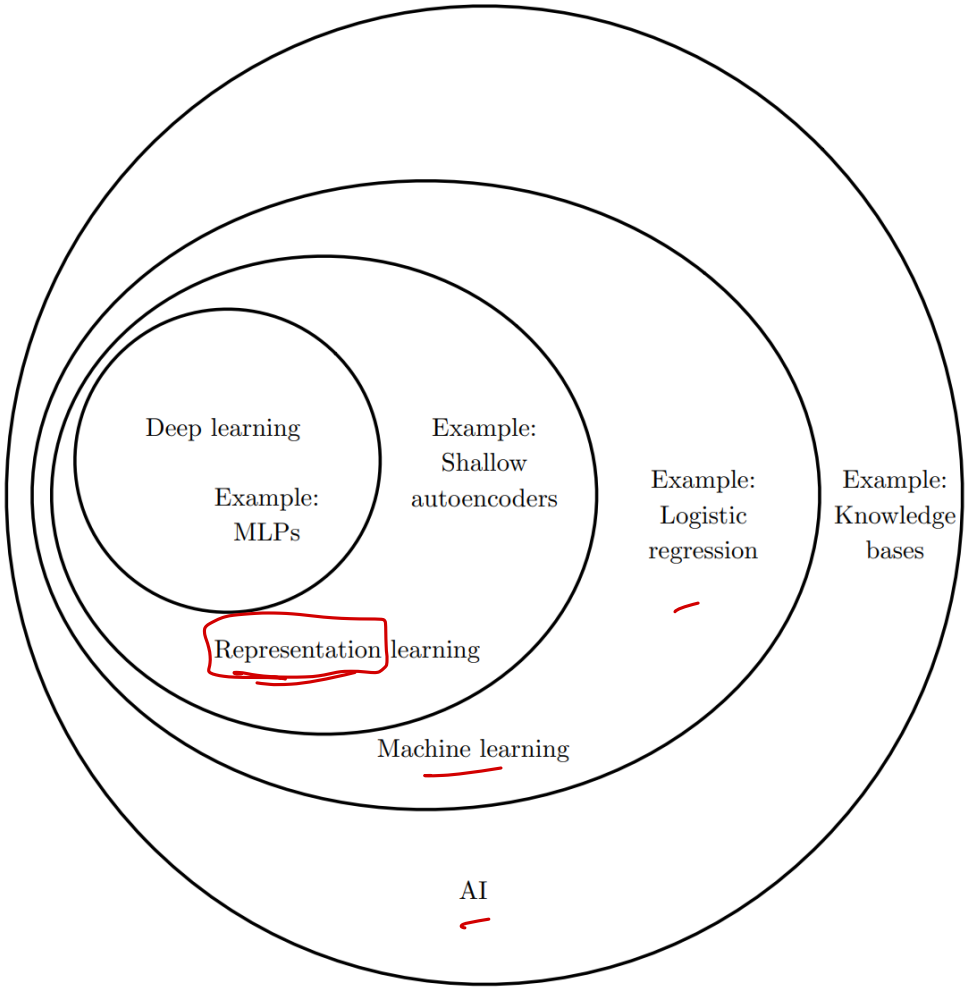
- **Explain** the data efficiency imperative and theoretical motivations behind self-supervised learning (SSL).
- **Identify** key cognitive principles that inspire self-supervision in machine learning.
- **Compare** reconstructive, predictive, and contrastive pretext tasks used in SSL.
- **Implement and evaluate** Denoising Autoencoders and Context Encoders for representation learning.
- **Formulate** the InfoNCE loss and **train** Contrastive Predictive Coding (CPC) models.
- **Describe** Instance Discrimination and **implement** the Momentum Contrast (MoCo) framework.
- **Critically assess** pretext tasks such as rotation prediction, relative patch prediction, and view prediction.

Data → Model → Objective → Optimize

Representations Matter



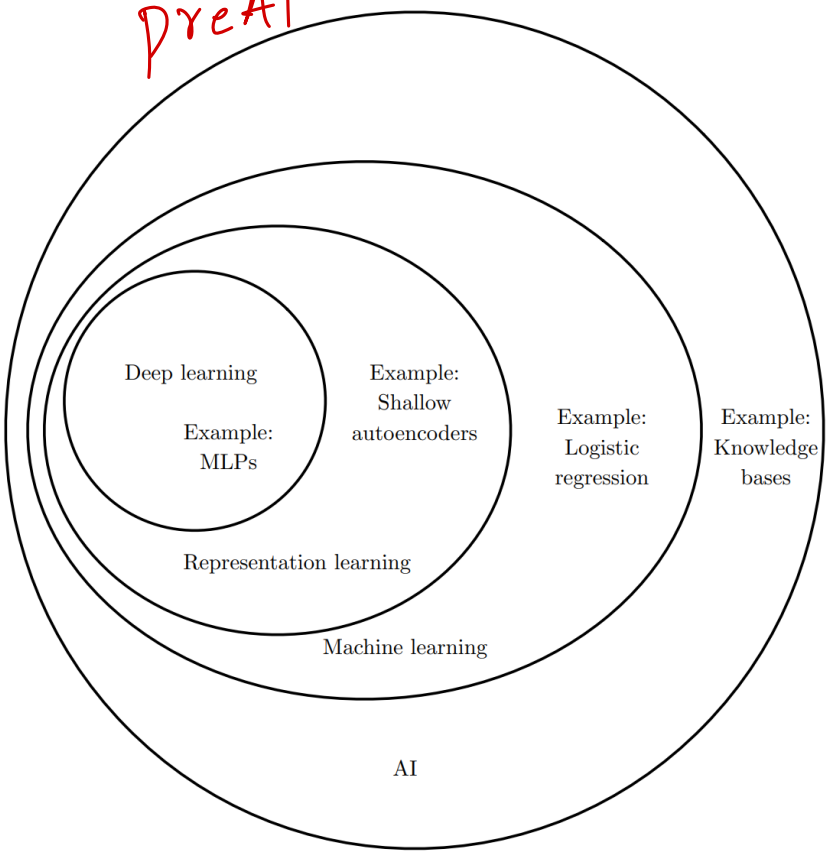
Introduction (cont.)



Deep Unsupervised Learning:

- Learns data representations without using labels.
- Is a subset of Deep Learning, which itself is a subset of Representation Learning, all within Machine Learning.

Introduction (cont.)



Deep Unsupervised Learning:

- Learns data representations without using labels.
- Is a subset of Deep Learning, which itself is a subset of Representation Learning, all within Machine Learning.

Self-Supervised Learning (SSL):

- A form of unsupervised learning that creates supervision from the data itself by designing *pretext tasks*.
- These tasks generate pseudo-labels, enabling the model to learn useful representations without manual annotation.
- Often used interchangeably with unsupervised learning, but SSL specifically focuses on leveraging intrinsic data structure.

Why Self-Supervised Learning?

- **High cost of dataset creation:** Each new task often requires building a new labeled dataset.
  - Involves preparing labeling manuals, defining categories, hiring annotators, building GUIs, and managing storage pipelines.
- **Expensive supervision:** High-quality labels can be costly or impractical to obtain (e.g., in medicine or legal domains).
- **Leverage unlabeled data:** The internet provides vast amounts of unlabeled images, videos, and text that can be utilized.
- **Cognitive motivation:** Animals and babies learn from their environment without explicit supervision, inspiring SSL approaches.

Why Self-Supervised Learning? (cont.)

"Give a robot a label and you feed it for a moment; teach a robot to label and you feed it for a lifetime."

— Pierre Sermanet

This quote highlights the core motivation behind self-supervised learning
→ enabling machines to generate their own supervision from data
→ thus reducing reliance on costly manual labeling and
→ fostering scalable, autonomous learning.

What is Self-Supervised Learning?

SSL is a type of unsupervised learning where the data itself provides supervision. Typically, part of the data is hidden, and a neural network is trained to predict it from the rest—this defines the *pretext task*. Well-chosen pretext tasks help models learn useful features without manual labels.

- **Pretext Task**: Examples include predicting masked patches, future representations (CPC), or distinguishing positive/negative pairs (contrastive).
- **Downstream Task**: After pre-training, the encoder is fine-tuned for tasks like classification or detection.
- **Key Trade-Offs**:
  - **Reconstructive** (e.g., autoencoders): capture low-level details, may miss semantics.
  - **Predictive/Contrastive** (e.g., CPC, MoCo): focus on high-level information.

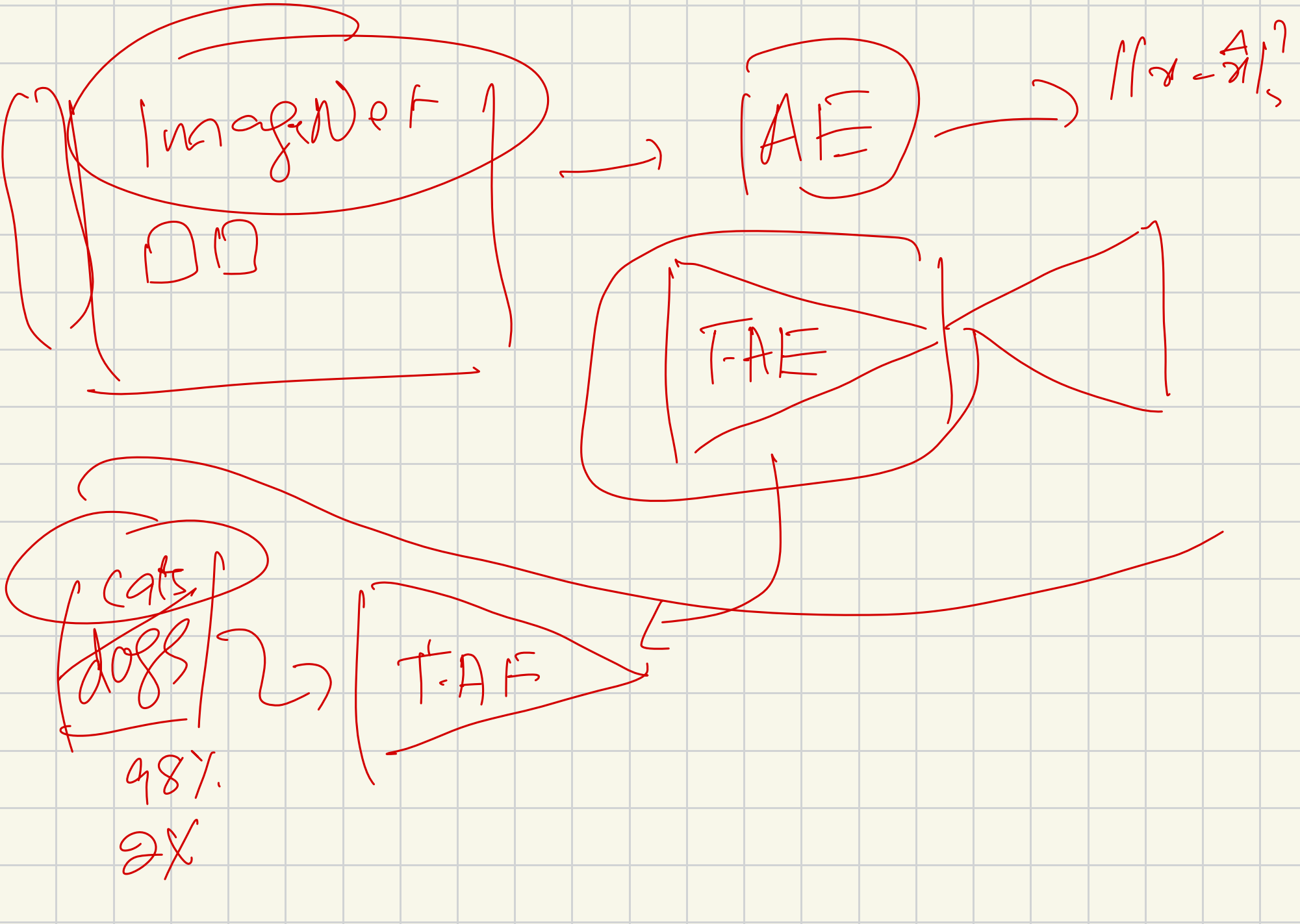


Image
AE
$\|x - \hat{x}\|_2^2$
T-AE
cats
dogs
T-AE
98%
2%

What is Self-Supervised Learning? (cont.)

- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Self-Supervised Learning (cake génoise)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

Cognitive Principles

- **Predictive Coding**: The brain optimizes to predict future sensory inputs.
- **Error Correction**: Learning signals arise from reconstructive or contrastive errors.
  - *Reconstruction from Corrupted Inputs*:
    - Denoising autoencoders
    - Inpainting
    - Colorization, split-brain autoencoders
  - *Visual Common Sense Tasks*:
    - Relative patch prediction
    - Jigsaw puzzles
    - Rotation prediction
  - *Contrastive Learning*:
    - word2vec

Cognitive Principles (cont.)

- Contrastive Predictive Coding (CPC)
- Instance discrimination
- Recent state-of-the-art methods

- **Multi-View Perception**: Integrating different sensory views enhances representations.

Pretext Task Taxonomy: Introduction

Pretext tasks are broadly categorized by their supervision signal and learning objective:

- **Reconstructive Methods**: Focus on reconstructing parts or properties of the input data.
- **Predictive Methods**: Involve predicting withheld or transformed aspects of the data.

Each category includes various techniques that have advanced SSL in computer vision.

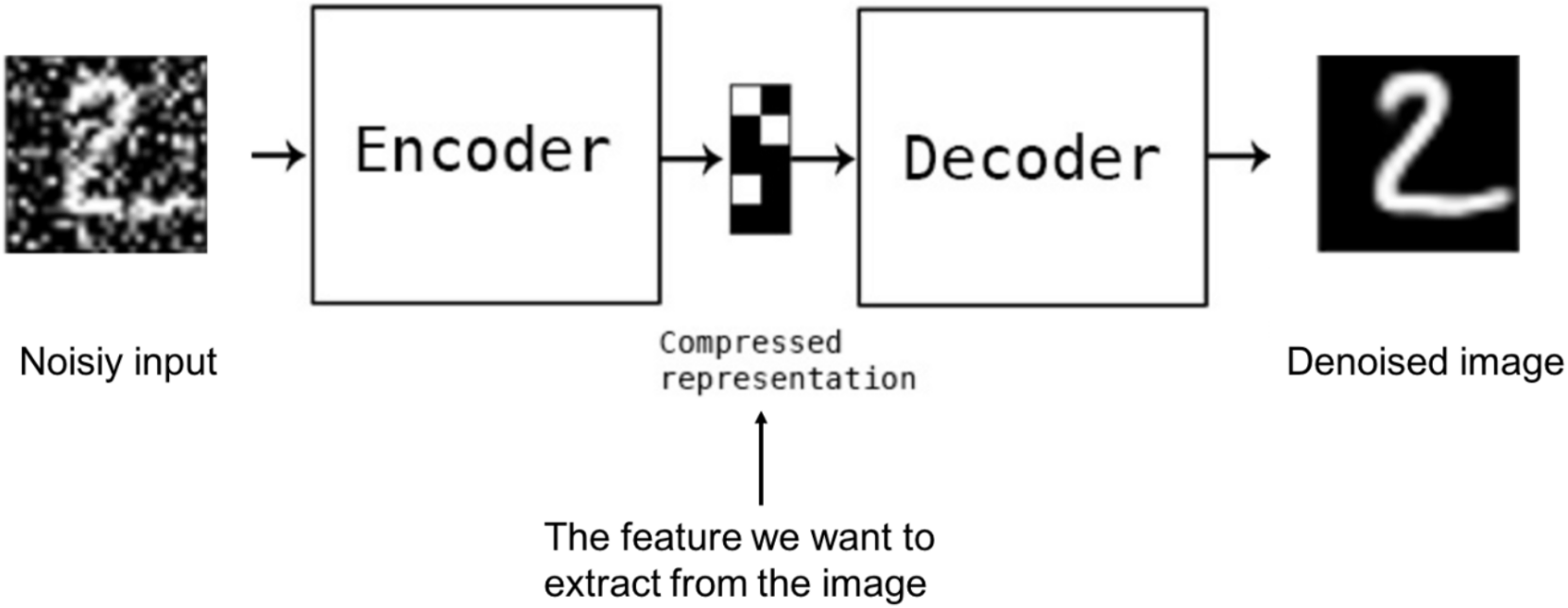
Understanding this taxonomy helps in:

- Selecting suitable pretext tasks for specific applications.
- Gaining insight into how SSL methods learn robust and transferable representations.

