photo by unsplash user @mireilleraad

COMP547

DEEP UNSUPERVISED LEARNING

Lecture #12 – Self-Supervised Learning

KOÇ UNIVERSITY

Aykut Erdem // Koç University // Spring 2022

Previously on COMP547

- Autoregressive models
- Flow models
- Latent Variable models
- Implicit models
- Diffusion Models


Synthetic image sampled from BigGAN, Flow++ and VQ-VAE-2 trained on ImageNet

Lecture overview

- Motivation
- Reconstruct from a corrupted (or partial) version
- Proxy tasks in computer vision
- Contrastive Learning

Disclaimer: Much of the material and slides for this lecture were borrowed from

—Pieter Abbeel, Peter Chen, Jonathan Ho, Aravind Srinivas' Berkeley CS294-158 class

—Aaron Courville's Université de Montréal IFT6268 class

Lecture overview

- **Motivation**
- Reconstruct from a corrupted (or partial) version
- Proxy tasks in computer vision
- Contrastive Learning

Course so far…

- Density modelling
  - Autoregressive, Normalizing Flows, Variational Inference
- Implicit models
  - Generative Adversarial Networks
- Applications of generative modelling

Today...

- How do learn rich and useful features from raw unlabeled data that can be useful for several downstream tasks?
- What are the various pretext (proxy) tasks that can be used to learn representations from unlabeled data?
- How can we improve data-efficiency and performance of downstream tasks with a good pre-trained network?

Learning "really useful" representations

- Longstanding dream of the Deep Learning community:
  - Use unsupervised learning to learn some feature representation that can be used to support effective supervised learning (like classification)

Input

Extract Information

Representation

Tasks

Human-level interaction

Dialogue

Generation

Translation

Anomaly detection

Retrieval

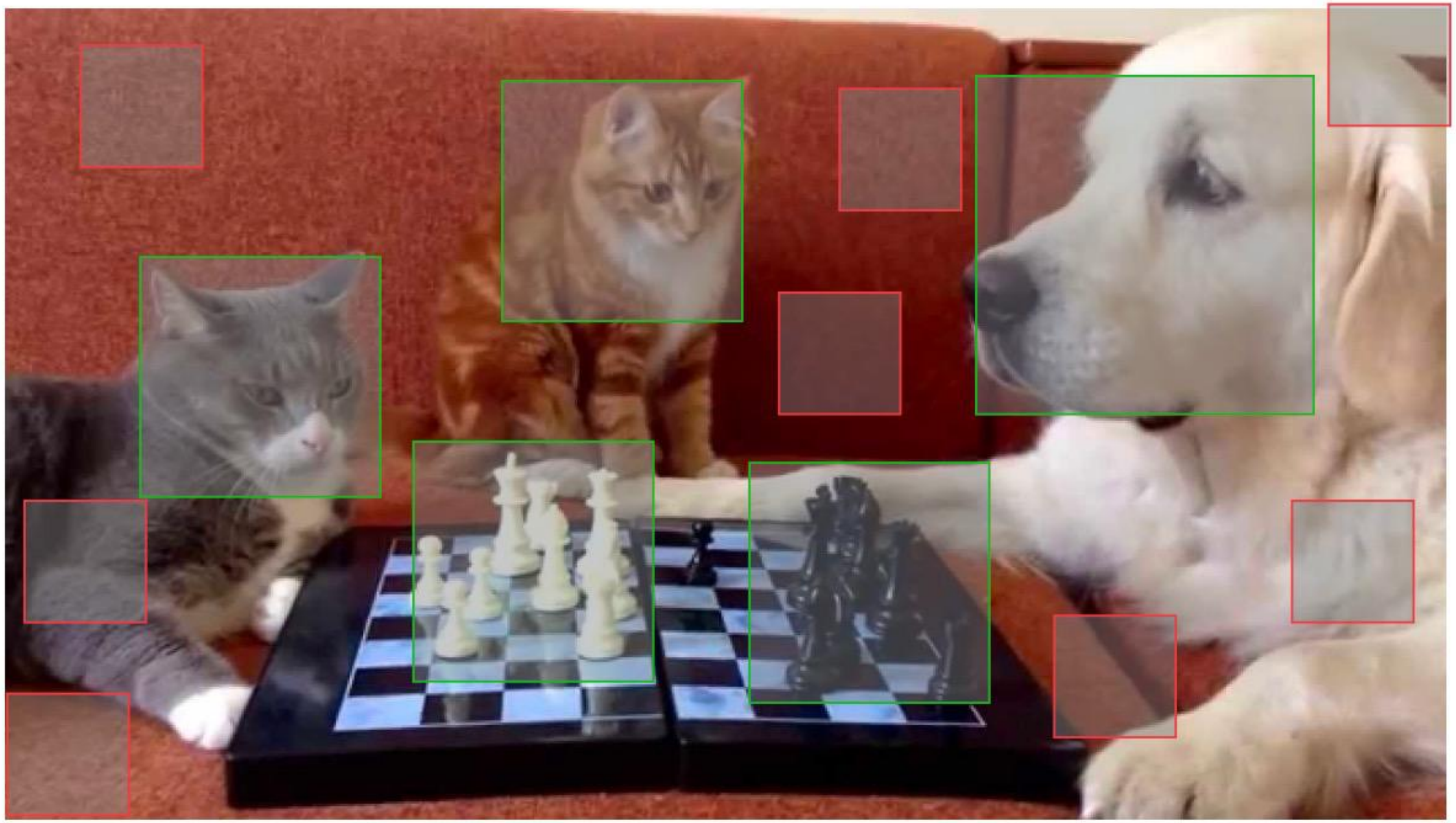
Semantic Segmentation

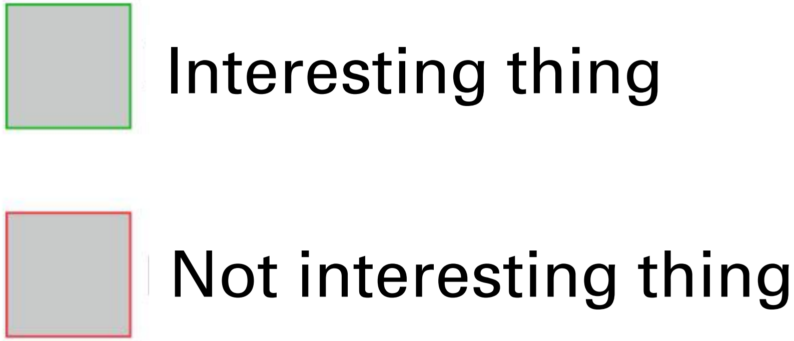
Classification

More utiliy = better

(Task) Generalization ⇔ Understanding

We don't need generation/reconstruction





- Generative models (in principle) care about all the pixels

Representation Learning for Supervised Learning

- Using generative models (AEs, VAEs, etc) have largely been ineffective with two exceptions:
  1. Natural Language Modelling (all SOTA models are build on BERT-like representations)
  2. In the very small dataset regime, unsupervised learning can actually help.
- Gradient-based supervised training with the right model (e.g. CNNs for vision problems) has been very difficult to beat with unsupervised methods.

It's worth asking … Why?

A speculative answer:

- Most (essentially all) existing unsupervised methods learn features that are overwhelmingly low-level (nonsemantic).
  - The features describe superficial aspects of the data and preserve few of the invariances that one would want from a representation learning scheme.
- Modern supervised learning methods (i.e. with NN) learn layers of representations that learn the relevant axes of variance in the data.
  - Eg. Higher level features of a CNN trained to recognize car makes and models should be relatively invariance to color but very sensitive to subtle differences in shape.

Self-Supervised Learning

- A version of unsupervised learning where data provides the supervision.
- In general, withhold some part of the data and the task a neural network to predict it from the remaining parts.
- Details decide what proxy loss or pretext task the network tries to solve, and depending on the quality of the task, good semantic features can be obtained without actual labels.

Motivation

- Supervised learning success story is heavily because of the utility of pre-trained classifier features for commercially useful downstream tasks like segmentation, detection, etc.
- Recipe is clear: Collect a large labeled dataset, train a model, deploy. Good data and sufficient data are what you need.
- Goal of self-supervised learning:
  – Learn equally good (if not better) features without supervision
  – Be able to deploy similar quality systems without relying on too many labels for the downstream tasks
  – Generalize better potentially because you learn more about the world

How Much Information Does the Machine Need to Predict?

Y LeCun

- "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**

- Unsupervised/Predictive Learning (cake)
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**

- **(Yes, I know, this picture is slightly offensive to RL folks. But I'll make it up)**

- LeCun's original cake analogy slide, presented at his keynote speech in NIPS 2016.