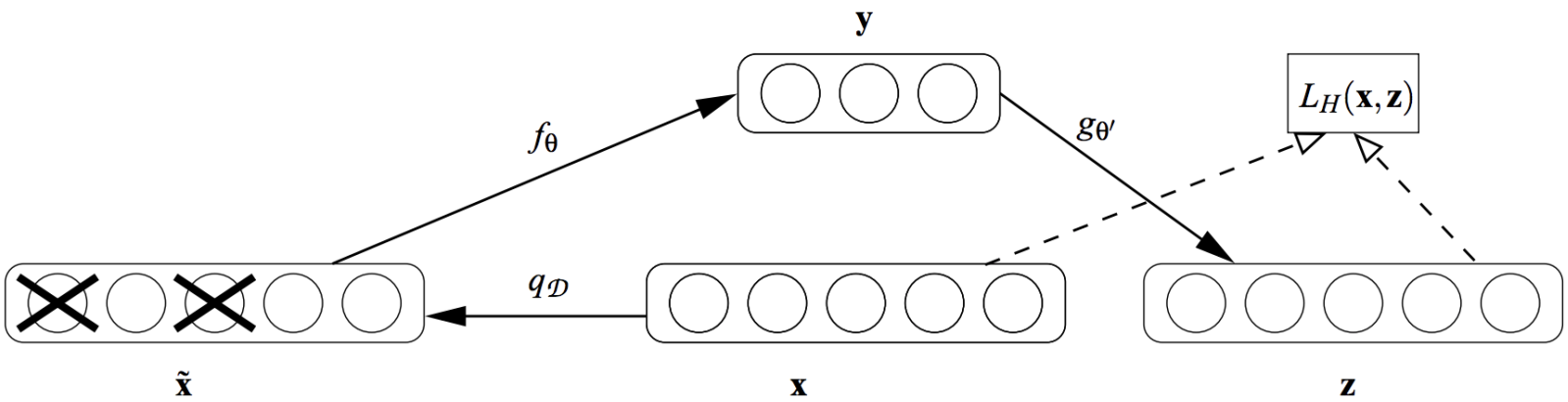
Denoising Autoencoders

- **Corrupt input:** Introduce random noise to the input data, such as masking or adding Gaussian noise. The autoencoder is then trained to reconstruct the original, clean input from the corrupted version.
- **Robust feature learning:** By learning to recover the original data, the autoencoder is encouraged to extract meaningful and robust features that are less sensitive to noise and irrelevant variations.
- **Incremental understanding:** This approach helps the model generalize better by preventing it from simply memorizing the input, thus improving its ability to handle unseen or noisy data.

Denoising Autoencoders (cont.)

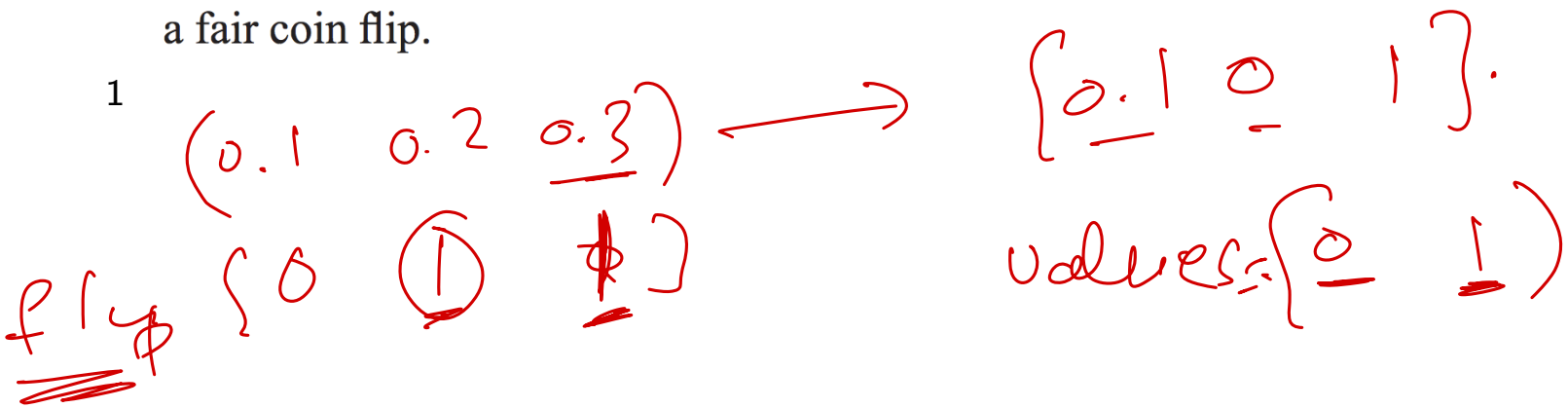


Denoising Autoencoders (cont.)



Denoising Autoencoders (cont.)

- Additive isotropic *Gaussian noise* (GS): $\tilde{\mathbf{x}}|\mathbf{x} \sim \mathcal{N}(\mathbf{x}, \sigma^2 I)$;
- *Masking noise* (MN): a fraction ν of the elements of $\mathbf{x}$ (chosen at random for each example) is forced to 0;
- *Salt-and-pepper noise* (SP): a fraction ν of the elements of $\mathbf{x}$ (chosen at random for each example) is set to their minimum or maximum possible value (typically 0 or 1) according to a fair coin flip.



[1] Vincent et al (2010). Denoising Autoencoders: Unsupervised Learning of Image Features from Noisy Data. ICML 2010.

Emphasizing corrupted dimensions

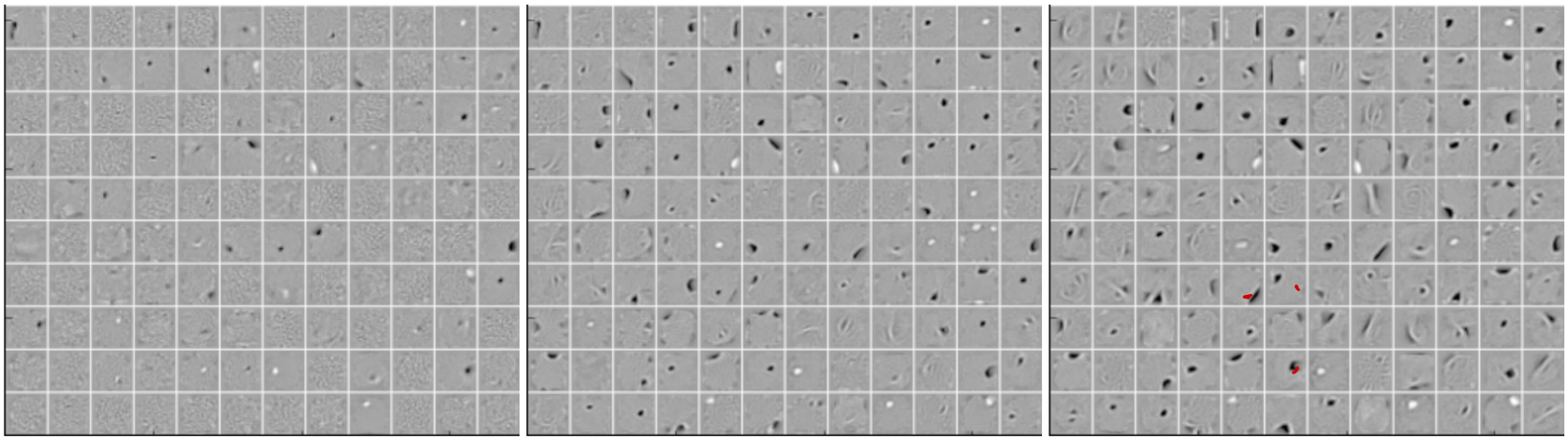
$$L_{2,\alpha}(\mathbf{x},\mathbf{z}) = \alpha\left(\sum_{j\in\mathcal{I}(\tilde{\mathbf{x}})}(\mathbf{x}_j-\mathbf{z}_j)^2\right)+\beta\left(\sum_{j\notin\mathcal{I}(\tilde{\mathbf{x}})}(\mathbf{x}_j-\mathbf{z}_j)^2\right)$$

$$L_{\mathrm{IH},\alpha}(\mathbf{x},\mathbf{z}) = \alpha\left(-\sum_{j\in\mathcal{I}(\tilde{\mathbf{x}})}[\mathbf{x}_j\log\mathbf{z}_j+(1-\mathbf{x}_j)\log(1-\mathbf{z}_j)]\right)$$
$$+\beta\left(-\sum_{j\notin\mathcal{I}(\tilde{\mathbf{x}})}[\mathbf{x}_j\log\mathbf{z}_j+(1-\mathbf{x}_j)\log(1-\mathbf{z}_j)]\right)$$

2

[2]Vincent et al (2010). Denoising Autoencoders: Unsupervised Learning of Image Features from Noisy Data. ICML 2010.

Denoising Autoencoders

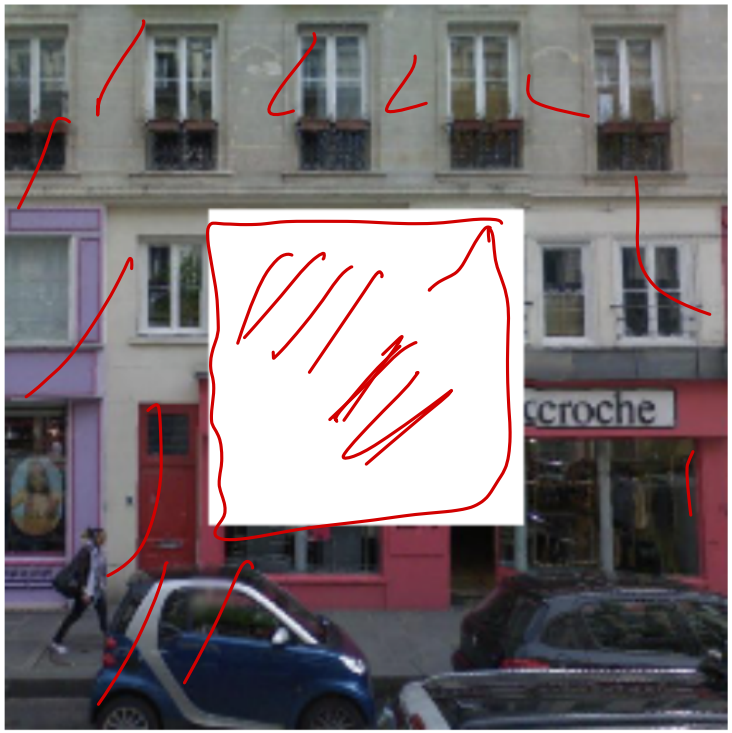


(a) No destroyed inputs (b) 25% destruction (c) 50% destruction

3

[3] Vincent et al (2010). Denoising Autoencoders: Unsupervised Learning of Image Features from Noisy Data. ICML 2010.

Predict missing pieces



4

[4] Pathak et al. (2016), "Context Encoders: Feature Learning by Inpainting," CVPR 2016.

Context Encoders

Context Encoders are designed to learn meaningful image representations by reconstructing missing parts of an image.

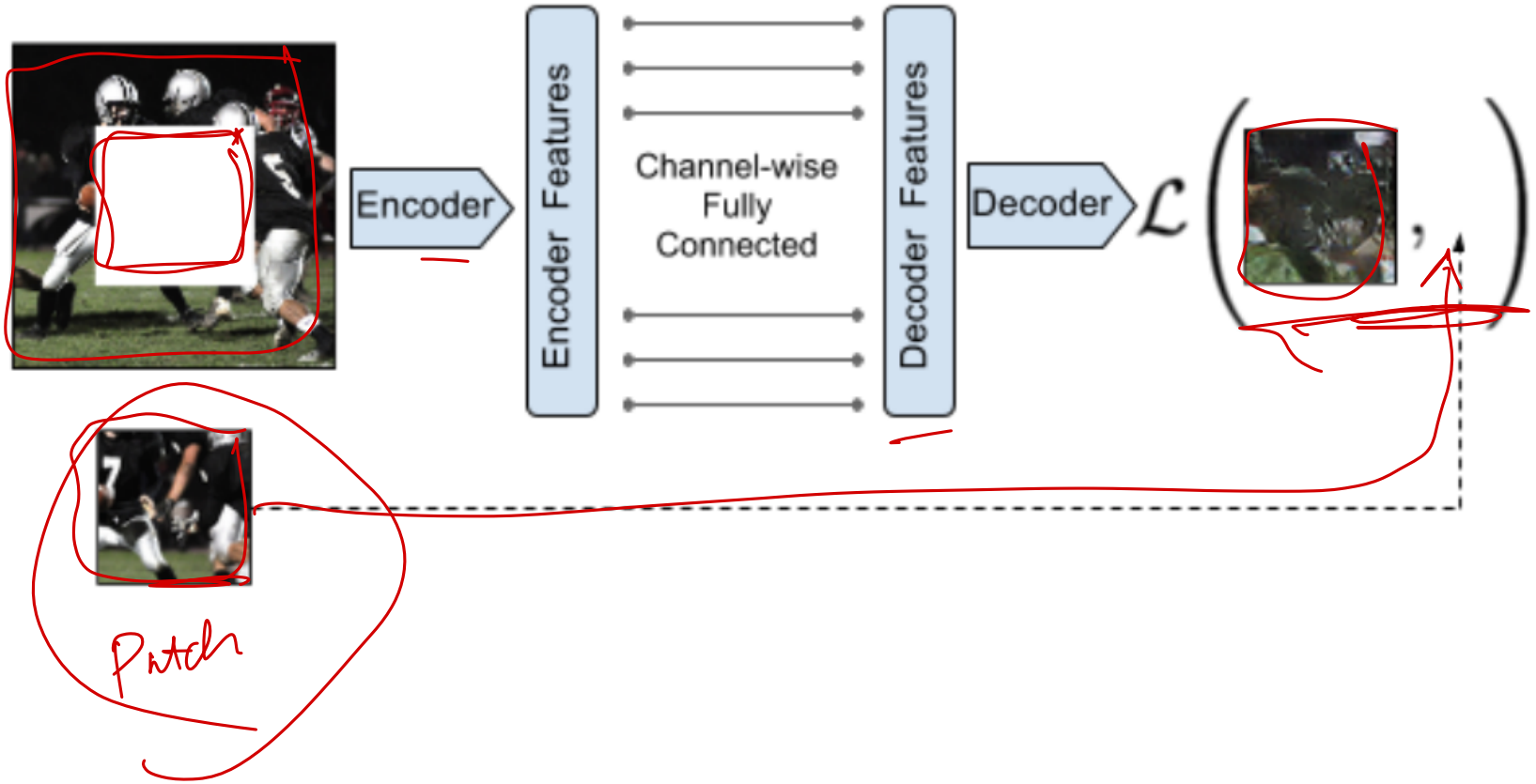
Key Idea:

- Randomly mask or remove patches from an input image.
- Use an encoder–decoder neural network to predict and reconstruct the missing content.
- The model is trained to minimize the difference between the predicted and actual image patches.

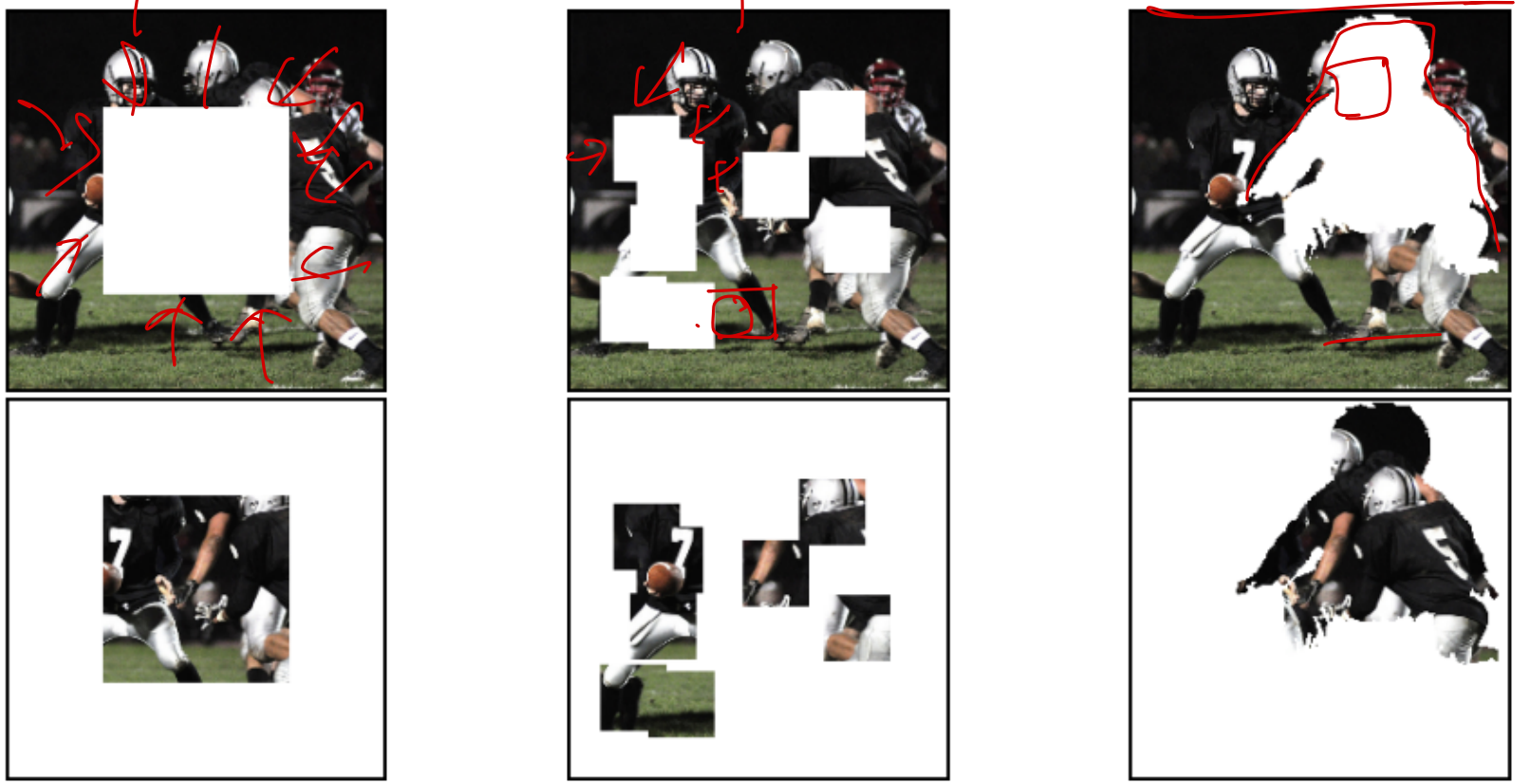
Architecture:

- **Encoder:** Processes the visible parts of the image and extracts feature representations.
- **Decoder:** Generates the missing image regions based on the encoded features.
- The entire network is trained end-to-end using a reconstruction loss (e.g., L2 loss).

Context Encoders (cont.)



Context Encoders (cont.)



(a) Central region (b) Random block (c) Random region

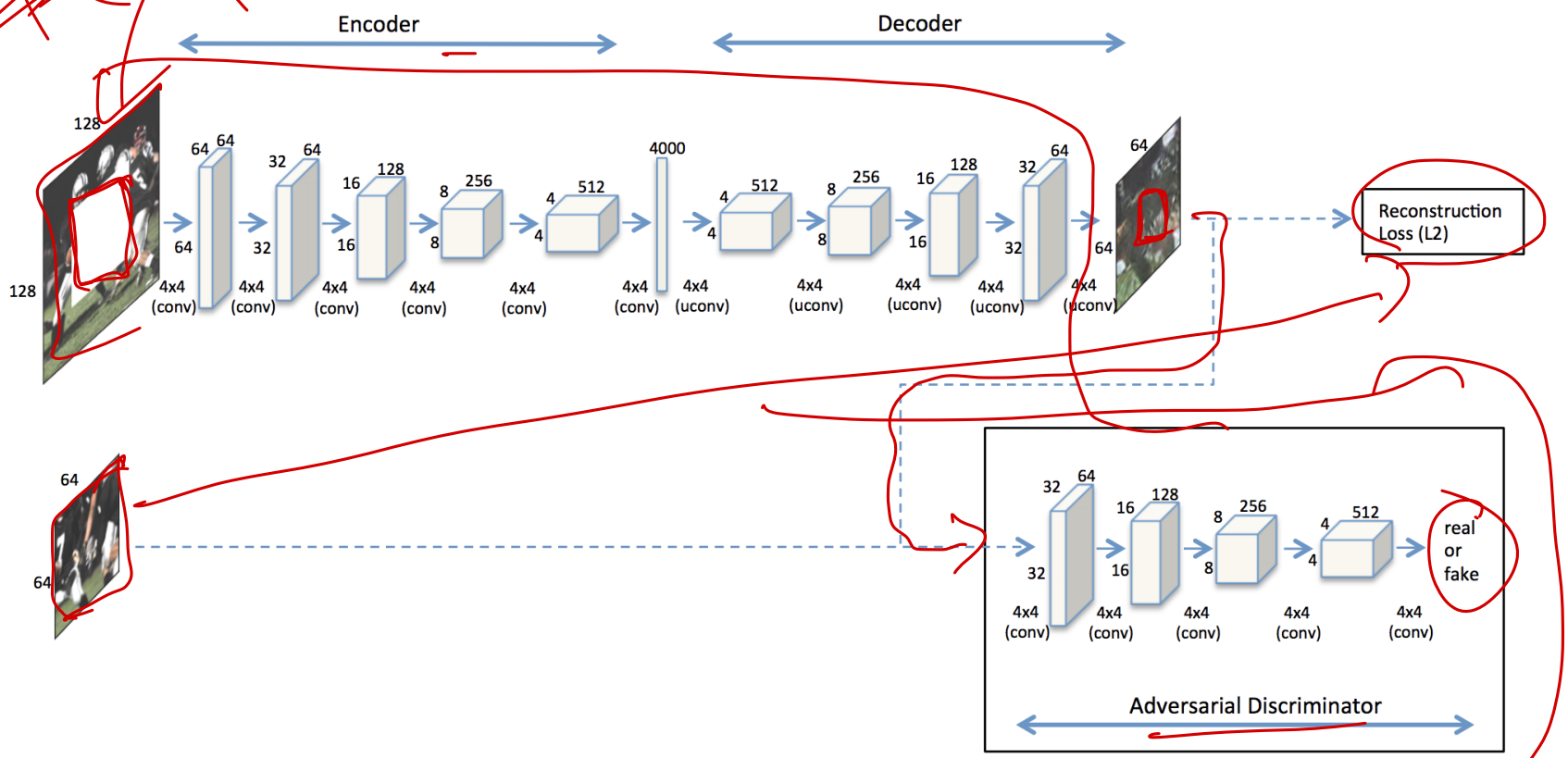
Context Encoders (cont.)

$$\mathcal{L}_{rec}(x) = \|\hat{M} \odot (x - F((1 - \hat{M}) \odot x))\|_2^2$$

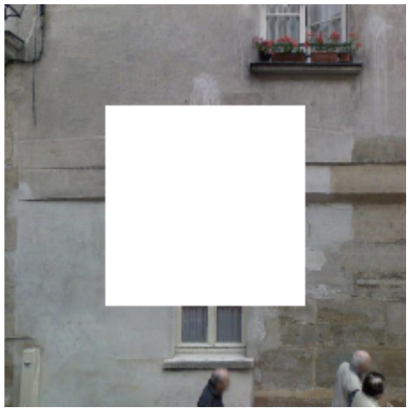
$$\mathcal{L}_{adv} = \max_D \quad \mathbb{E}_{x \in \mathcal{X}}[\log(D(x)) + \log(1 - D(F((1 - \hat{M}) \odot x)))]$$

$$\mathcal{L} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{adv}\mathcal{L}_{adv}$$

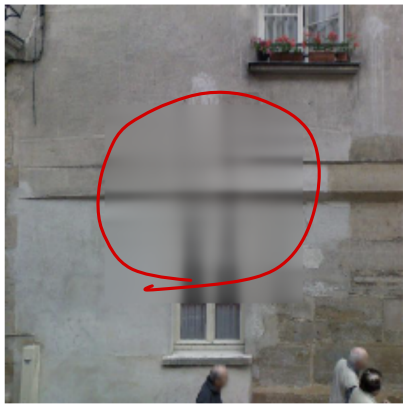
Context Encoders (cont.)



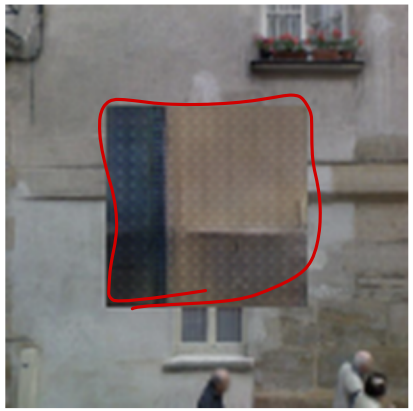
Context Encoders (cont.)



Input Image



L2 Loss



Adversarial Loss

Joint Loss

Context Encoders (cont.)

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%	-
Doersch *et al.* [7]	context	4 weeks	55.3%	**46.6%**	-
Wang *et al.* [39]	motion	1 week	**58.4%**	44.0%	
Ours	context	14 hours	56.5%	44.5%	**29.7%**

Table 2: Quantitative comparison for classification, detection and semantic segmentation. Classification and Fast-RCNN Detection results are on the PASCAL VOC 2007 test set. Semantic segmentation results are on the PASCAL VOC 2012 validation set from the FCN evaluation described in Section 5.2.3, using the additional training data from [18], and removing overlapping images from the validation set [28].

Context Encoders (cont.)

Why does this work?

- The network must understand the global and local context of the image to generate plausible content.
- This forces the model to learn semantic features and relationships within the image.
- The learned representations can be transferred to downstream tasks such as classification, detection, or segmentation.

Applications:

- Image inpainting
- Representation learning for transfer to other vision tasks
- Anomaly detection (by comparing predicted and actual content)

Reference:

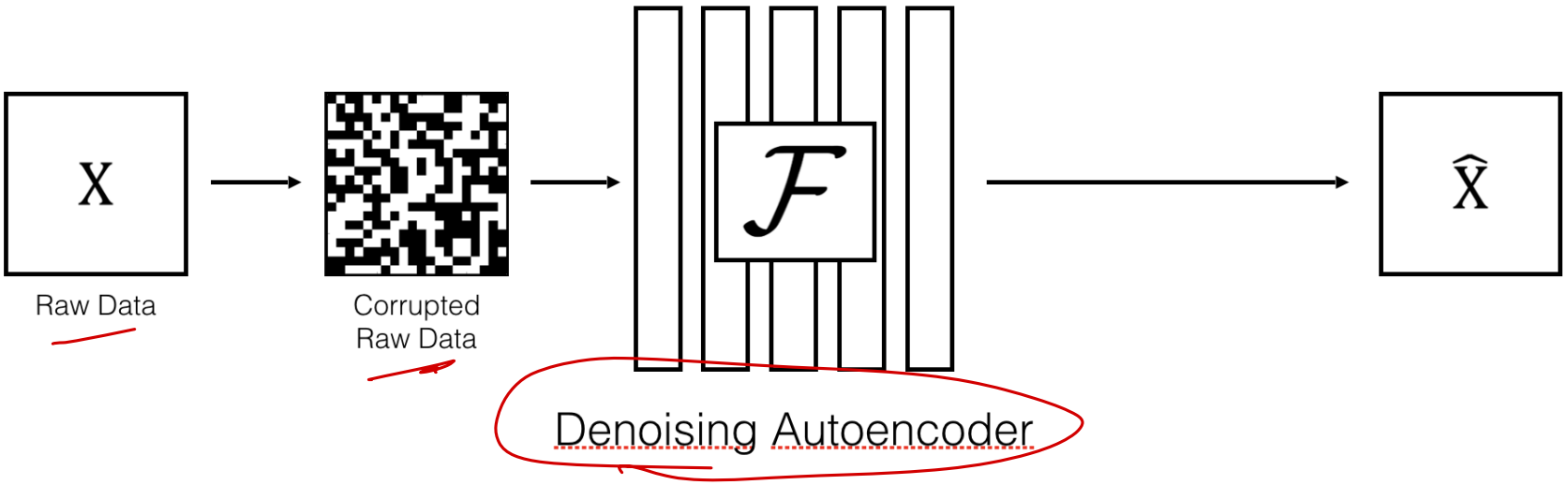
- Pathak et al., "Context Encoders: Feature Learning by Inpainting," CVPR 2016.

Predicting One View from Another

Learn to translate between modalities or augmentations

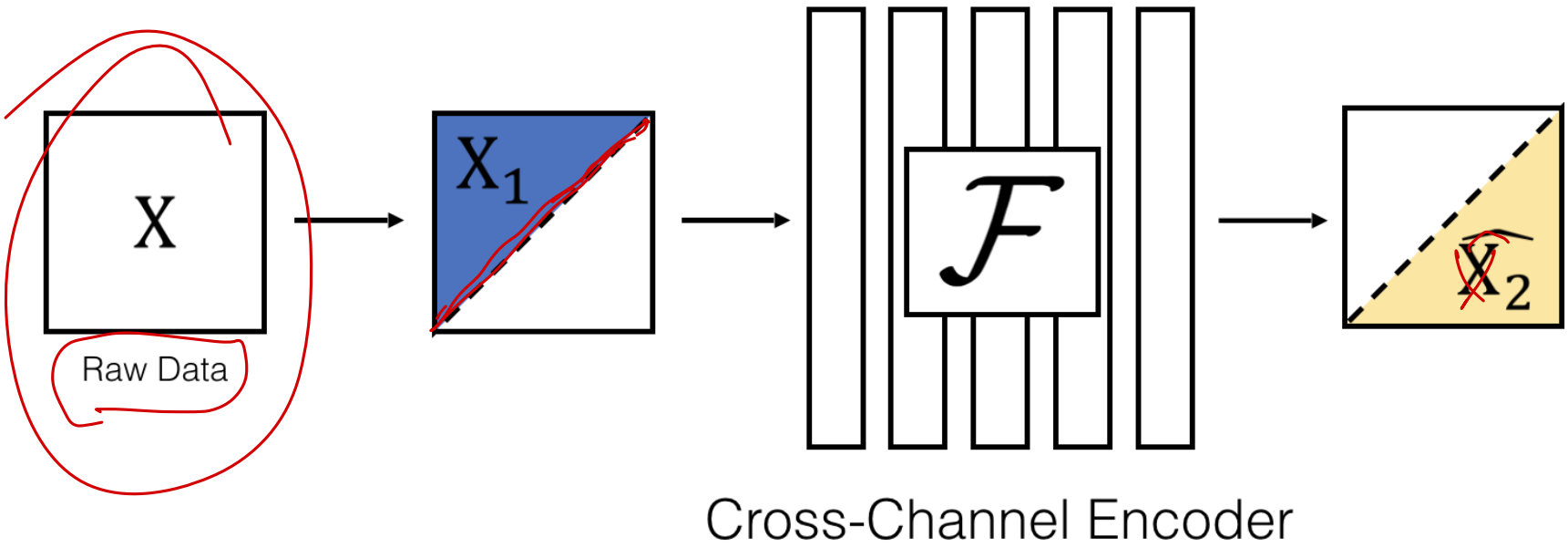
- The core idea is to train a model that can predict or reconstruct one view of the data given another.
- Views can be different modalities (e.g., infrared vs. RGB), or different augmentations (e.g., color vs. grayscale, rotated images, etc.).
- This approach encourages the model to learn shared representations that capture the underlying semantics of the data, rather than superficial features.

Predicting One View from Another (cont.)



Slide: Richard Zhang (2019). Predicting One View from Another.

Predicting One View from Another (cont.)



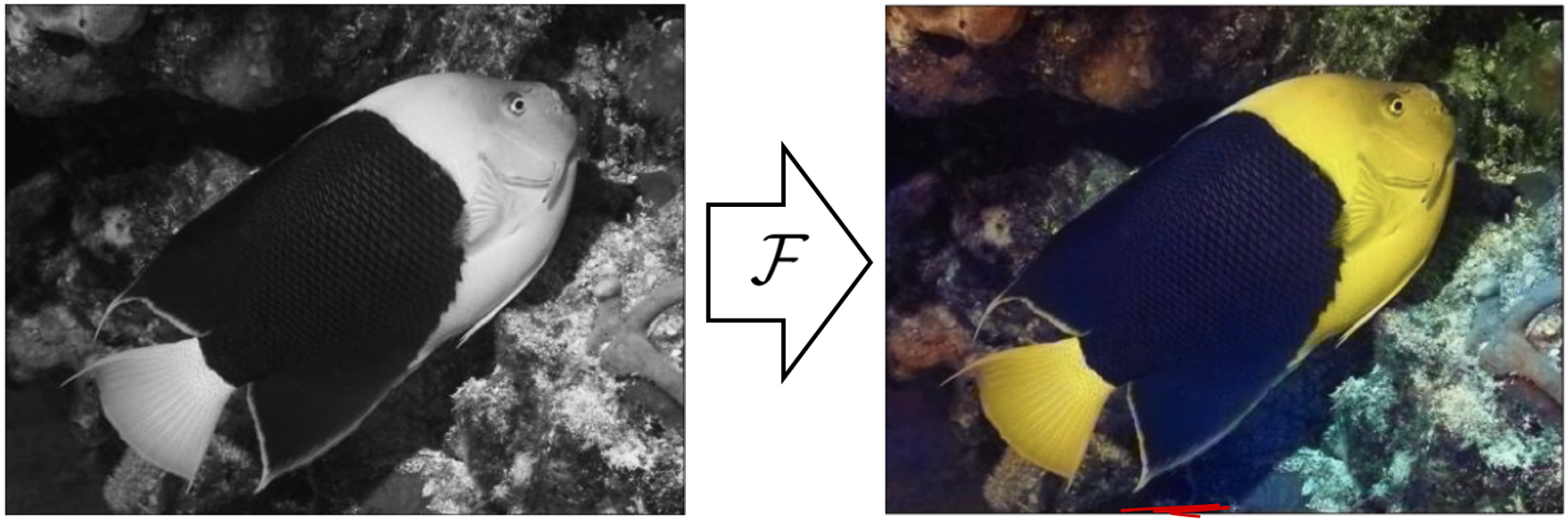
Slide: Richard Zhang (2019). Predicting One View from Another.