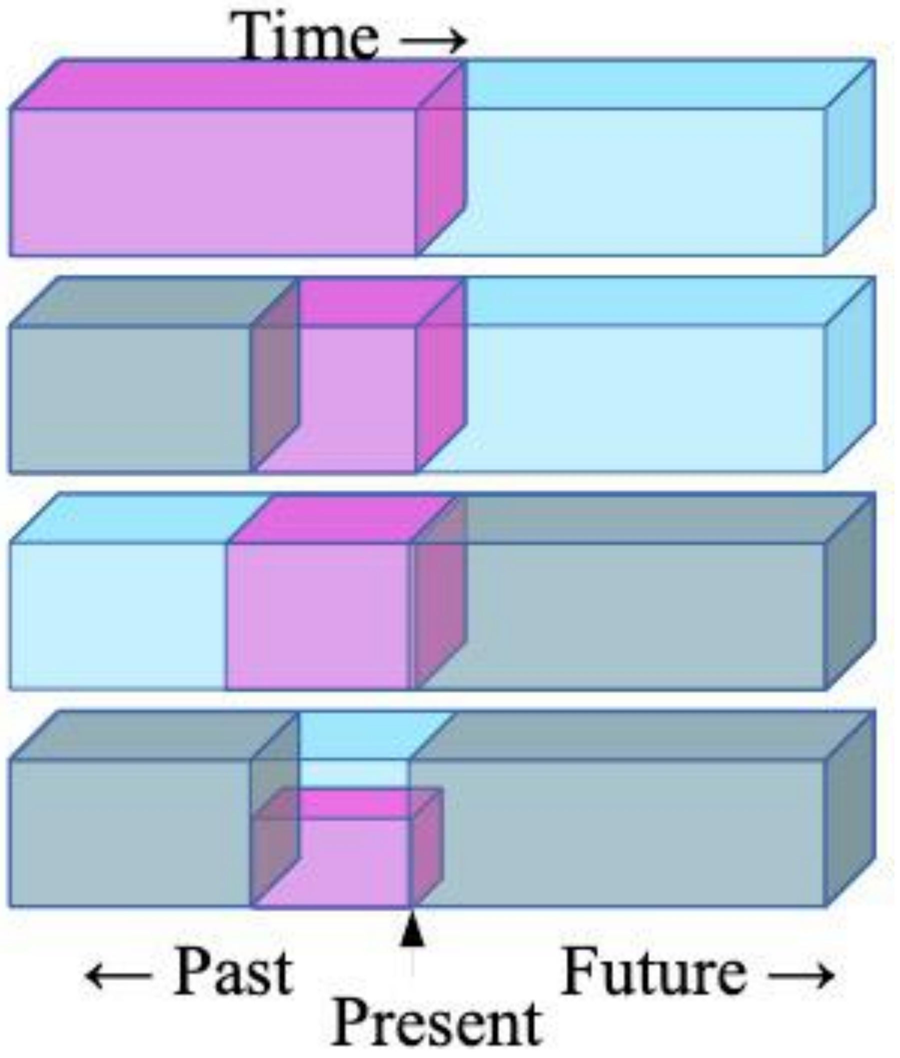
How Much Information is the Machine Given during Learning?

- **"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**

- Updated version at (ISSCC 2019, where he replaced "**unsupervised learning**" with "**self-supervised learning**".

Self-Supervised/Predictive Learning

- Predict any part of the input from any other part.
- Predict the future from the past.
- Predict the future from the recent past.
- Predict the past from the present.
- Predict the top from the bottom.
- Predict the occluded from the visible
- **Pretend there is a part of the input you don't know and predict that.**



What/Why Self-Supervision?

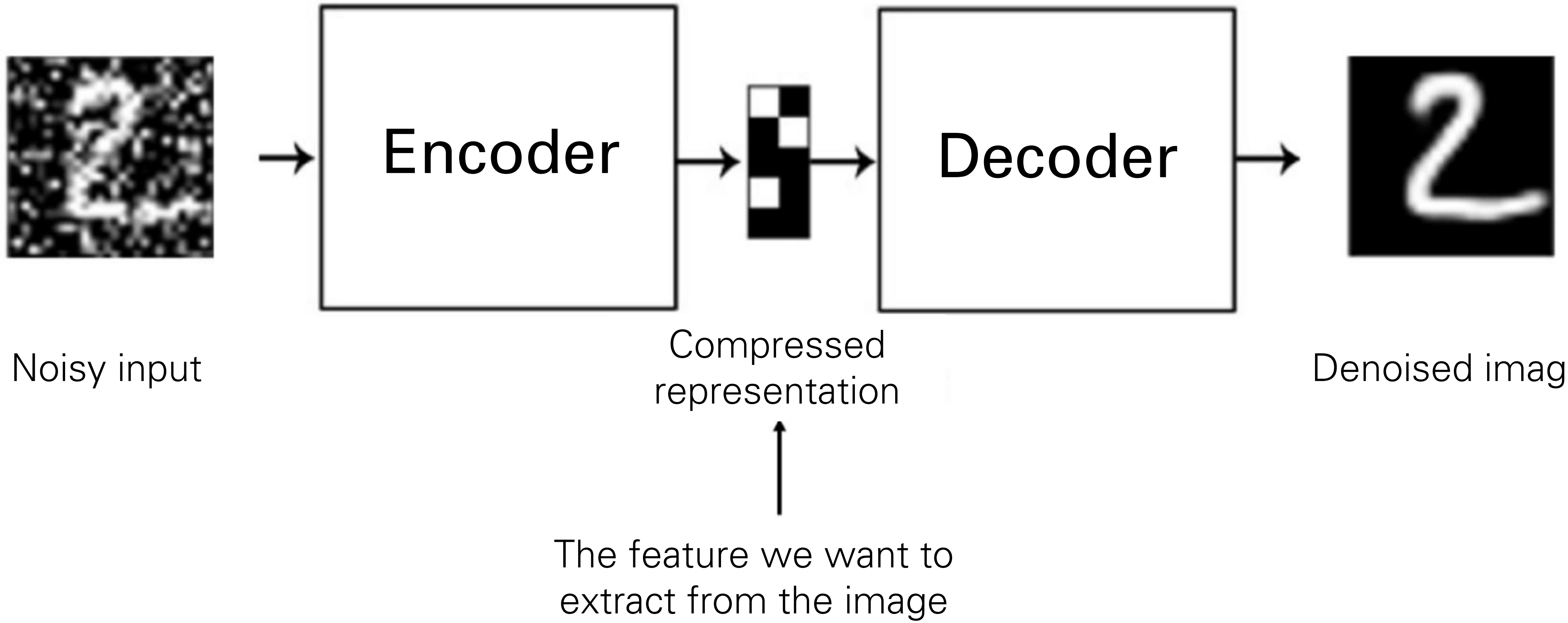
Self-supervision: Recover useful/semantic representations by training models to answer specific questions about the data.

- Good:
  - Can procedurally generate potentially infinite amounts of annotation.
  - We can borrow tricks from supervised learning without labels.
  - Focus on only the information that you need (e.g., not pixels).
  - **Answering these questions requires more fundamental understanding of data.**
- Not so good: designing good questions also requires some fundamental understanding of the data (e.g., structure).

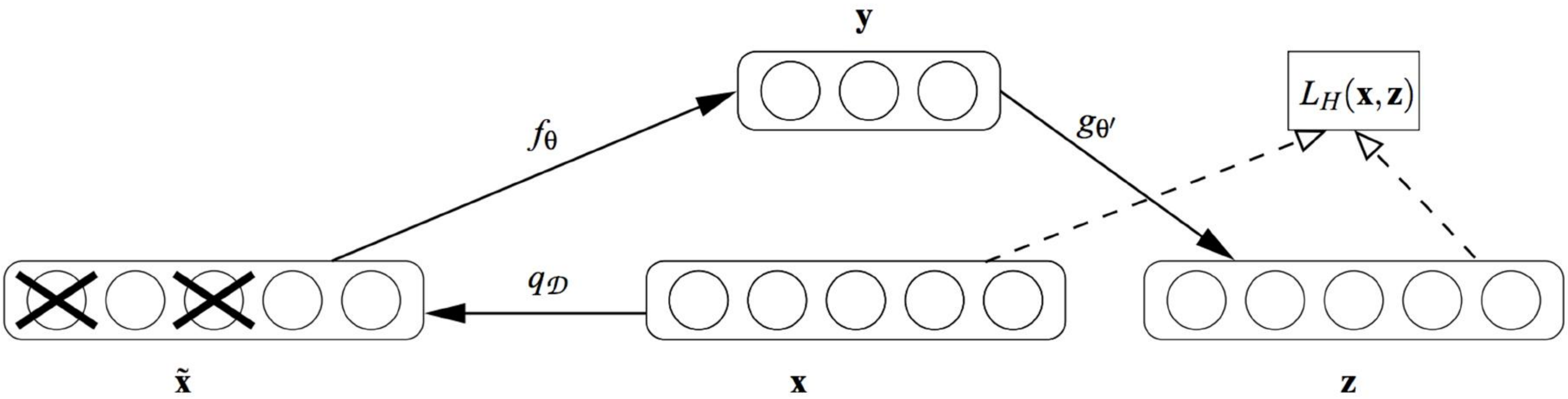
Lecture overview

- Motivation
- Reconstruct from a corrupted (or partial) version
  - Denoising Autoencoder
  - In-painting
  - Colorization, Split-Brain Autoencoder
- Proxy tasks in computer vision
- Contrastive Learning

Denoising Autoencoder



Denoising Autoencoder



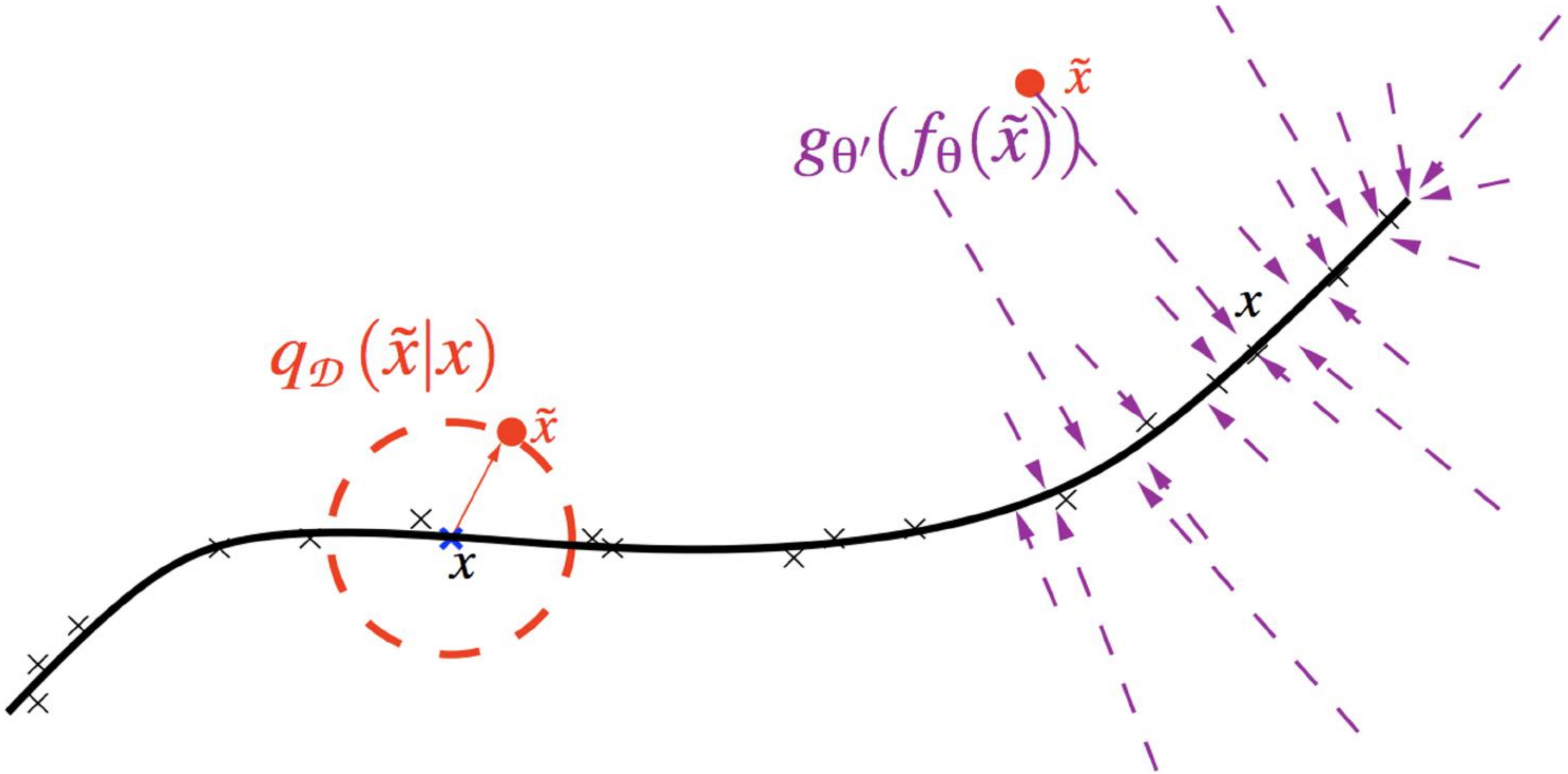
Vincent et al. 2010

Denoising Autoencoder

- 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.

Vincent et al. 2010

Denoising Autoencoder



Vincent et al. 2010

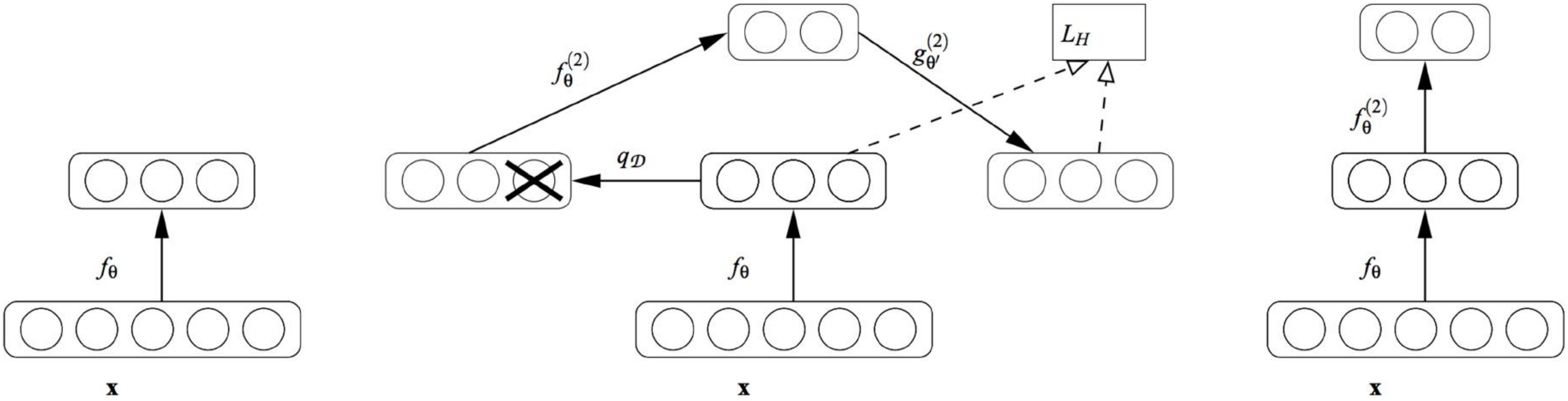
Emphasizing corrupted dimensions

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

$$\begin{aligned} L_{\mathbb{H},\alpha}(\mathbf{x},\mathbf{z}) \quad = \quad & \alpha\left(-\sum_{j\in\mathcal{J}(\tilde{\mathbf{x}})}[\mathbf{x}_j\log\mathbf{z}_j+(1-\mathbf{x}_j)\log(1-\mathbf{z}_j)]\right) \\ & +\beta\left(-\sum_{j\notin\mathcal{J}(\tilde{\mathbf{x}})}[\mathbf{x}_j\log\mathbf{z}_j+(1-\mathbf{x}_j)\log(1-\mathbf{z}_j)]\right) \end{aligned}$$