Predicting One View from Another (cont.)

Example: Learn mapping from infrared to RGB imagery

- In remote sensing, images are often captured in both infrared and RGB channels.
- A model can be trained to predict the RGB image given the infrared image, or vice versa.
- This task forces the model to understand the relationship between the two modalities, which can improve its generalization and robustness.
- Such cross-modal prediction is useful for applications where one modality may be missing or corrupted.

Predicting One View from Another (cont.)

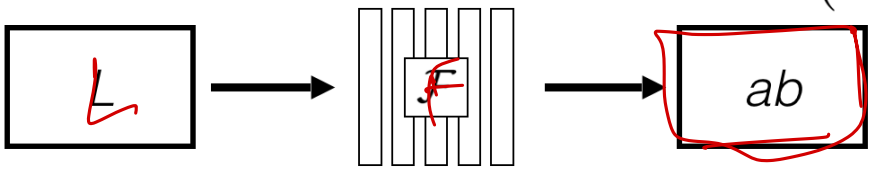


Grayscale image: L channel

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

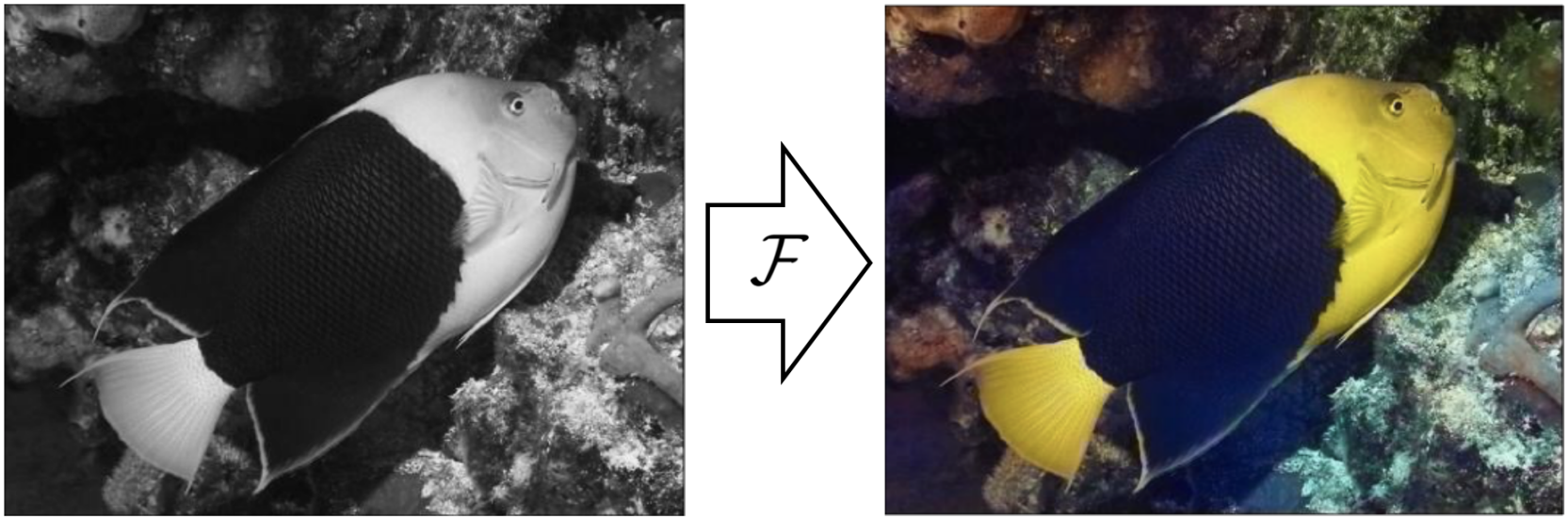
Concatenate (L,ab) channels

$$(\mathbf{X}, \widehat{\mathbf{Y}})$$



Slide: Richard Zhang (2019). Predicting One View from Another.

Predicting One View from Another (cont.)

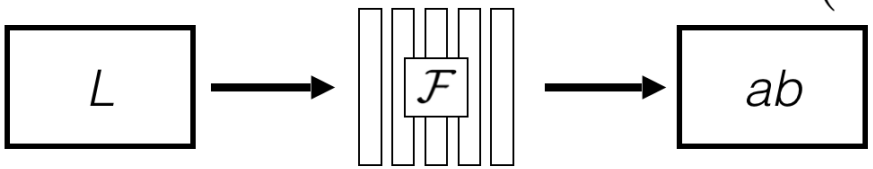


Grayscale image: L channel

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

Concatenate (L,ab) channels

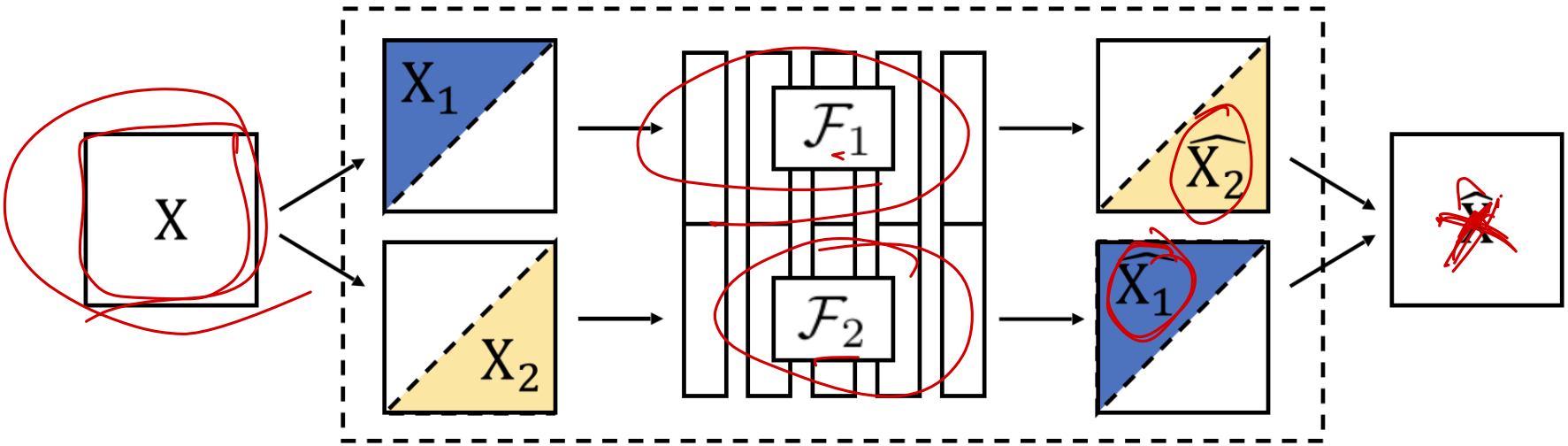
$(\mathbf{X}, \widehat{\mathbf{Y}})$



6

Slide: Richard Zhang (2019). Predicting One View from Another.

Predicting One View from Another (cont.)



Split-Brain Autoencoder

Slide: Richard Zhang (2019). Predicting One View from Another.

Predicting One View from Another (cont.)

- **Applications and Benefits**
  - **Data augmentation**: By learning to predict one augmentation from another, models become more invariant to transformations.
  - **Multi-modal learning**: Enables leveraging complementary information from different data sources.
  - **Self-supervised learning**: No need for manual labels, as the prediction task itself provides the supervision.
- **Challenges**
  - The mapping between modalities may be complex and non-linear.
  - Requires careful design of architectures and loss functions to ensure meaningful learning.

Relative Position of Image Patches

Unsupervised Visual Representation Learning by Context Prediction

Carl Doersch[1,2] Abhinav Gupta[1] Alexei A. Efros[2]

[1] School of Computer Science
Carnegie Mellon University

[2] Dept. of Electrical Engineering and Computer Science
University of California, Berkeley

Vasil?

Slide: Zisserman et al (2019). Relative Position of Image Patches.

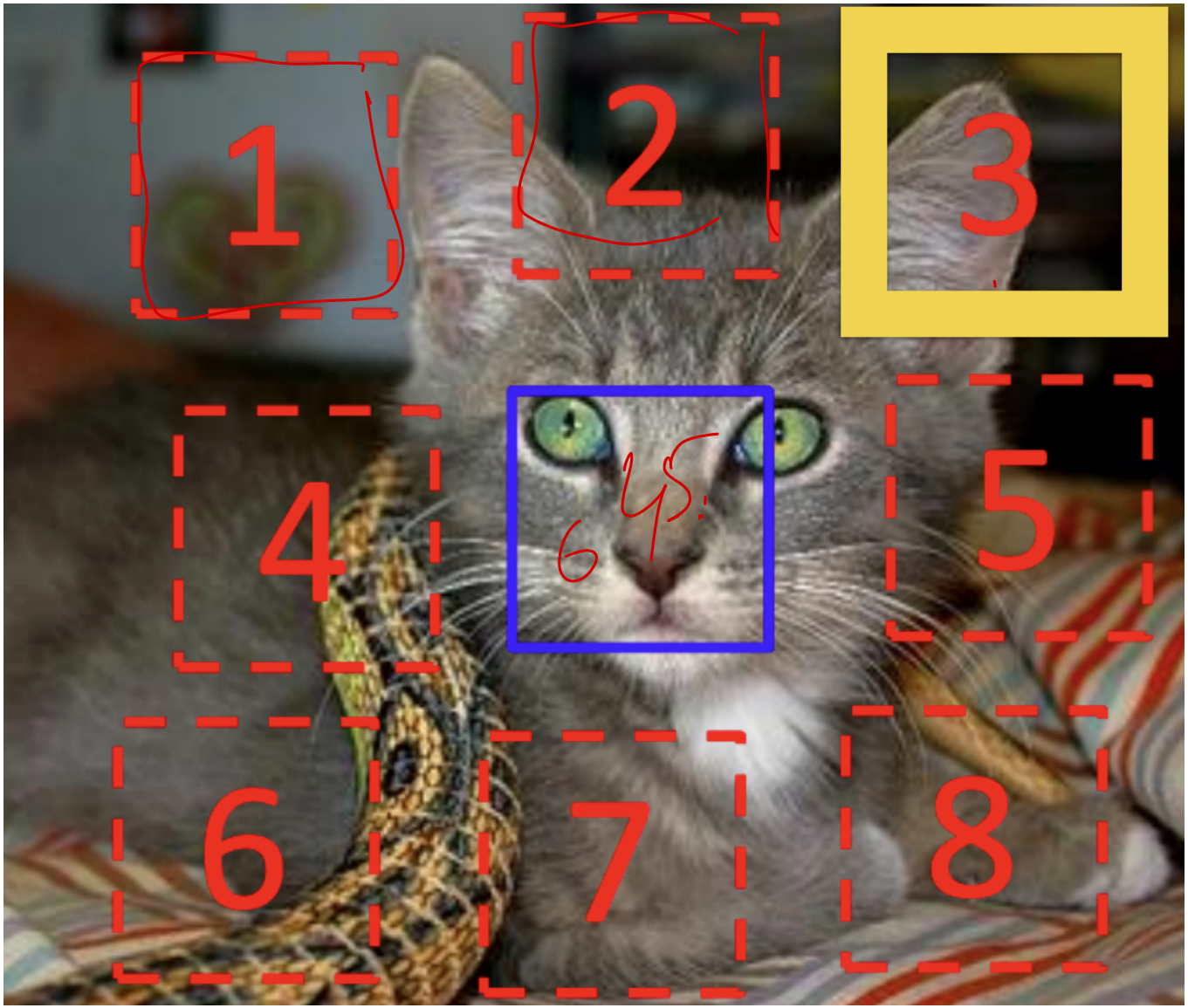
Task: Predict the relative position of the second patch with respect to the first

Relative Position of Image Patches (cont.)

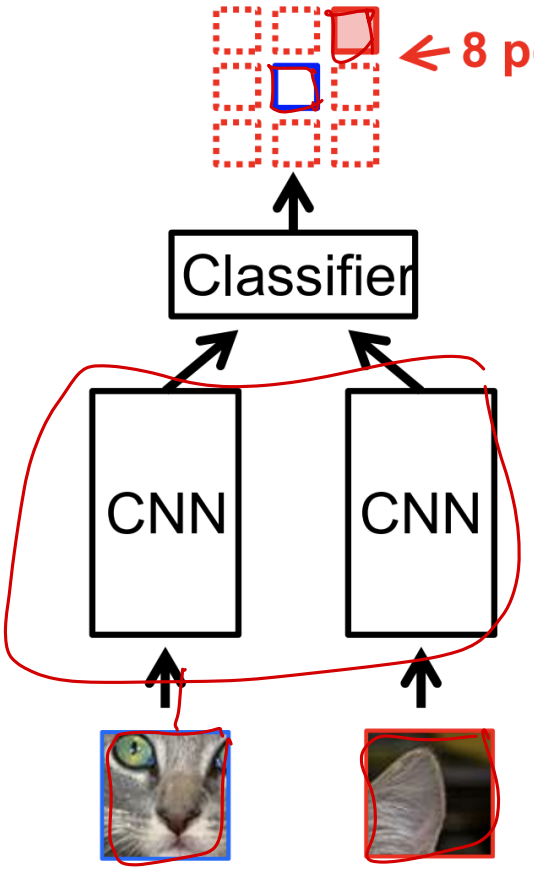
Shuffle image patches; predict their relative spatial positions.

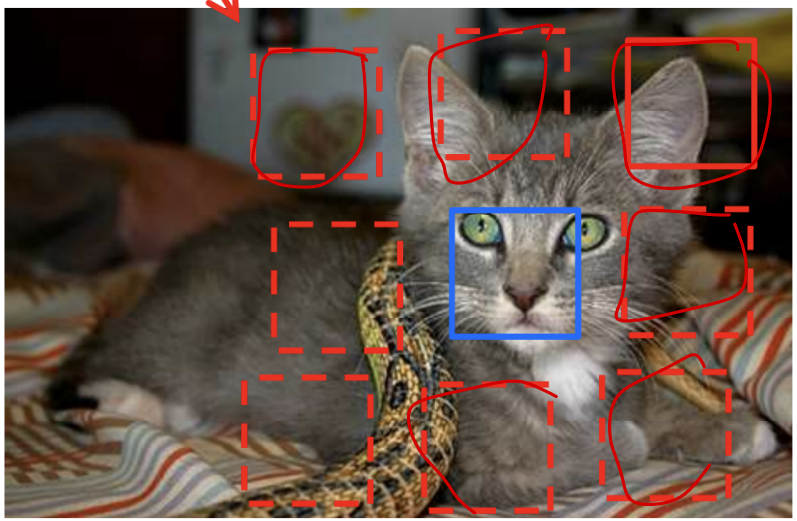
- The input image is divided into several non-overlapping patches.
- These patches are randomly shuffled to disrupt their original spatial arrangement.
- The model is tasked with predicting the original relative positions of the shuffled patches.

Relative Position of Image Patches (cont.)



Relative Position of Image Patches (cont.)





Randomly Sample Patch
Sample Second Patch

Unsupervised visual representation learning by context prediction,
Carl Doersch, Abhinav Gupta, Alexei A. Efros, ICCV 2015

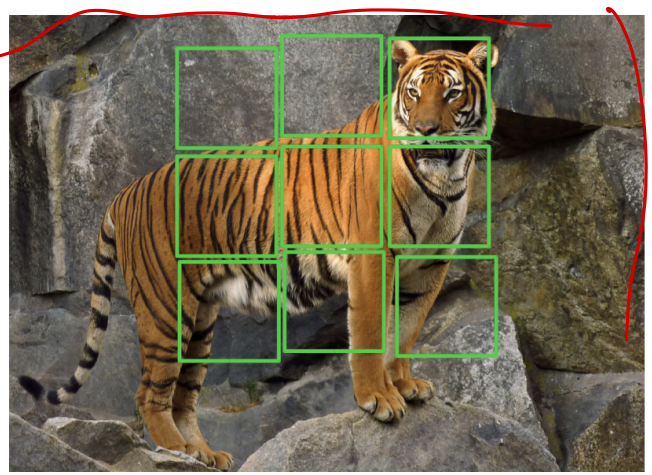
Relative Position of Image Patches (cont.)

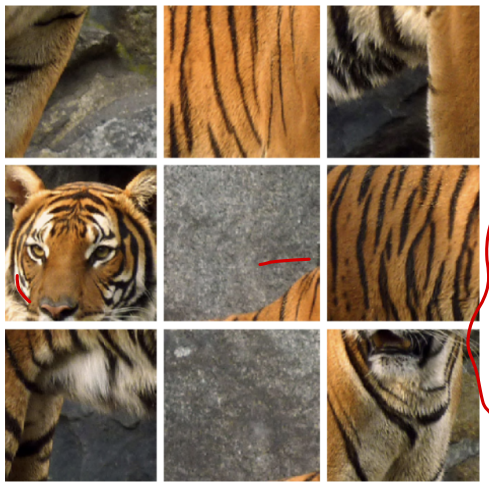
Model learns geometric relationships and spatial context.

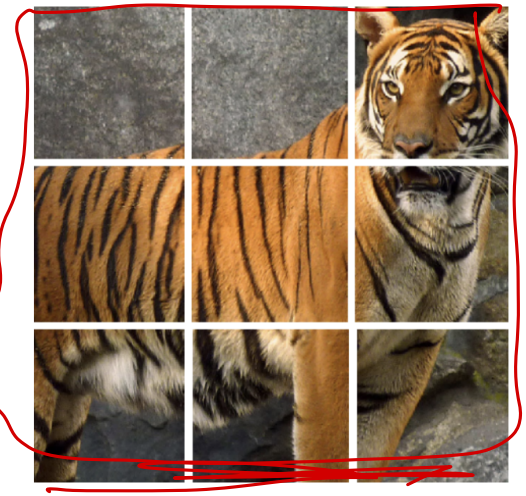
- By solving the patch position prediction task, the model is forced to understand the underlying spatial structure of objects and scenes.
- This encourages the learning of features that capture geometric relationships between different parts of the image.
- Such self-supervised tasks help the model develop a strong sense of spatial context, which is beneficial for downstream vision tasks like object detection and segmentation.

Relative Position of Image Patches (cont.)

Solving Jigsaw puzzles as a pretext task.





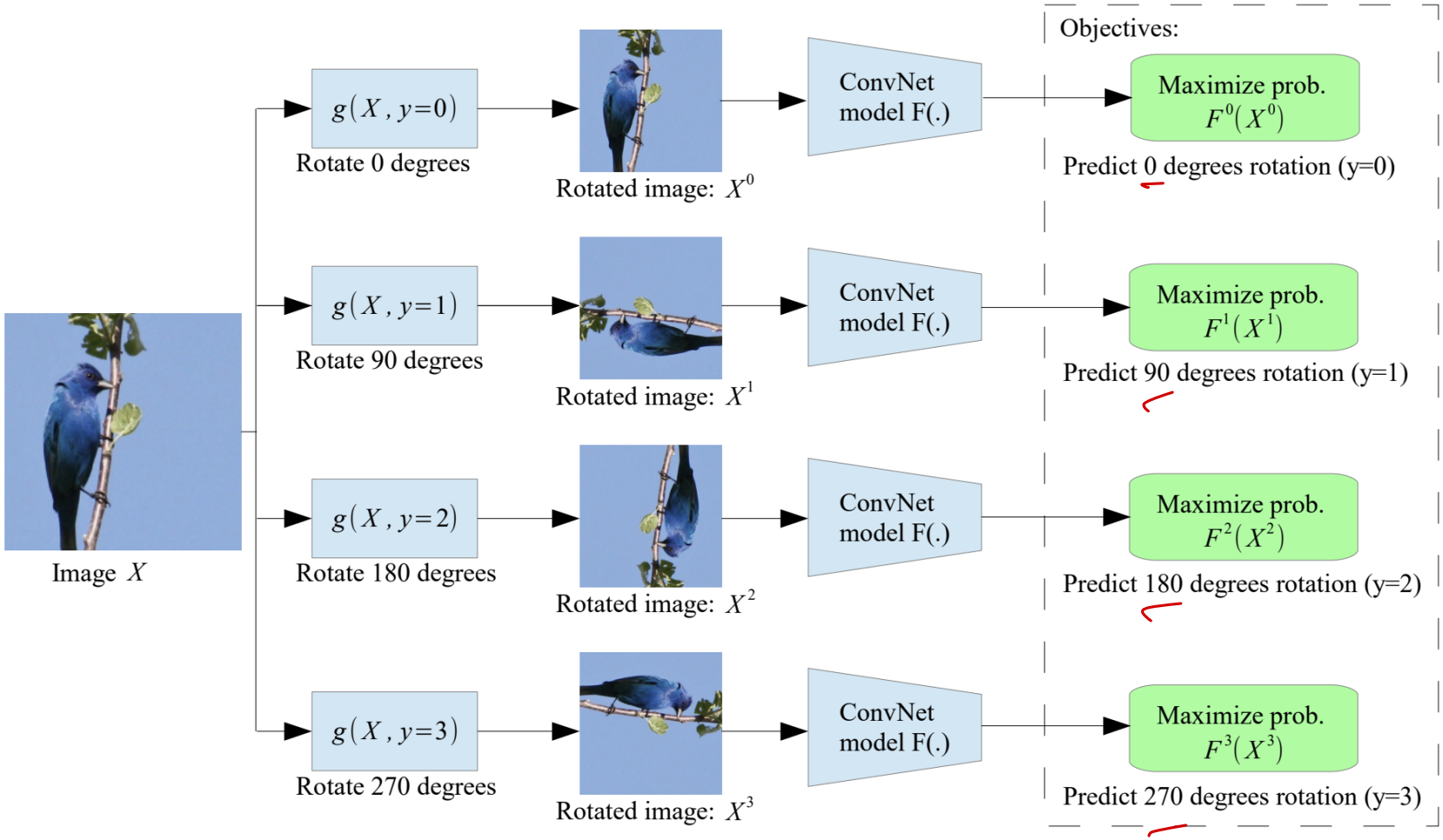


Rotation Prediction

Rotation Prediction is commonly used to learn useful image representations without requiring manual labels.

- **Task:** Rotate input images by a set of predefined angles (e.g., 0°, 90°, 180°, 270°).
- **Objective:** Train a neural network to predict the rotation angle applied to each image.
- **Implementation Steps:**
  - Randomly select a rotation angle from the set.
  - Apply the rotation to the input image.
  - Feed the rotated image into the network.
  - Use a classification head to predict the rotation angle.
  - Compute the loss (e.g., cross-entropy) and update the model.

Rotation Prediction (cont.)



Rotation Prediction (cont.)

# Rotations	Rotations	CIFAR-10 Classification Accuracy
4	0°, 90°, 180°, 270°	**89.06**
8	0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°	88.51
2	0°, 180°	87.46
2	90°, 270°	85.52

Rotation Prediction (cont.)

Benefits:

- Simple to implement and computationally inexpensive.
- Effective for learning global shape and semantic features.
- Can be used as a pretext task for downstream applications (e.g., image classification, object detection).

References:

- Gidaris, S., Singh, P., Komodakis, N. (2018). Unsupervised Representation Learning by Predicting Image Rotations. *ICLR 2018*.

أكاديمية كاوست
KAUST ACADEMY

Representation Learning with Contrastive Predictive Coding

Aaron van den Oord
DeepMind
avdnoord@google.com

Yazhe Li
DeepMind
yazhe@google.com

Oriol Vinyals
DeepMind
vinyals@google.com

Abstract

While supervised learning has enabled great progress in many applications, unsupervised learning has not seen such widespread adoption, and remains an important and challenging endeavor for artificial intelligence. In this work, we propose a universal unsupervised learning approach to extract useful representations from high-dimensional data, which we call Contrastive Predictive Coding. The key insight of our model is to learn such representations by predicting the future in *latent* space by using powerful autoregressive models. We use a probabilistic contrastive loss which induces the latent space to capture information that is maximally useful to predict future samples. It also makes the model tractable by using negative sampling. While most prior work has focused on evaluating representations for a particular modality, we demonstrate that our approach is able to learn useful representations achieving strong performance on four distinct domains: speech, images, text and reinforcement learning in 3D environments.

Contrastive Predictive Coding (CPC) (cont.)

Contrastive Predictive Coding (CPC) learns useful representations by predicting future information in latent space using powerful autoregressive models. The key components of CPC are:

- **Encoder**: Maps the input sequence $\mathbf{x}_t$ to a sequence of latent representations $\mathbf{z}_t$.
  - Typically implemented as a convolutional or recurrent neural network.
  - $\mathbf{z}_t = \text{Encoder}(\mathbf{x}_t)$
- **Autoregressor**: Aggregates the sequence of past latents $\{\mathbf{z}_1, \ldots, \mathbf{z}_t\}$ into a context vector $\mathbf{c}_t$.
  - Often implemented as an RNN or masked transformer.
  - $\mathbf{c}_t = \text{AR}(\mathbf{z}_{\leq t})$
- **Prediction**: The model predicts future latent representations using the context vector.
  - For each step k, a linear transformation W_k is applied to the context: $\hat{\mathbf{z}}_{t+k} = W_k \mathbf{c}_t$

Contrastive Predictive Coding (CPC) (cont.)

- The goal is to distinguish the true future latent $\mathbf{z}_{t+k}$ from negative samples.

- **InfoNCE Loss**: A contrastive loss that encourages the predicted future latent to be similar to the true future latent and dissimilar to negative samples.

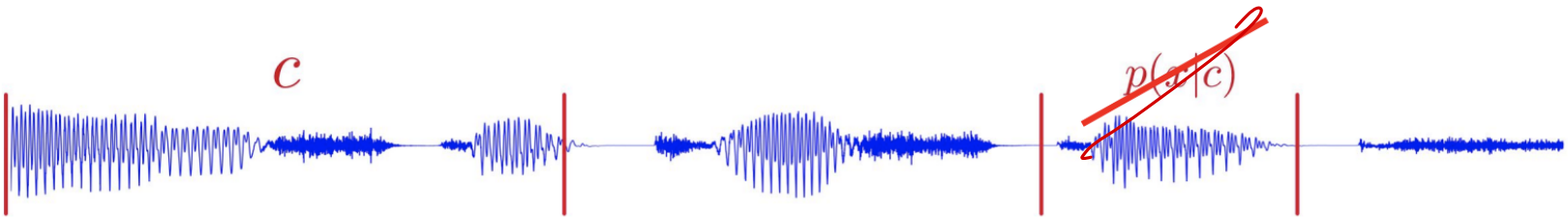
  - For each prediction step k:

$$\mathcal{L}_{\text{InfoNCE}} = -\mathbb{E}\left[\log \frac{\exp\left(\mathbf{z}_{t+k}^{\top} W_k \mathbf{c}_t\right)}{\sum_j \exp\left(\mathbf{z}_j^{\top} W_k \mathbf{c}_t\right)}\right]$$

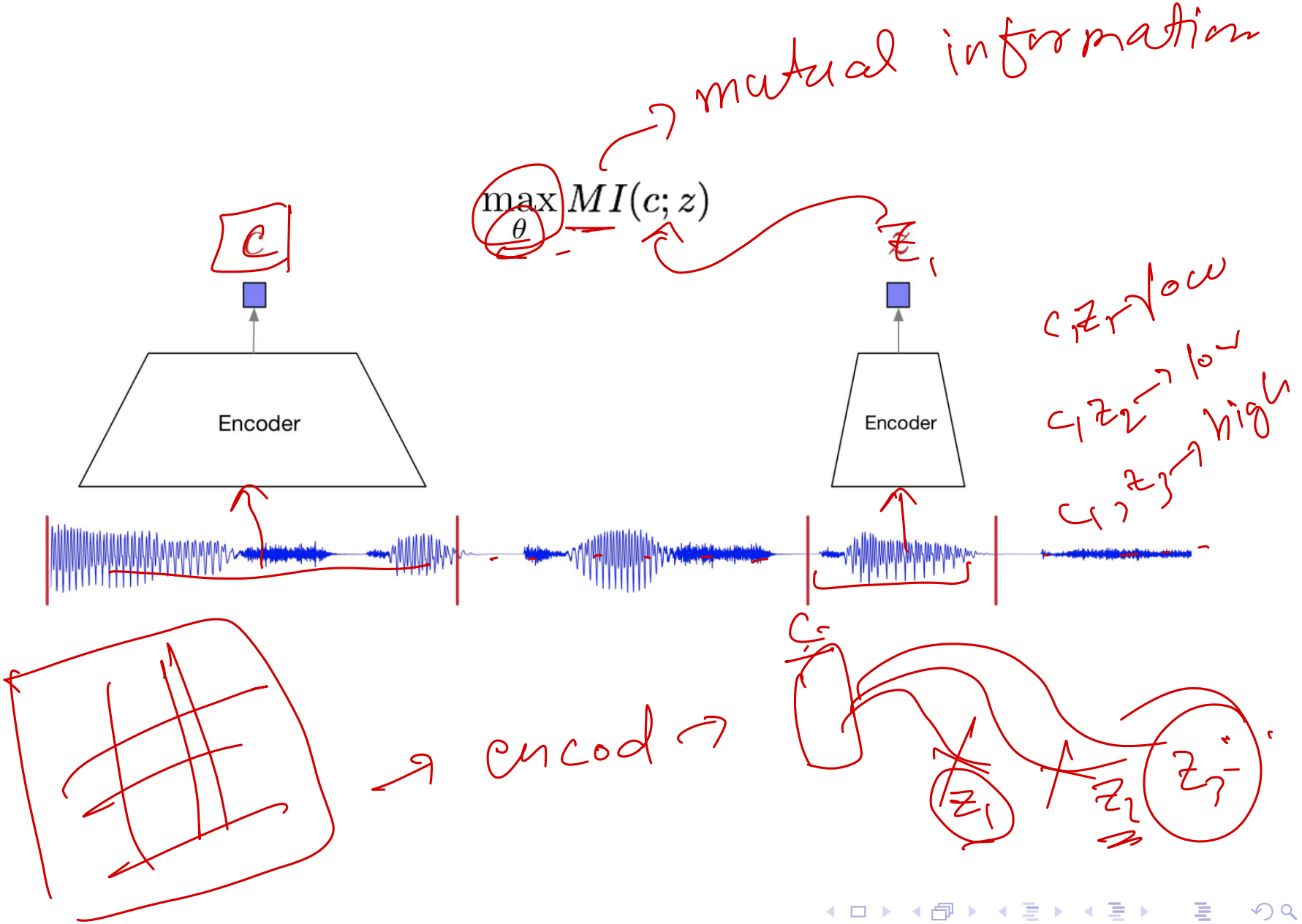
  - The denominator sums over one positive and multiple negative samples.

Contrastive Predictive Coding (CPC) (cont.)

Contrastive Predictive Coding (CPC) (cont.)



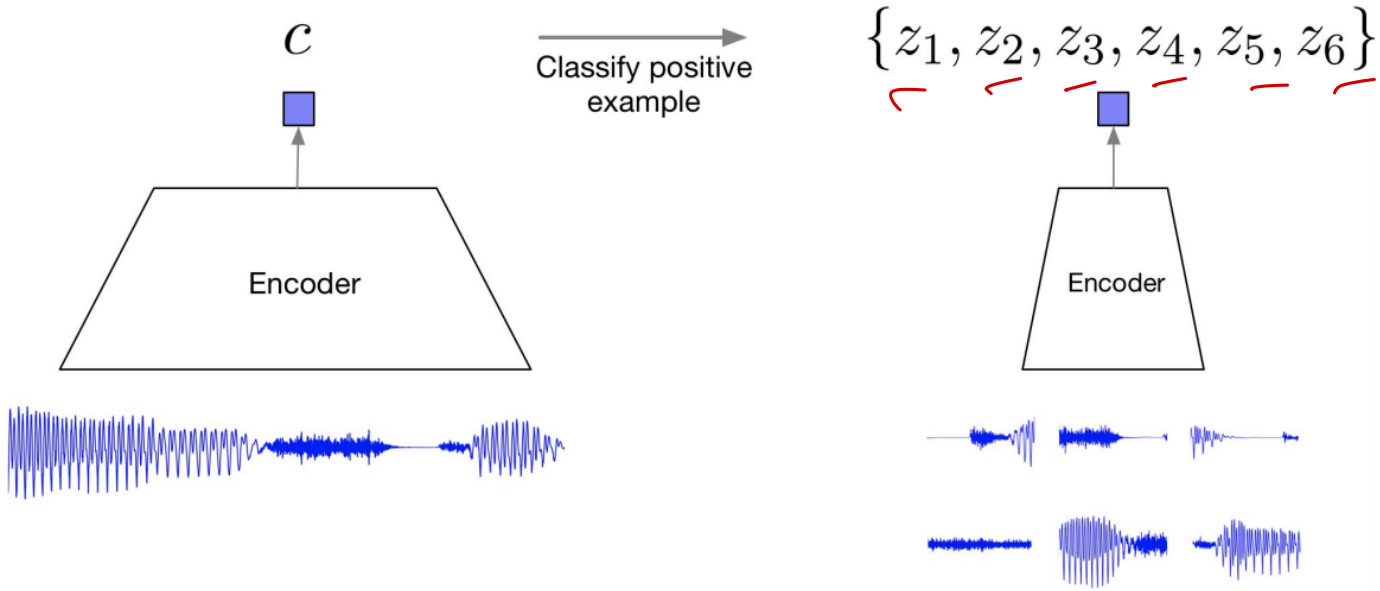
Contrastive Predictive Coding (CPC) (cont.)