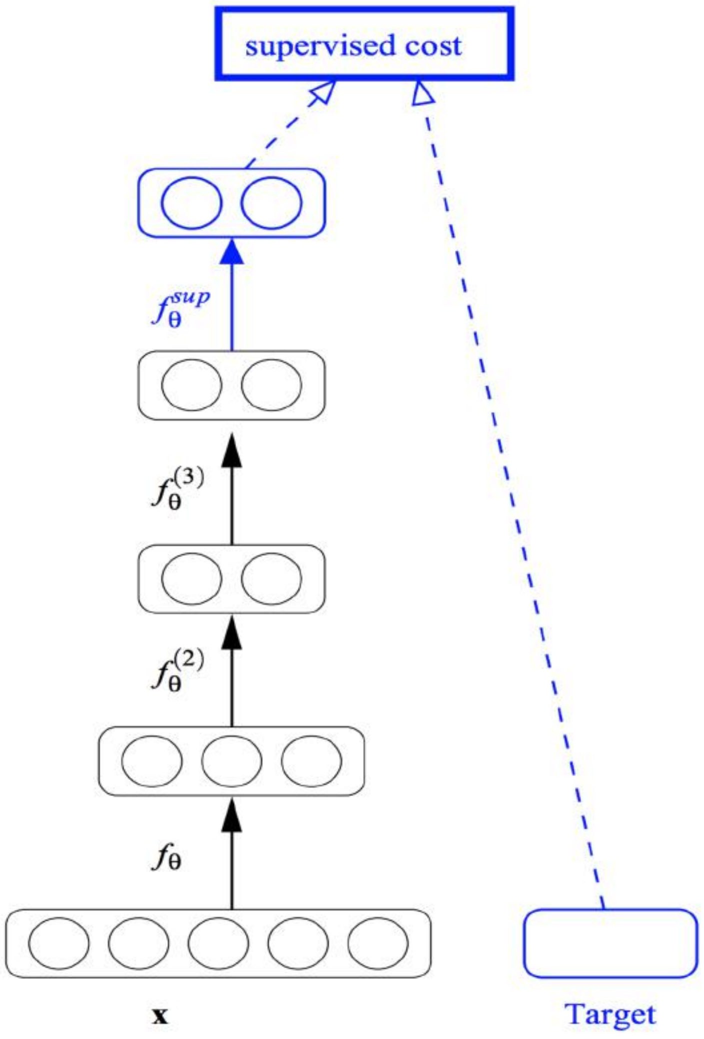
Vincent et al. 2010

Stacked Denoising Autoencoder



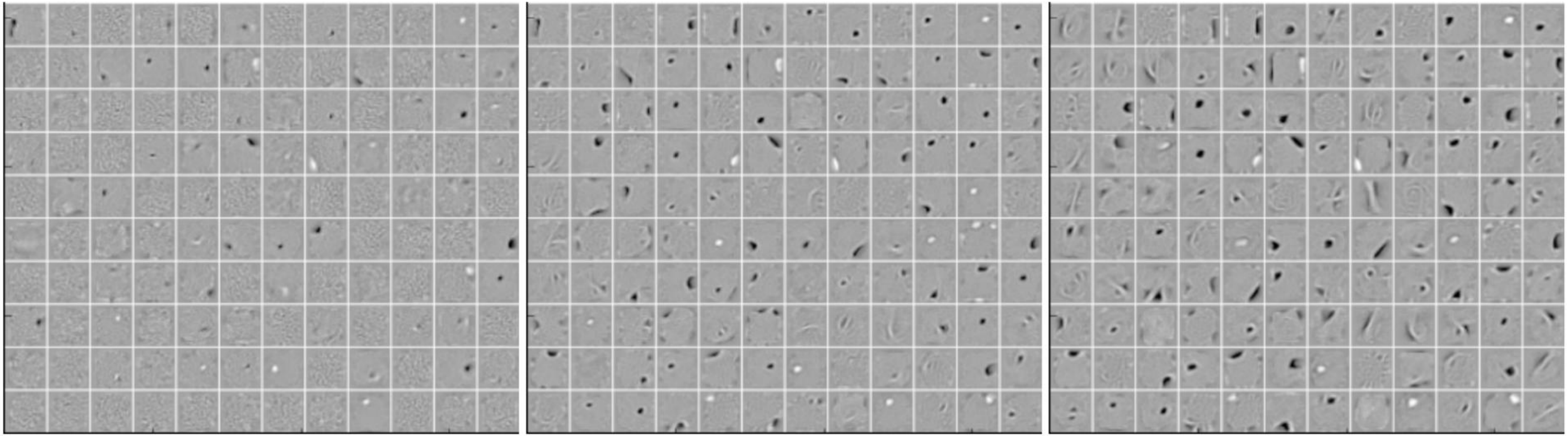
Vincent et al. 2010

Denoising Autoencoder



Vincent et al. 2010

Denoising Autoencoder



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

Vincent et al. 2010

Denoising Autoencoder

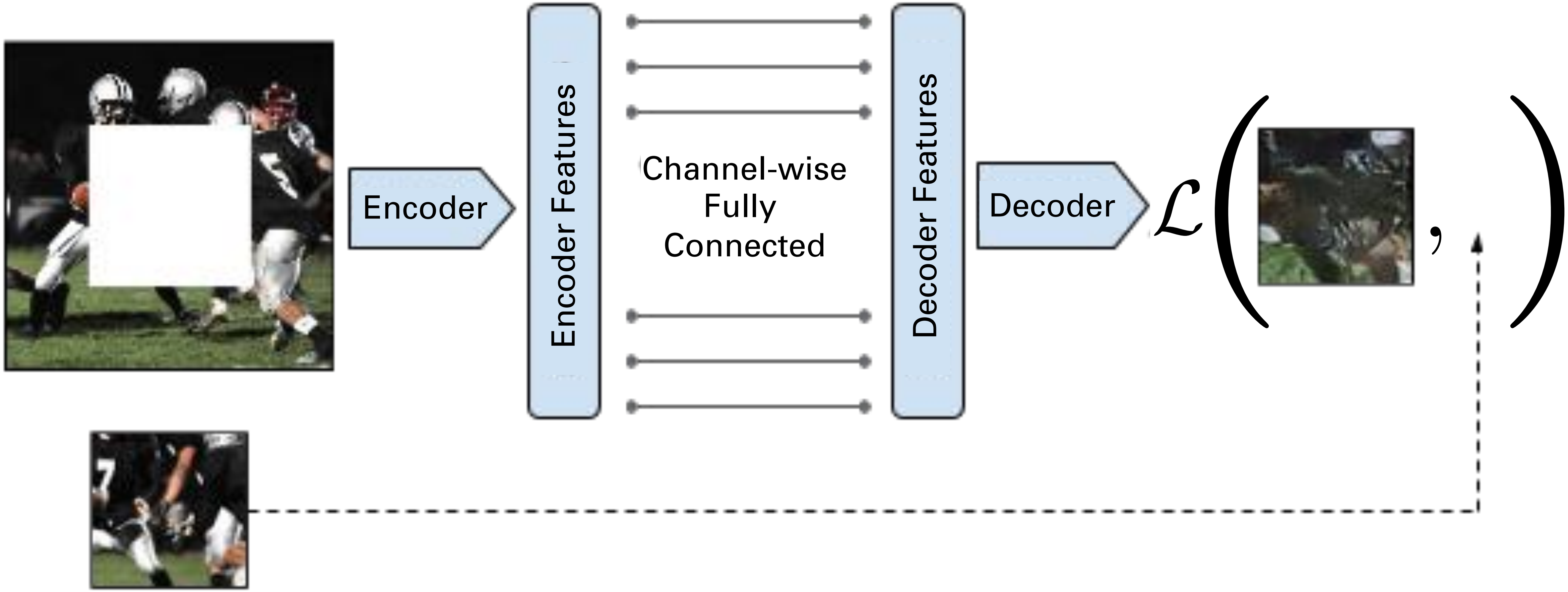
Dataset	**SVM**$_{rbf}$	**SVM**$_{poly}$	**DBN-1**	**SAA-3**	**DBN-3**	**SdA-3** (ν)
basic	**3.03±0.15**	3.69±0.17	3.94±0.17	3.46±0.16	3.11±0.15	**2.80±0.14** (10%)
rot	11.11±0.28	15.42±0.32	14.69±0.31	**10.30±0.27**	**10.30±0.27**	**10.29±0.27** (10%)
bg-rand	14.58±0.31	16.62±0.33	9.80±0.26	11.28±0.28	**6.73±0.22**	10.38±0.27 (40%)
bg-img	22.61±0.37	24.01±0.37	**16.15±0.32**	23.00±0.37	**16.31±0.32**	**16.68±0.33** (25%)
rot-bg-img	55.18±0.44	56.41±0.43	52.21±0.44	51.93±0.44	47.39±0.44	**44.49±0.44** (25%)
rect	**2.15±0.13**	**2.15±0.13**	4.71±0.19	2.41±0.13	2.60±0.14	**1.99±0.12** (10%)
rect-img	24.04±0.37	24.05±0.37	23.69±0.37	24.05±0.37	22.50±0.37	**21.59±0.36** (25%)
convex	19.13±0.34	19.82±0.35	19.92±0.35	**18.41±0.34**	**18.63±0.34**	**19.06±0.34** (10%)