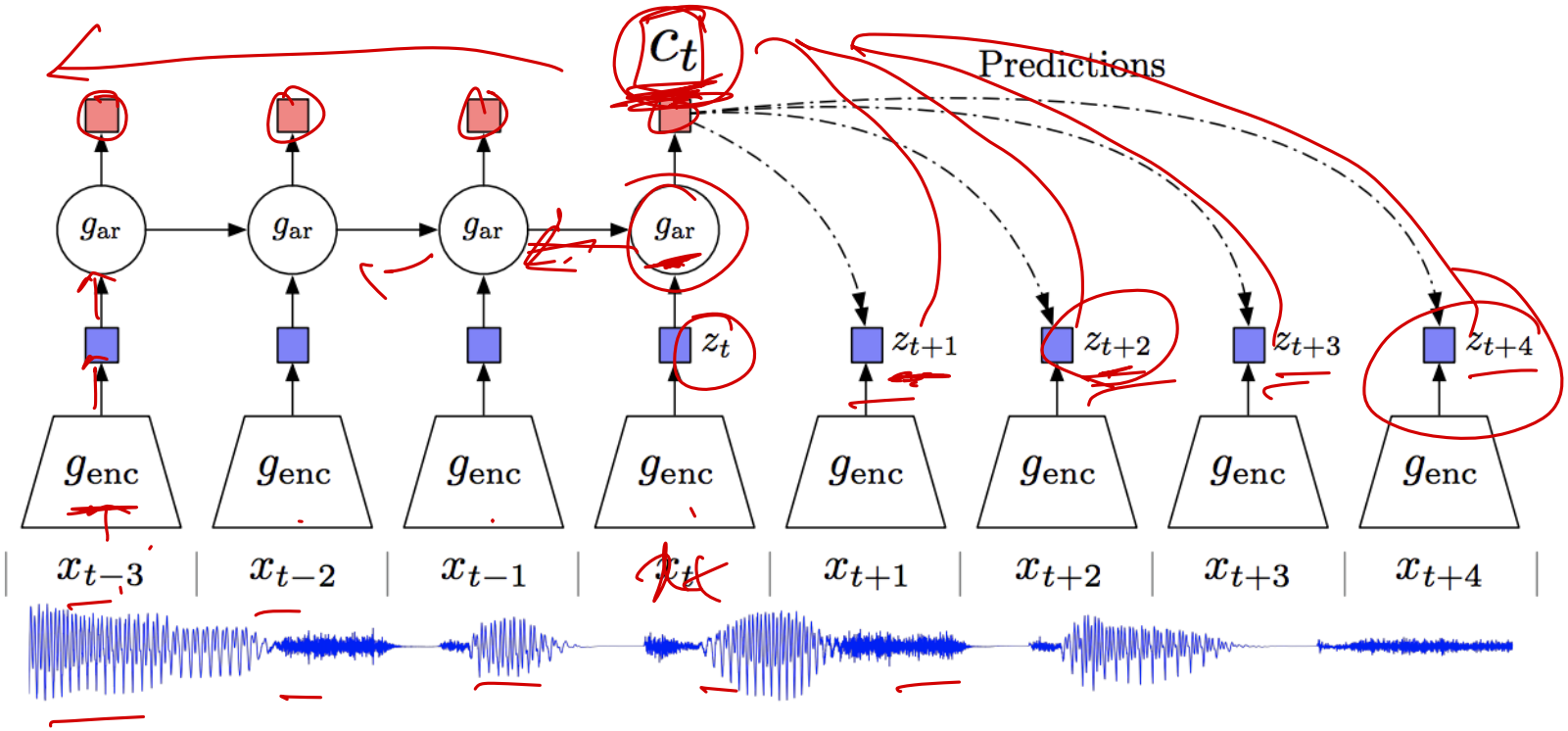
Contrastive Predictive Coding (CPC) (cont.)

$$\frac{\exp f(c, z_i)}{\sum_j \exp f(c, z_j)}$$

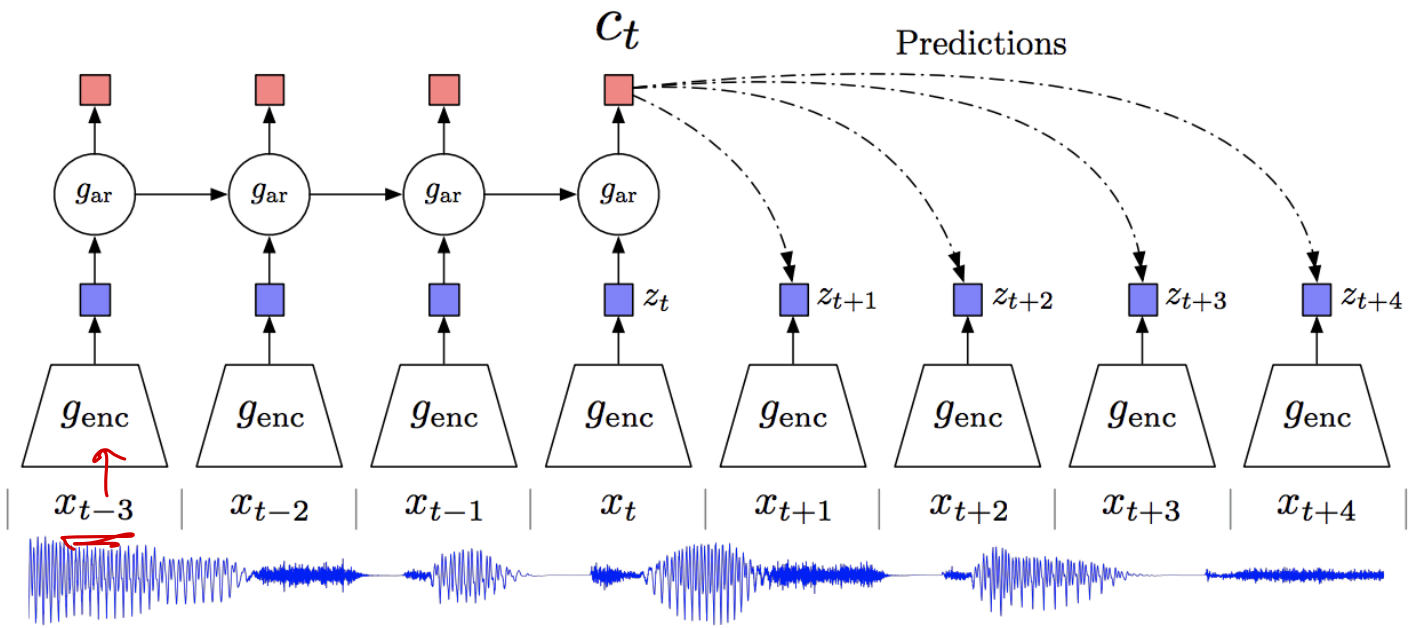
$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$



Contrastive Predictive Coding (CPC) (cont.)



Contrastive Predictive Coding (CPC) (cont.)



$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

$$\mathcal{L}_N = -\mathop{\mathbb{E}}_X \left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

Contrastive Predictive Coding (CPC) (cont.)

Key Points about CPC:

- CPC learns representations by maximizing mutual information between context and future latent representations.
- It is widely used in audio, vision, and language domains for self-supervised pretraining.
- The contrastive loss enables learning without explicit labels, relying on the structure of the data itself.

CPC: Speech

Method	ACC
Phone classification	
Random initialization	27.6
MFCC features	39.7
CPC	64.6
Supervised	74.6
Speaker classification	
Random initialization	1.87
MFCC features	17.6
CPC	97.4
Supervised	98.5

Table 1: LibriSpeech phone and speaker classification results. For phone classification there are 41 possible classes and for speaker classification 251. All models used the same architecture and the same audio input sizes.

Method	ACC
#steps predicted	
2 steps	28.5
4 steps	57.6
8 steps	63.6
12 steps	64.6
16 steps	63.8
Negative samples from	
Mixed speaker	64.6
Same speaker	65.5
Mixed speaker (excl.)	57.3
Same speaker (excl.)	64.6
Current sequence only	65.2

Table 2: LibriSpeech phone classification ablation experiments. More details can be found in Section 3.1.

CPC: ImageNet

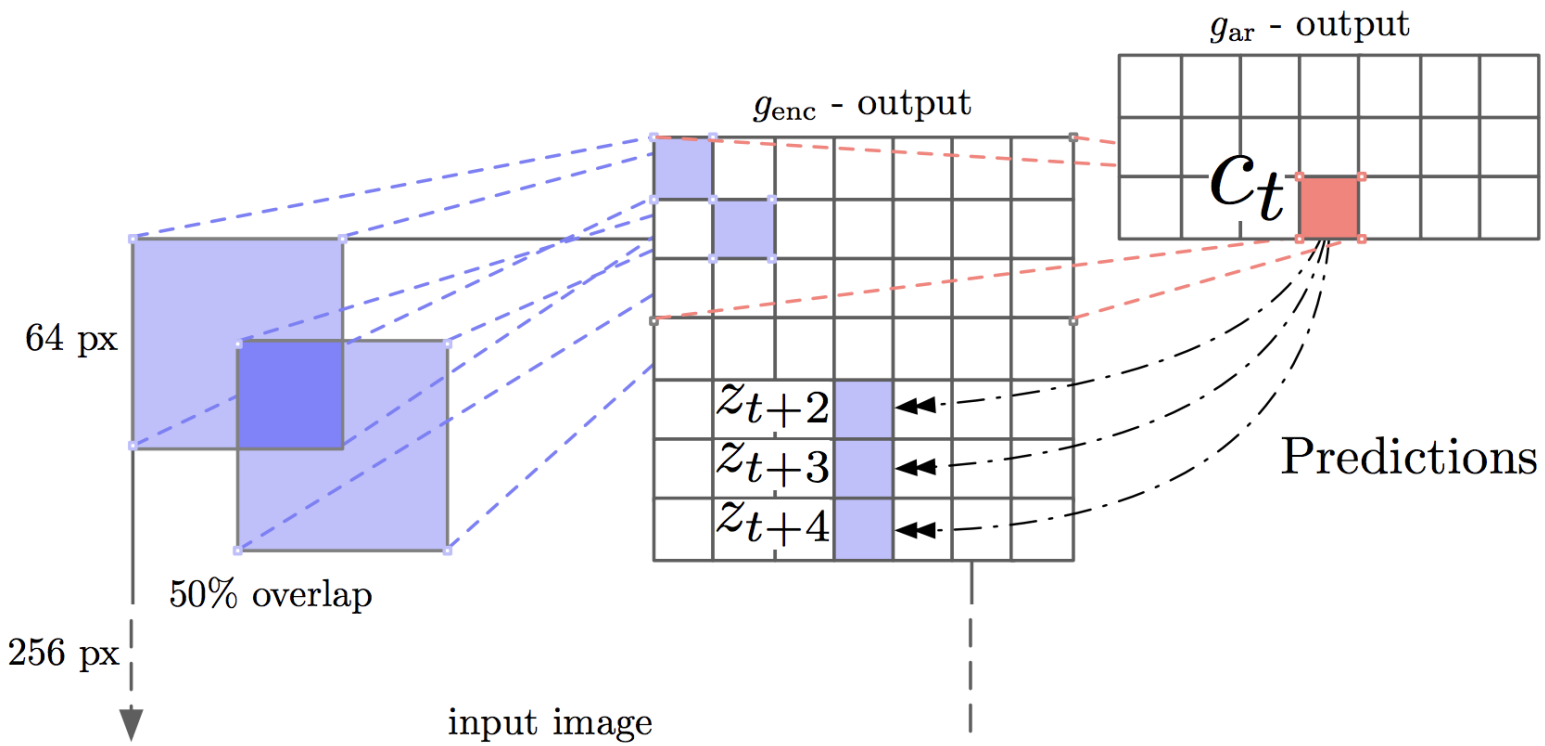


Figure 4: Visualization of Contrastive Predictive Coding for images (2D adaptation of Figure 1).

CPC: ImageNet (cont.)

CPC: ImageNet (cont.)

Method	Top-1 ACC
Using AlexNet conv5	
Video [28]	29.8
Relative Position [11]	30.4
BiGan [35]	34.8
Colorization [10]	35.2
Jigsaw [29] *	38.1
Using ResNet-V2	
Motion Segmentation [36]	27.6
Exemplar [36]	31.5
Relative Position [36]	36.2
Colorization [36]	39.6
CPC	**48.7**

Table 3: ImageNet top-1 unsupervised classification results. *Jigsaw is not directly comparable to the other AlexNet results because of architectural differences.

Method	Top-5 ACC
Motion Segmentation (MS)	48.3
Exemplar (Ex)	53.1
Relative Position (RP)	59.2
Colorization (Col)	62.5
Combination of MS + Ex + RP + Col	69.3
CPC	**73.6**

Table 4: ImageNet top-5 unsupervised classification results. Previous results with MS, Ex, RP and Col were taken from [36] and are the best reported results on this task.

CPC: ImageNet (cont.)

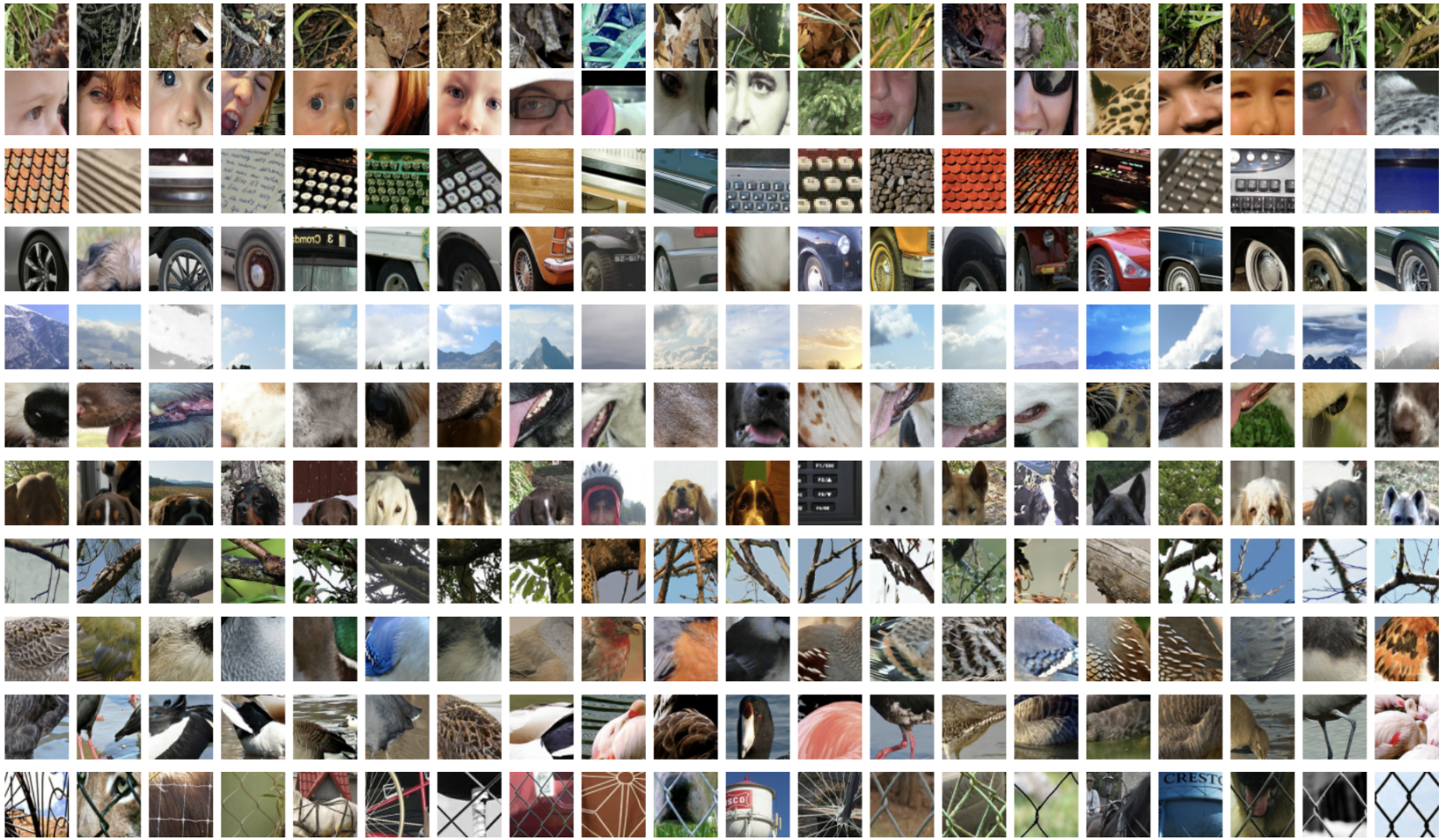


Figure 5: Every row shows image patches that activate a certain neuron in the CPC architecture.