Vincent et al. 2010

Predict missing pieces

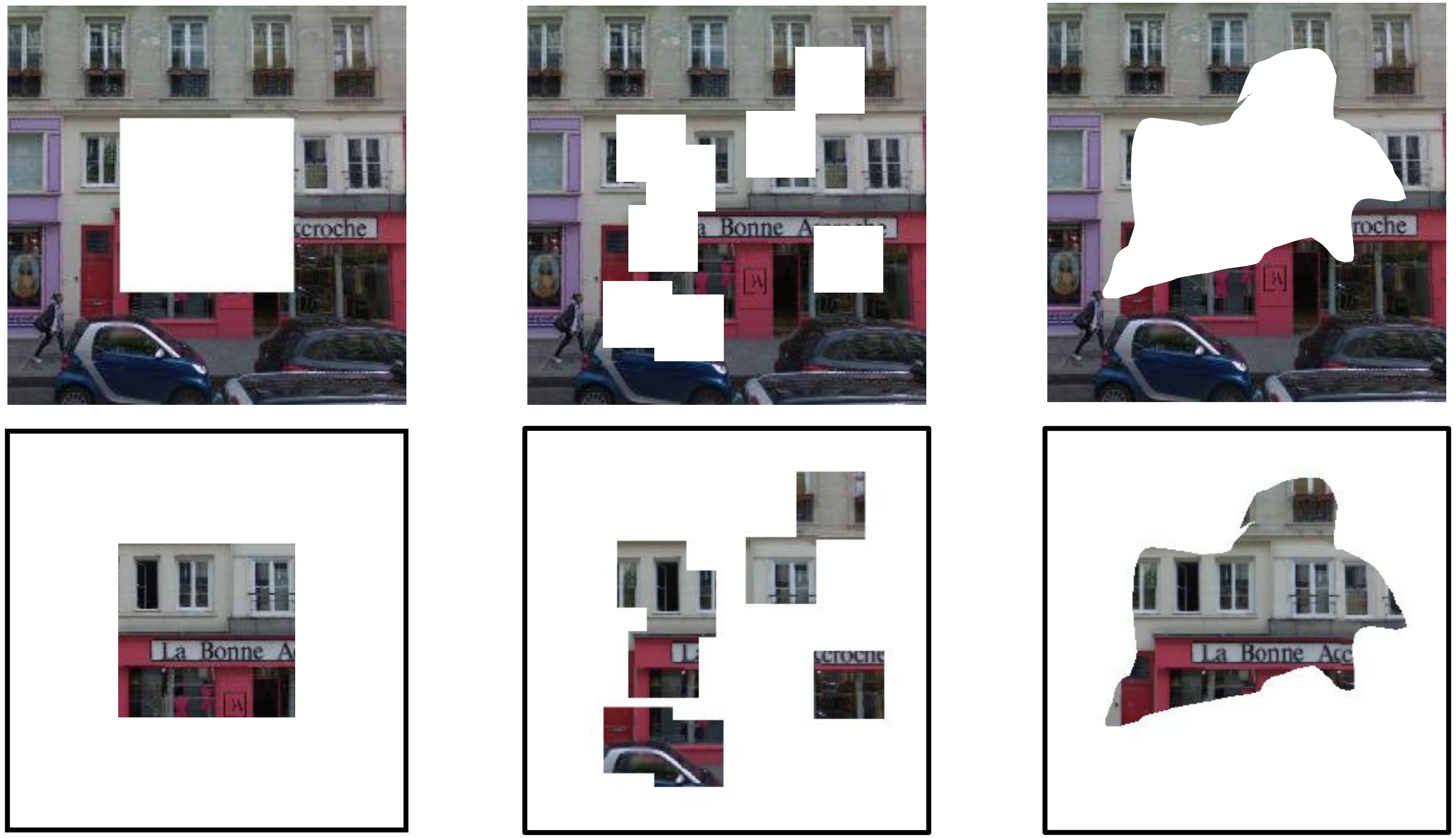
Pathak et al. 2016

Context Encoders



Pathak et al. 2016

Context Encoders



(a) Center Region (b) Random Blocks (c) Random Shapes

Pathak et al. 2016

Context Encoders

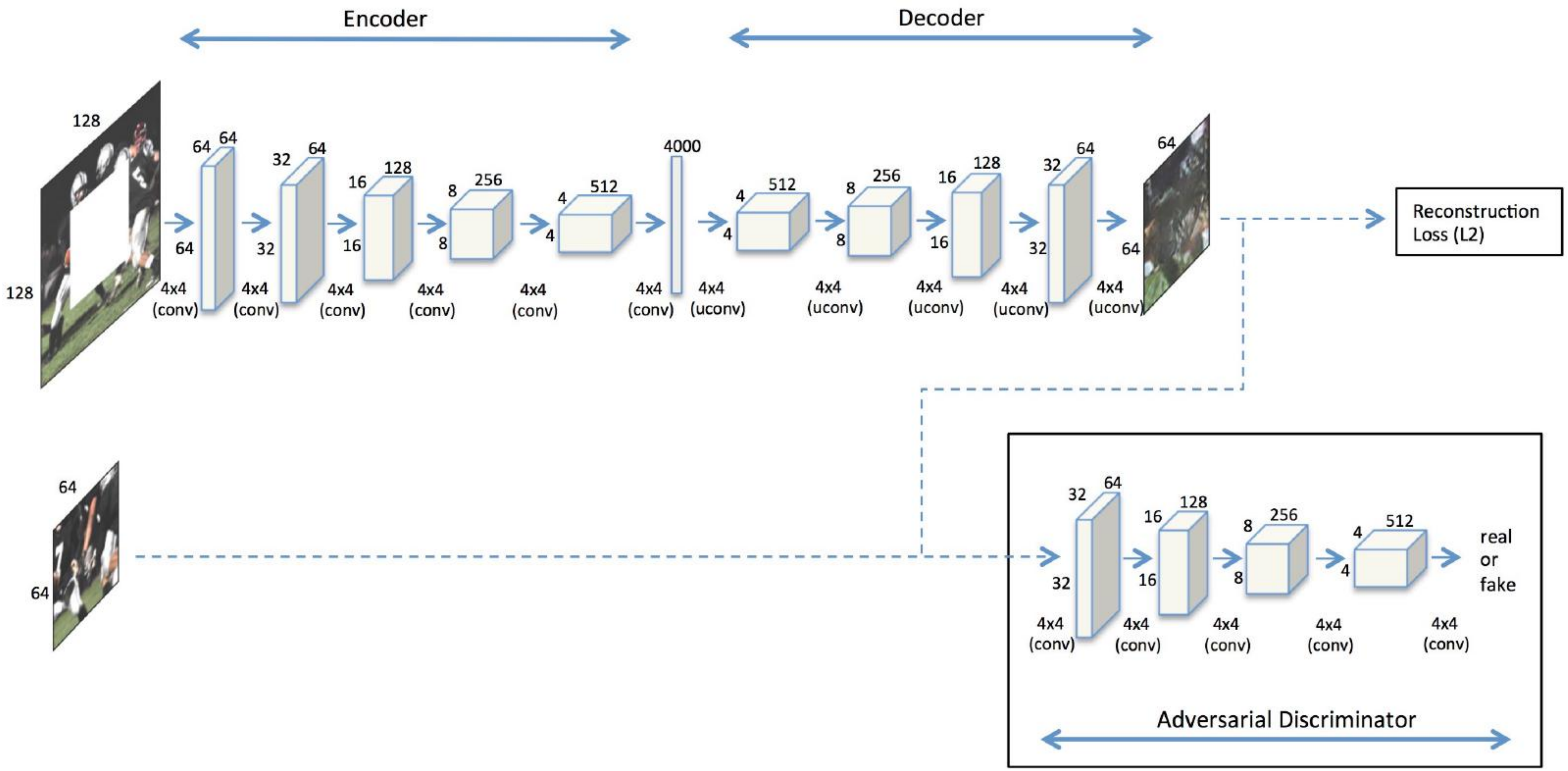
$$\mathcal{L}_{rec}(x) = \| \hat{M} \odot (x - F((1 - \hat{M}) \odot x)) \|_2^2$$

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

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

Pathak et al. 2016

Context Encoders



Pathak et al. 2016

Context Encoders

Input Image L2 Loss Adversarial Loss Joint Loss

Pathak et al. 2016

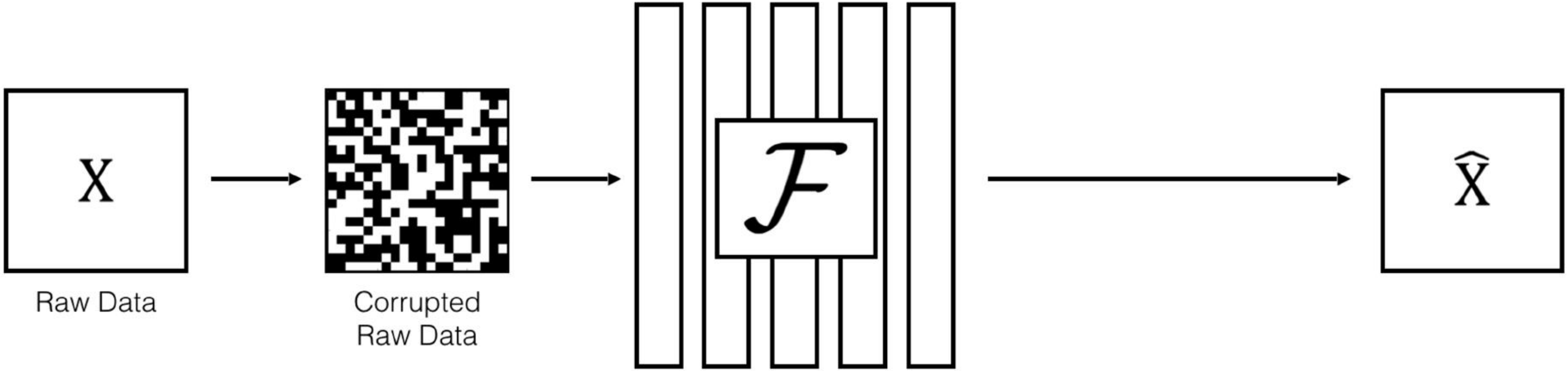
Context Encoders

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].

Pathak et al. 2016

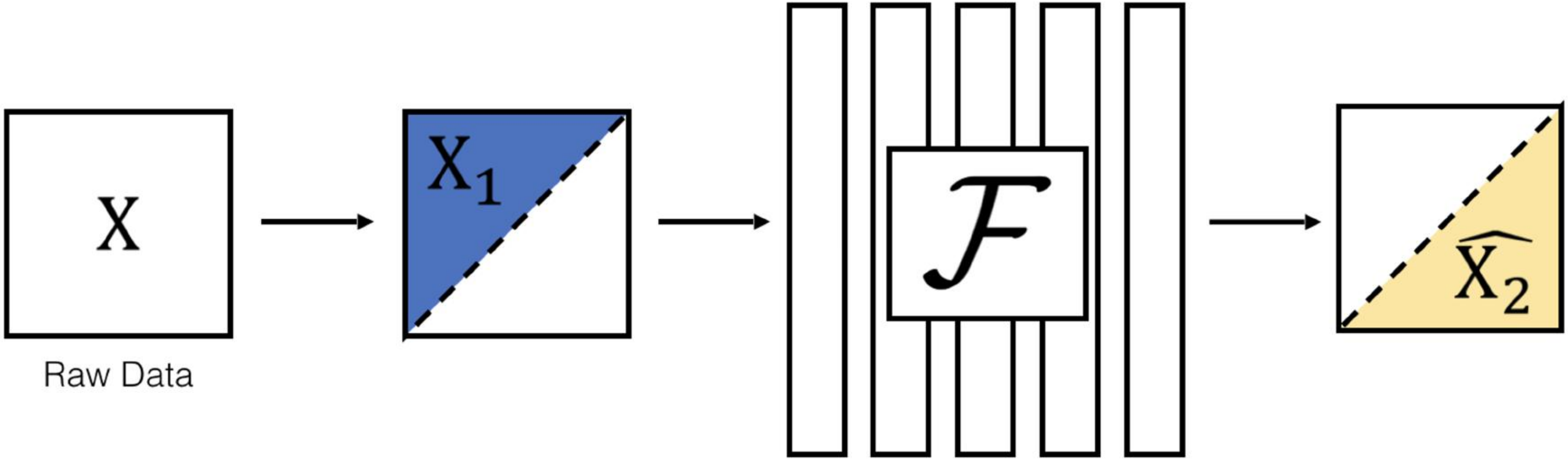
Predicting one view from another



Denoising Autoencoder

Slide credit: Richard Zhang

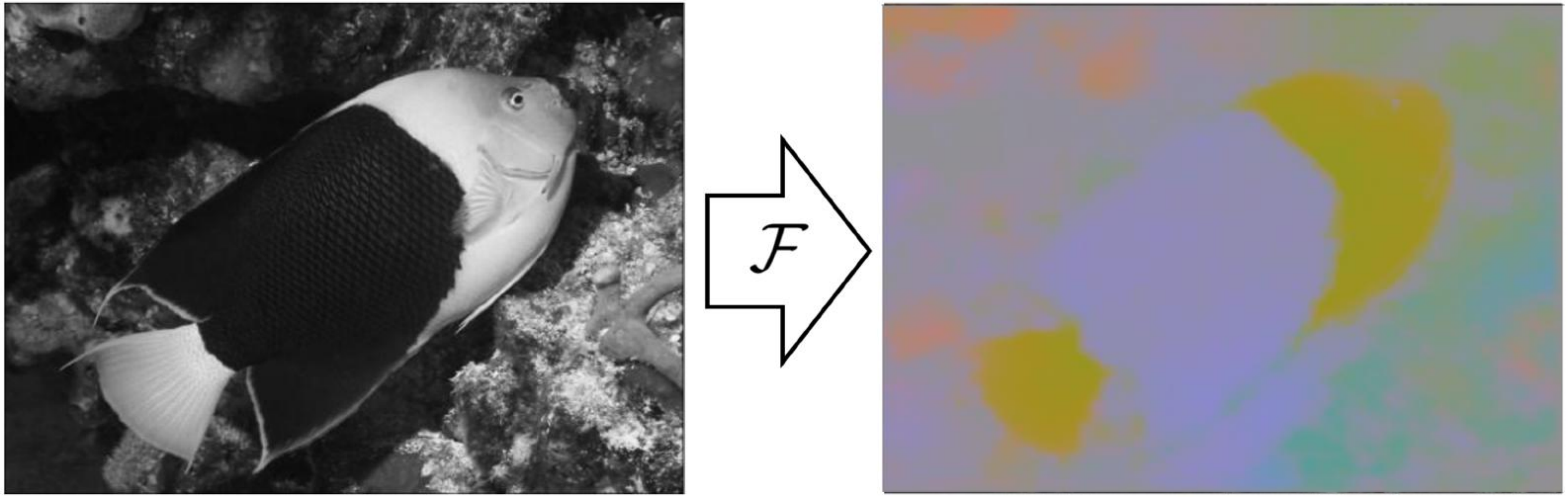
Predicting one view from another



Cross-Channel Autoencoder

Slide credit: Richard Zhang

Predicting one view from another

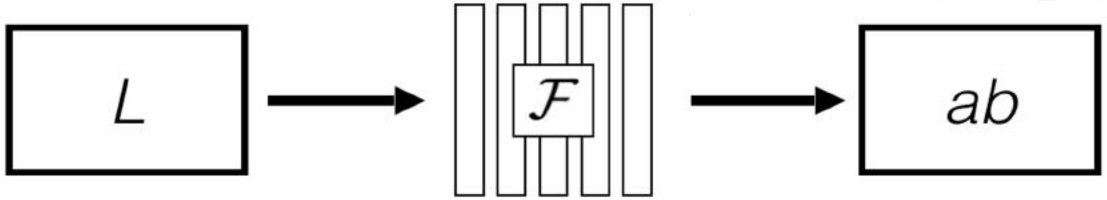


Grayscale image: *L* channel

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

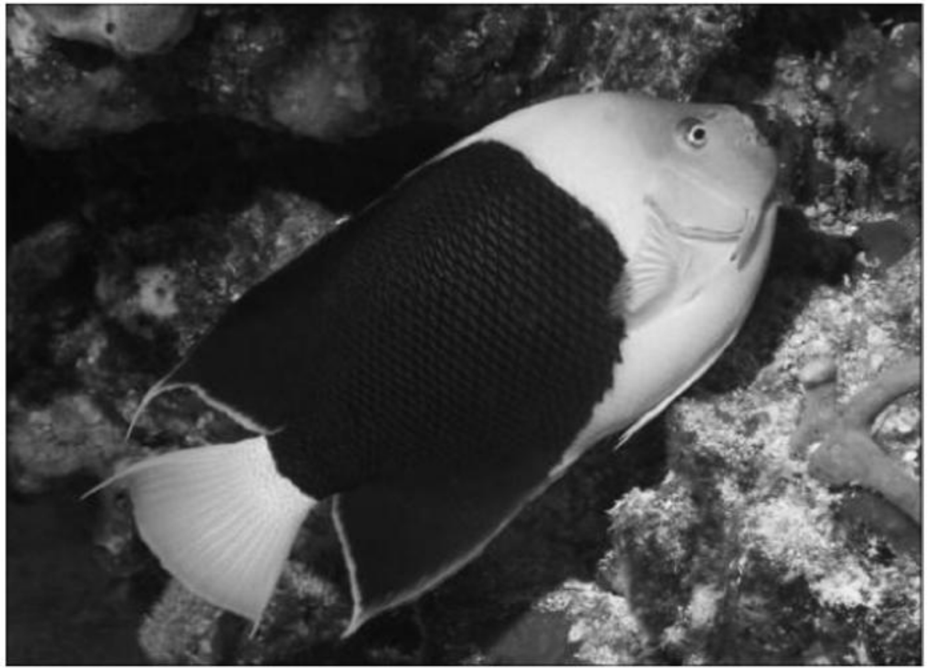
Color information: *ab* channels

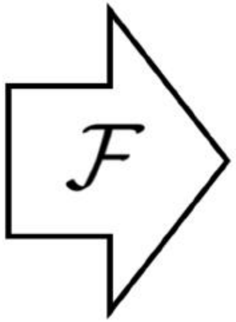
$\widehat{\mathbf{Y}} \in \mathbb{R}^{H \times W \times 2}$