CPC: Natural Language Processing

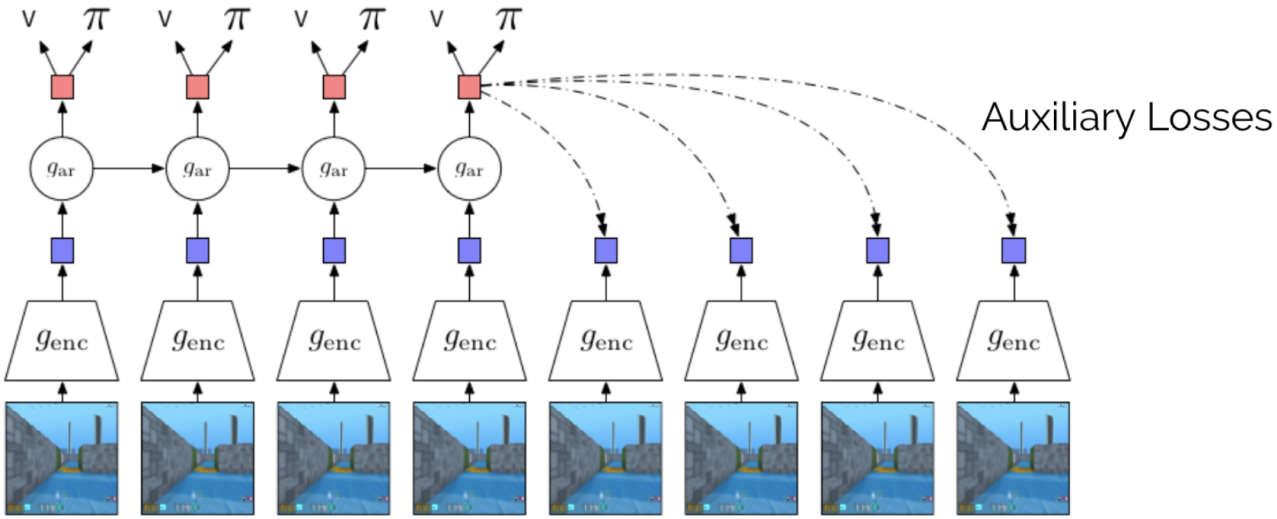
أكاديمية كاوست
KAUST ACADEMY

Method	MR	CR	Subj	MPQA	TREC
Paragraph-vector [40]	74.8	78.1	90.5	74.2	91.8
Skip-thought vector [26]	75.5	79.3	92.1	86.9	91.4
Skip-thought + LN [41]	79.5	82.6	93.4	89.0	-
CPC	76.9	80.1	91.2	87.7	96.8

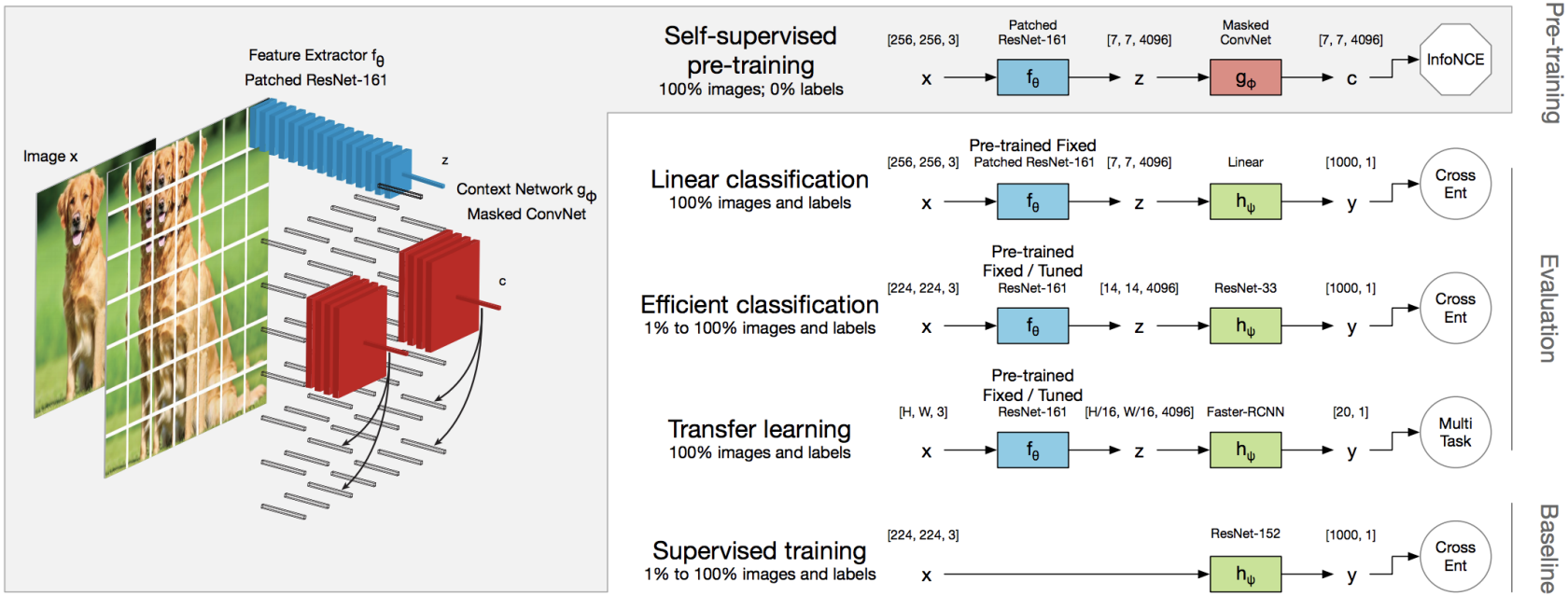
Table 5: Classification accuracy on five common NLP benchmarks. We follow the same transfer learning setup from Skip-thought vectors [26] and use the BookCorpus dataset as source. [40] is an unsupervised approach to learning sentence-level representations. [26] is an alternative unsupervised learning approach. [41] is the same skip-thought model with layer normalization trained for 1M iterations.

CPC: Reinforcement Learning

Auxiliary loss is on policy
Predict 30 steps in the future



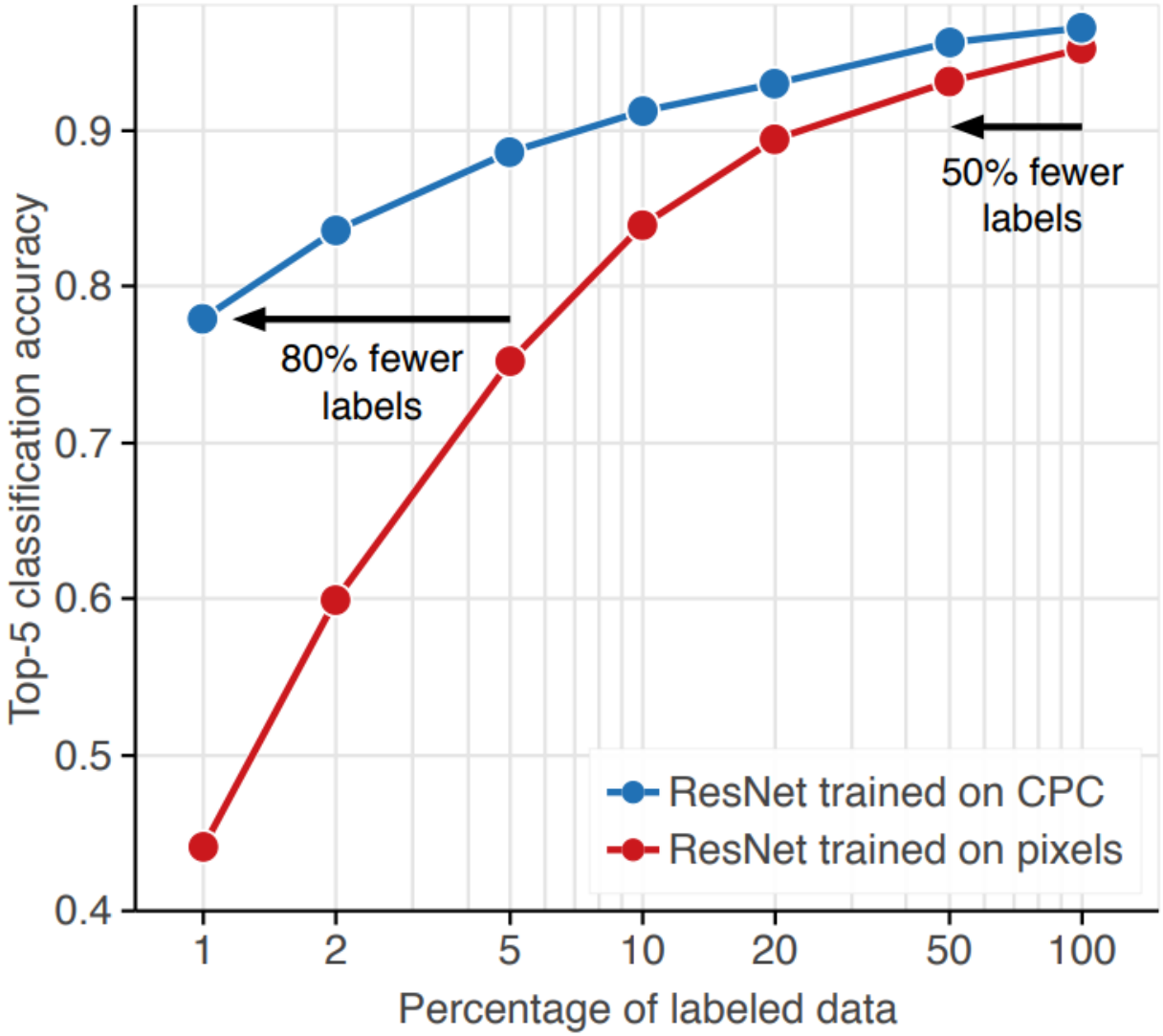
CPCv2: Large Scale CPC on ImageNet



CPCv2: Linear Classification

Method	Architecture	Params. (M)	Top-1 Acc.
Motion Segmentation	ResNet-101	28	27.6
Exemplar	ResNet-101	28	31.5
Relative Position	ResNet-101	28	36.2
Colorization	ResNet-101	28	39.6
CPC v1	ResNet-101	28	48.7
Rotation	RevNet-50 ×4	86	55.4
BigBiGAN	RevNet-50 ×4	86	61.3
AMDIM	Custom-103	626	68.1
CMC	ResNet-50 ×2	188	68.4
Momentum Contrast	ResNet-50 ×4	375	68.6
CPC v2	ResNet-161	305	**71.5**
Local Aggregation	ResNet-50	24	60.2
Momentum Contrast	ResNet-50	24	60.6
CPC v2	ResNet-50	24	**63.8**

CPCv2: Data-Efficient Image Recognition



CPCv2: Data-Efficient Supervised Learning

Method	Architecture	Top-5 accuracy				
Labeled data		1%	5%	10%	50%	100%
†Supervised baseline	ResNet-200	44.1	75.2⋆	83.9	93.1	95.2#
Methods using label-propagation:						
Pseudolabeling [63]	ResNet-50	51.6	-	82.4	-	-
VAT + Entropy Minimization [63]	ResNet-50	47.0	-	83.4	-	-
Unsup. Data Augmentation [61]	ResNet-50	-	-	88.5	-	-
Rotation + VAT + Ent. Min. [63]	ResNet-50 ×4	-	-	**91.2**	-	95.0
Methods using representation learning only:						
Instance Discrimination [60]	ResNet-50	39.2	-	77.4	-	-
Rotation [63]	ResNet-152 ×2	57.5	-	86.4	-	-
ResNet on BigBiGAN (fixed)	RevNet-50 ×4	55.2	73.7	78.8	85.5	87.0
ResNet on AMDIM (fixed)	Custom-103	67.4	81.8	85.8	91.0	92.2
ResNet on CPC v2 (fixed)	ResNet-161	77.1	87.5	90.5	95.0	96.2
ResNet on CPC v2 (fine-tuned)	ResNet-161	**77.9**⋆	**88.6**	**91.2**	**95.6**#	**96.5**

Pretext Task Evolution:

- Earlier: handcrafted tasks (e.g., jigsaw, colorization).
- Now: contrastive learning directly optimizes for invariance and discrimination.
- Leads to better, more transferable features and narrows the gap with supervised learning.

Instance Discrimination & Contrastive Learning (cont.)

Instance Discrimination:

- Treats each image as its own class.
- Positive pairs: augmented views of the same image.
- Negative pairs: views from different images.
- Promotes invariance to augmentations and discrimination between instances.

Contrastive Learning:

- Self-supervised approach for learning representations, especially in vision.
- Uses data structure instead of labels.
- Learns by pulling together similar (positive) pairs and pushing apart dissimilar (negative) pairs using a contrastive loss (e.g., InfoNCE).

Fundamentals of Instance Discrimination

Instance Discrimination is a self-supervised learning approach where each individual sample in the dataset is treated as a distinct class.

- The core idea is to learn representations by distinguishing between different instances, rather than relying on human-annotated labels.
- **Positive pairs** are generated by applying different augmentations (such as cropping, color jittering, or flipping) to the same image. These augmented views are considered to belong to the same class (i.e., the same instance).
- **Negative pairs** are formed by pairing augmented views of different images. These are treated as belonging to different classes (i.e., different instances).

Fundamentals of Instance Discrimination (cont.)

1. MoCo
2. SimCLR

Fundamentals of Instance Discrimination (cont.)

- The learning objective is to **maximize agreement** (similarity) between representations of positive pairs, while minimizing agreement between negative pairs.
- This is typically achieved using a contrastive loss function, such as InfoNCE, which encourages the model to bring positive pairs closer in the embedding space and push negative pairs apart.
- Instance discrimination forms the basis for many popular self-supervised learning methods, such as MoCo, SimCLR, and PIRL.
- By maximizing agreement across views of the same instance, the model learns to extract meaningful and invariant features, which can be transferred to downstream tasks.

أكاديمية كاوست
KAUST ACADEMY

Momentum Contrast for Unsupervised Visual Representation Learning

Kaiming He Haoqi Fan Yuxin Wu Saining Xie Ross Girshick

Facebook AI Research (FAIR)

Momentum Contrast (MoCo) (cont.)