Slide credit: Richard Zhang

Predicting one view from another



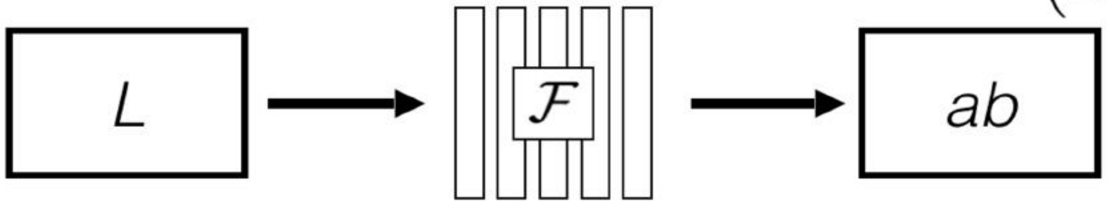


Grayscale image: *L* channel

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

Concatenate (*L*,*ab*) channels

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



Slide credit: Richard Zhang

Predicting one view from another

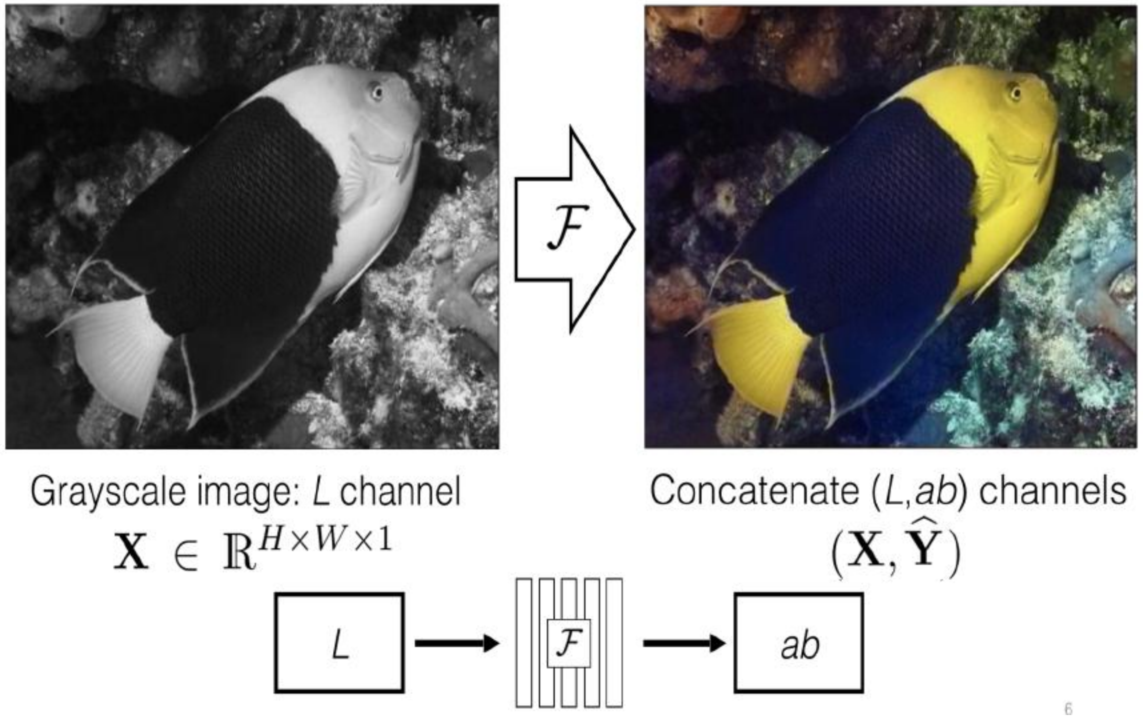
Ground Truth

L2 regression

Pixelwise classification

Slide credit: Richard Zhang

Predicting one view from another



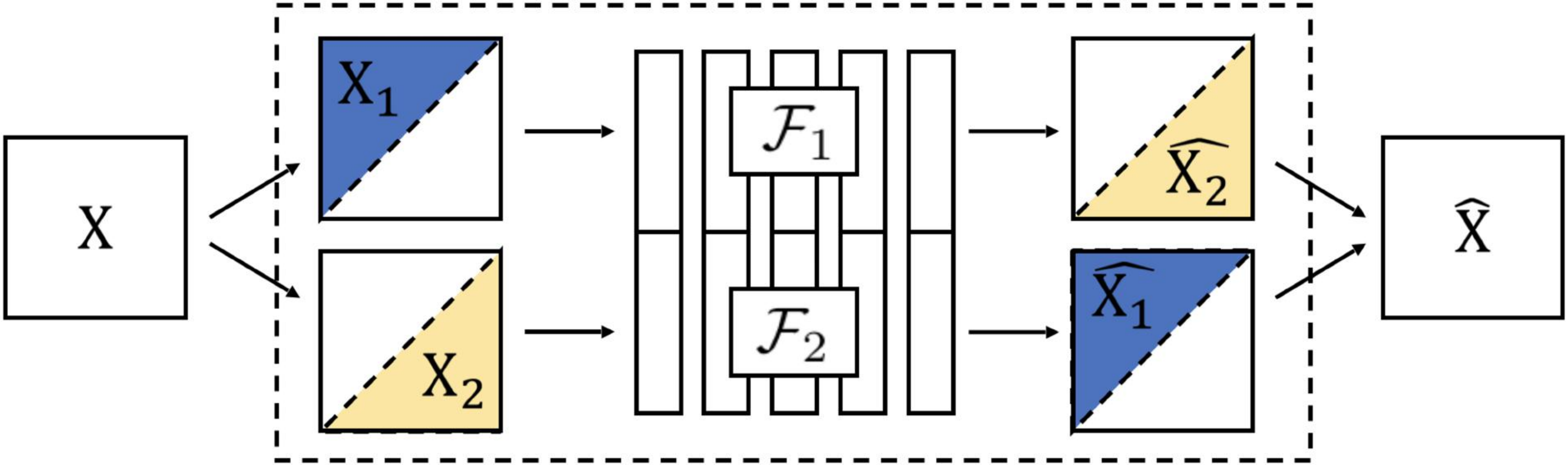
$$\mathrm{L}_2(\widehat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w}\|_2^2$$

$$\downarrow$$

$$\mathrm{L}(\widehat{\mathbf{Z}}, \mathbf{Z}) = -\frac{1}{HW} \sum_{h,w} \sum_{q} \mathbf{Z}_{h,w,q} \log\left(\widehat{\mathbf{Z}}_{h,w,q}\right)$$