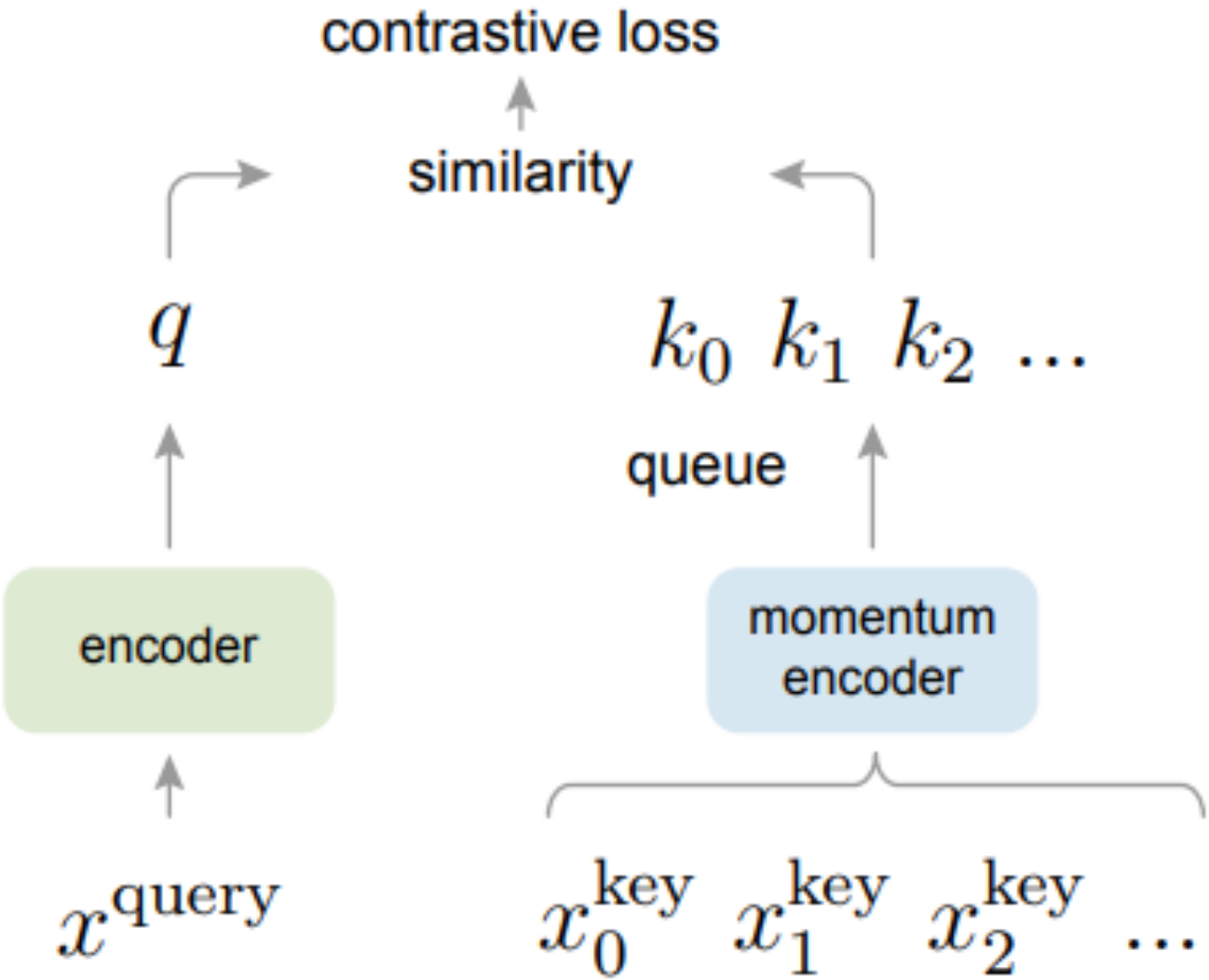
Momentum Contrast (MoCo) is a self-supervised learning framework designed for visual representation learning. Its key components are:

Query & Key Encoders: Two neural networks, f_q (query encoder) and f_k (key encoder), are used. The key encoder f_k is updated as an exponential moving average of the query encoder f_q:

$$\theta_k \leftarrow m\theta_k + (1 - m)\theta_q$$

where θ_k and θ_q are the parameters of f_k and f_q, and m is the momentum coefficient (e.g., $m = 0.999$).

Momentum Contrast (MoCo) (cont.)



Momentum Contrast (MoCo) (cont.)

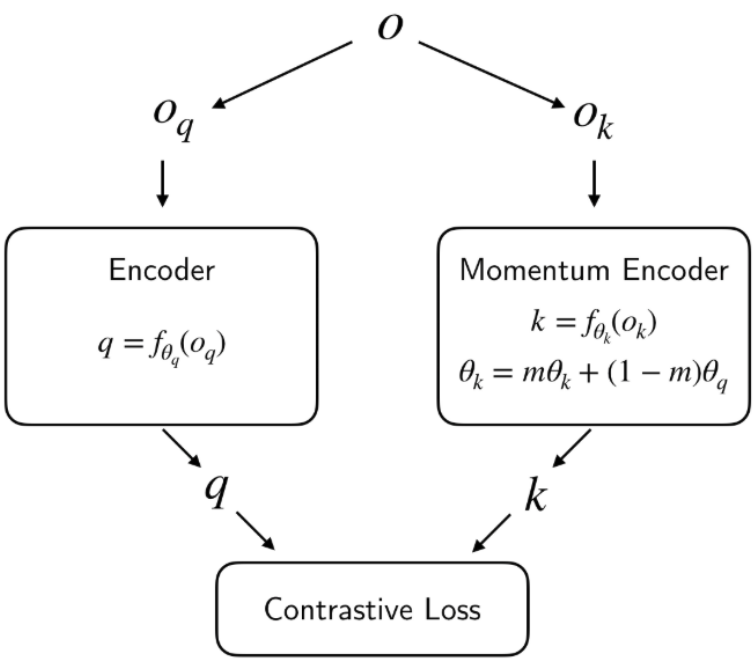
Dictionary Queue: MoCo maintains a large queue (dictionary) of encoded keys from previous batches. This enables the use of a large and consistent set of negative samples for contrastive learning, which is crucial for effective representation learning.

Contrastive Loss: The InfoNCE loss is used to train the encoders. For a given query q and its positive key k^+, along with a set of negative keys $\{k_0, k_1, ..., k_N\}$ from the dictionary, the loss is:

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

where τ is a temperature hyperparameter.

Momentum Contrast (MoCo) (cont.)



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

Momentum Contrast (MoCo) (cont.)

Key Steps in MoCo Training:

1. For each image, generate two augmentations: one for the query encoder (f_q), one for the key encoder (f_k).
2. Encode the query and key.
3. Compute the InfoNCE loss using the current positive key and the dictionary of negative keys.
4. Update f_q via backpropagation; update f_k via momentum.
5. Enqueue the new key and dequeue the oldest key to maintain the dictionary size.

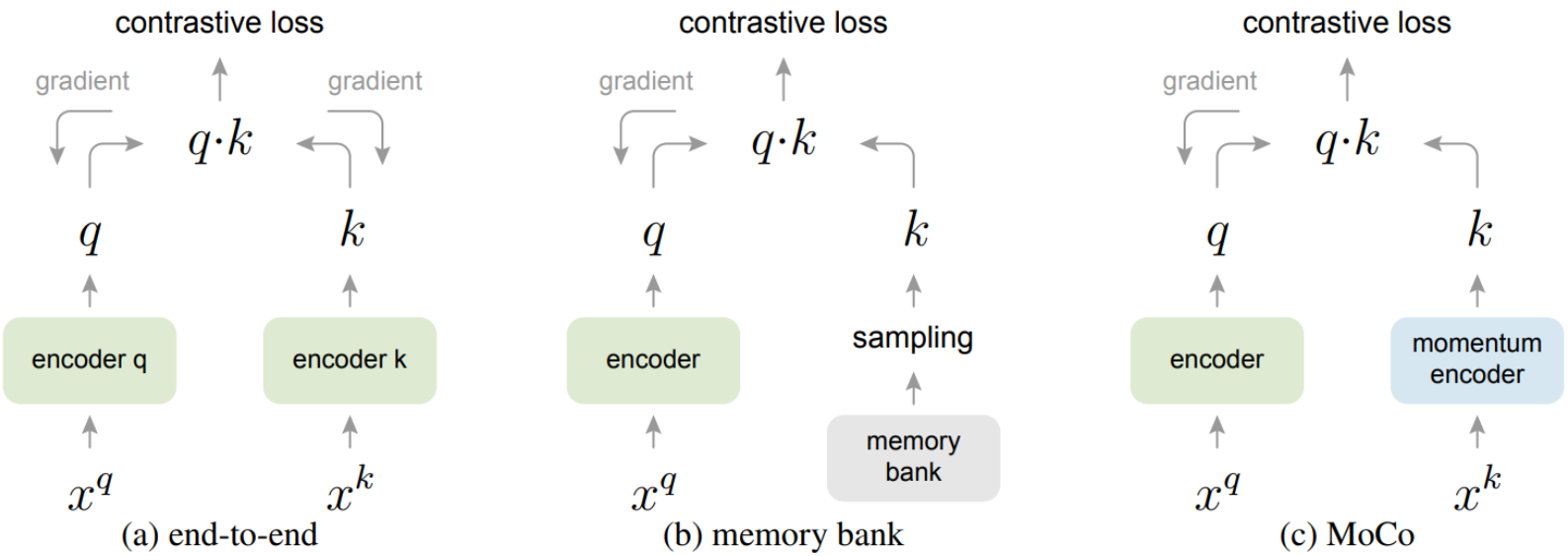
Momentum Contrast (MoCo) (cont.)

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

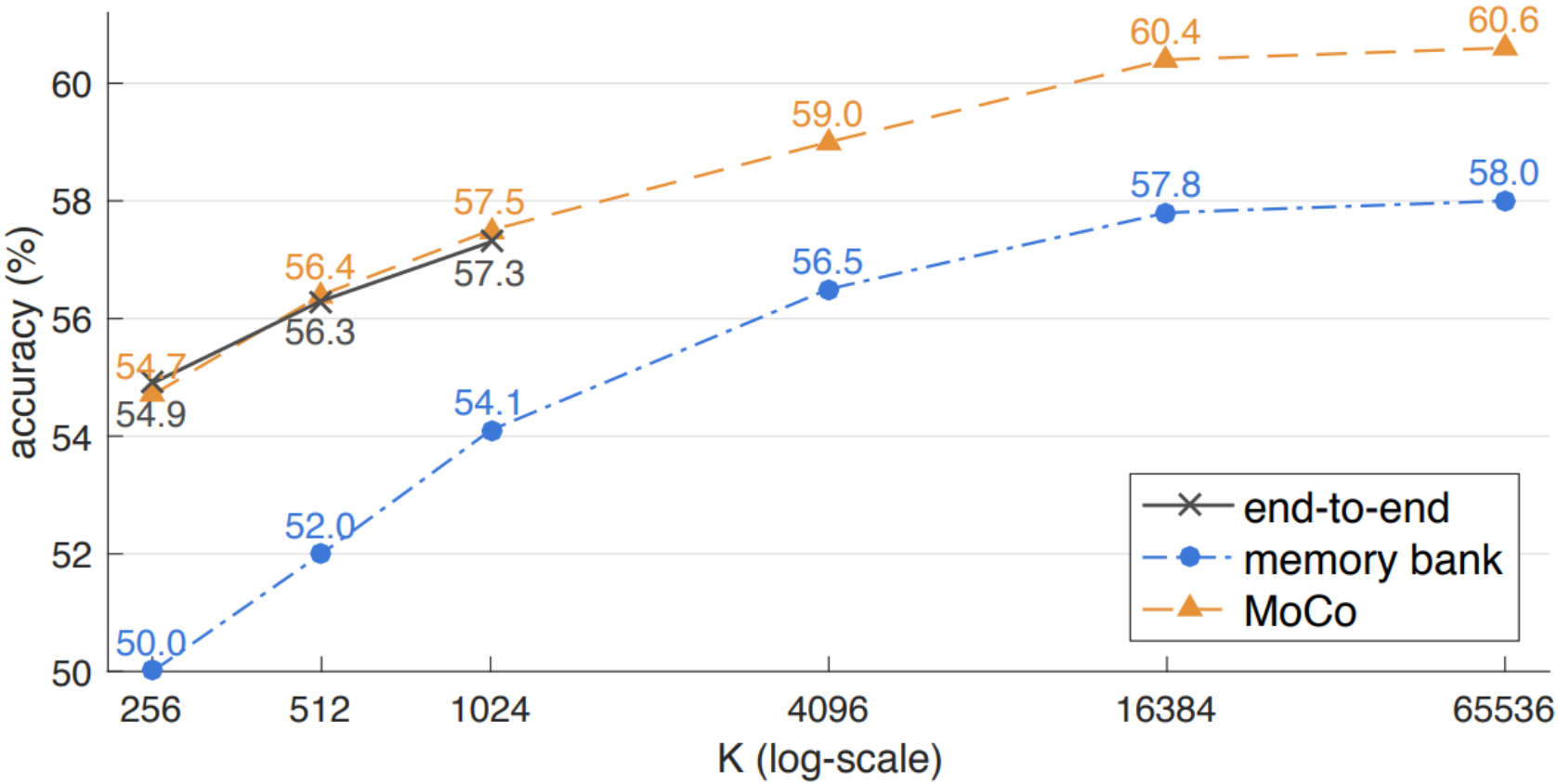
```
# f_q, f_k: encoder networks for query and key
# queue: dictionary as a queue of K keys (CxK)
# m: momentum
# t: temperature

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.

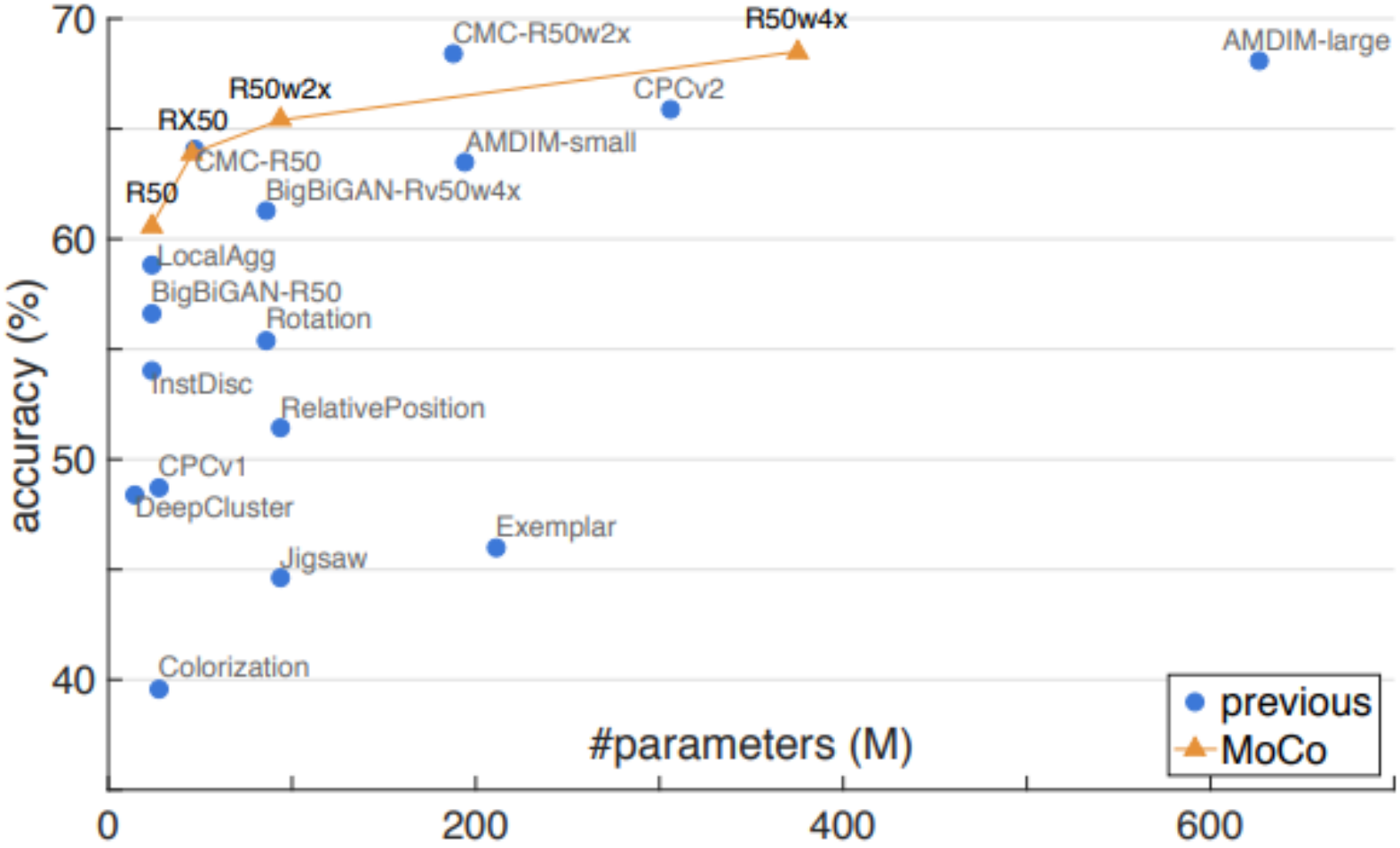
Momentum Contrast (MoCo) (cont.)



Momentum Contrast (MoCo) (cont.)



Momentum Contrast (MoCo) (cont.)



Momentum Contrast (MoCo) (cont.)

Key Advantages:

- Enables large and consistent negative sets for contrastive learning.
- Momentum update stabilizes the key encoder, improving training.
- Scalable to large datasets and high-dimensional representations.

References

[1] Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P.-A. (2008). *Extracting and Composing Robust Features with Denoising Autoencoders*. ICML.

[2] Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T., & Efros, A. A. (2016). *Context Encoders: Feature Learning by Inpainting*. CVPR.

[3] Noroozi, M., & Favaro, P. (2016). *Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles*. ECCV.

[4] Gidaris, S., Singh, P., & Komodakis, N. (2018). *Unsupervised Representation Learning by Predicting Image Rotations*. ICLR.

[5] van den Oord, A., Li, Y., & Vinyals, O. (2018). *Representation Learning with Contrastive Predictive Coding*. arXiv:1807.03748.

References (cont.)

[6] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space*. arXiv:1301.3781.

[7] He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). *Momentum Contrast for Unsupervised Visual Representation Learning*. CVPR.

Credits

Dr. Prashant Aparajeya

Computer Vision Scientist — Director(AISimply Ltd)

p.aparajeya@aisimply.uk

This project benefited from external collaboration, and we acknowledge their contribution with gratitude.