Slide credit: Richard Zhang

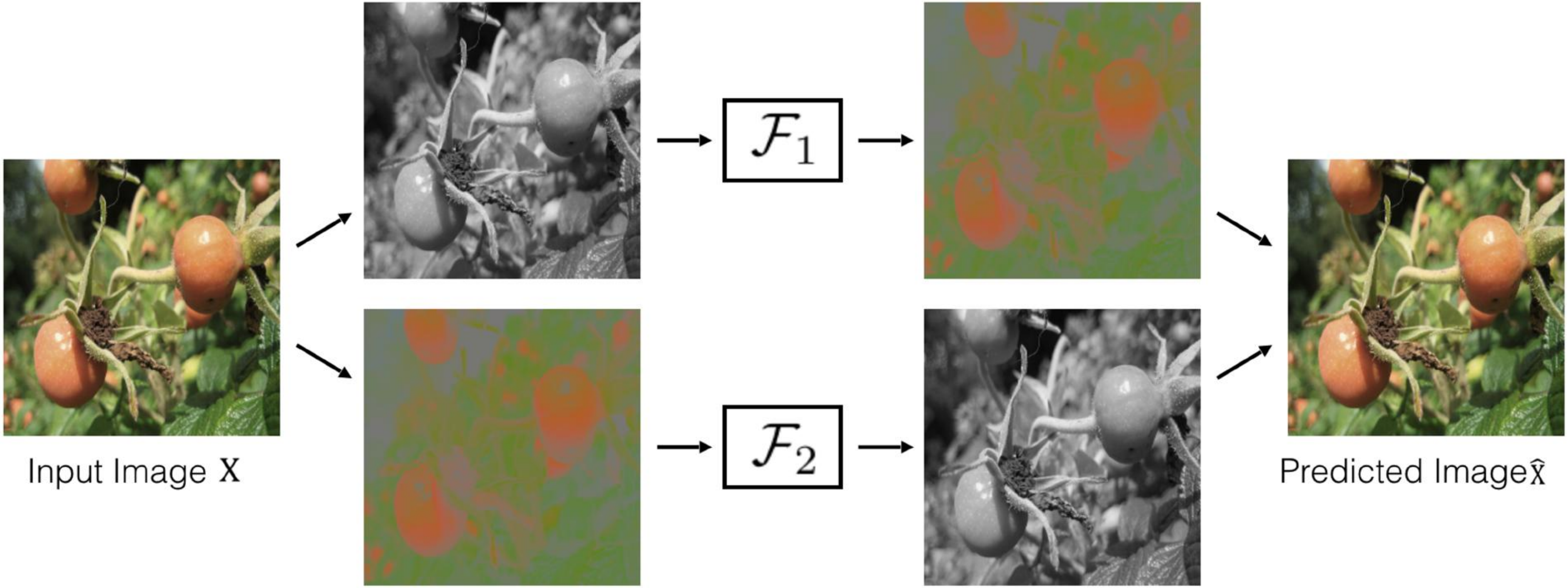
Predicting one view from another



Split-Brain Autoencoder

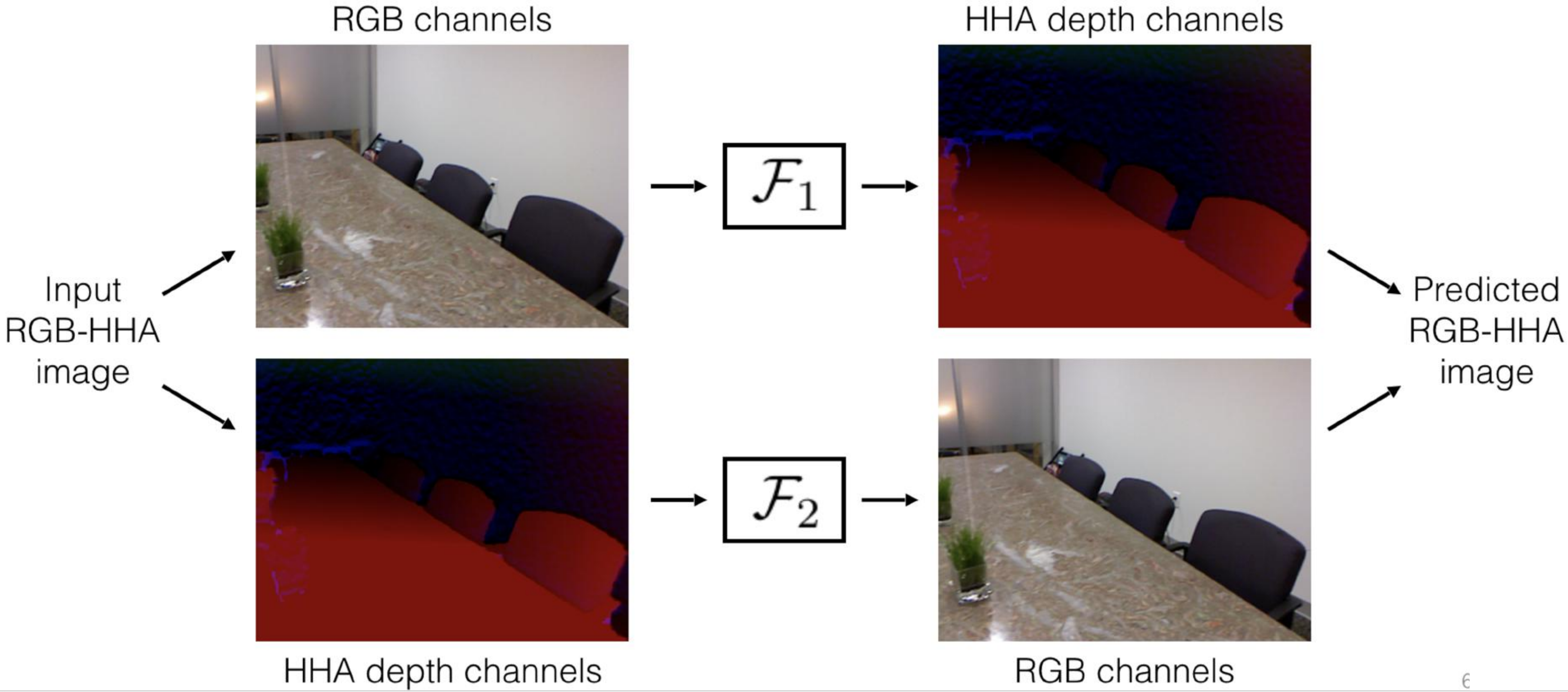
Slide credit: Richard Zhang

Predicting one view from another



Slide credit: Richard Zhang

Predicting one view from another



Slide credit: Richard Zhang

Lecture overview

- Motivation
- Reconstruct from a corrupted (or partial) version
- **Proxy tasks in computer vision**
  - **Relative patch prediction**
  - **Jigsaw puzzles**
  - **Rotation**
- Contrastive Learning

Relative Position of Image Patches

Unsupervised Visual Representation Learning by Context Prediction

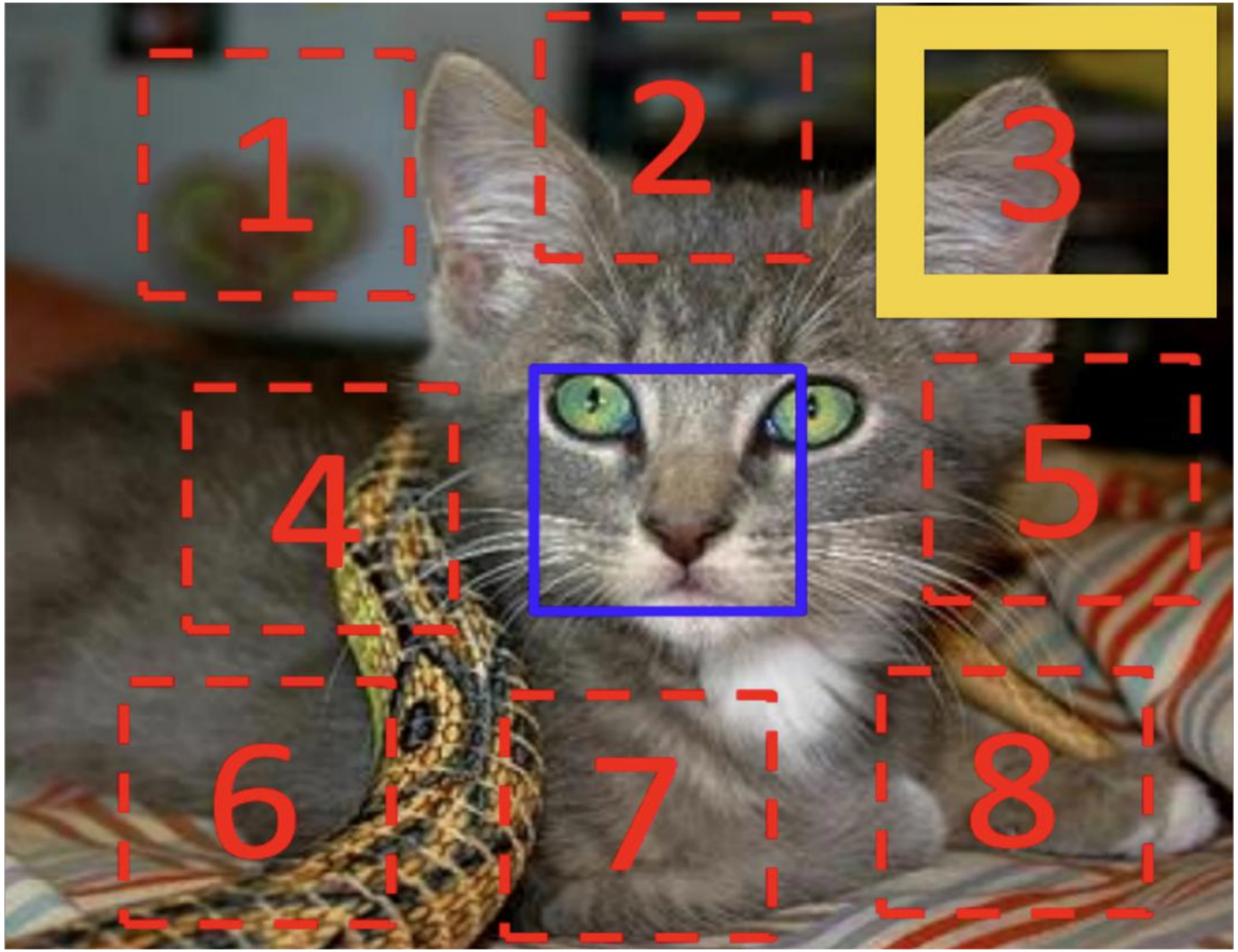
Carl Doersch[1,2] Abhinav Gupta[1] Alexei A. Efros[2]
[1] School of Computer Science, Carnegie Mellon University
[2] Dept. of Electrical Engineering and Computer Science, University of California, Berkeley

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

Slide credit: Andrew Zisserman

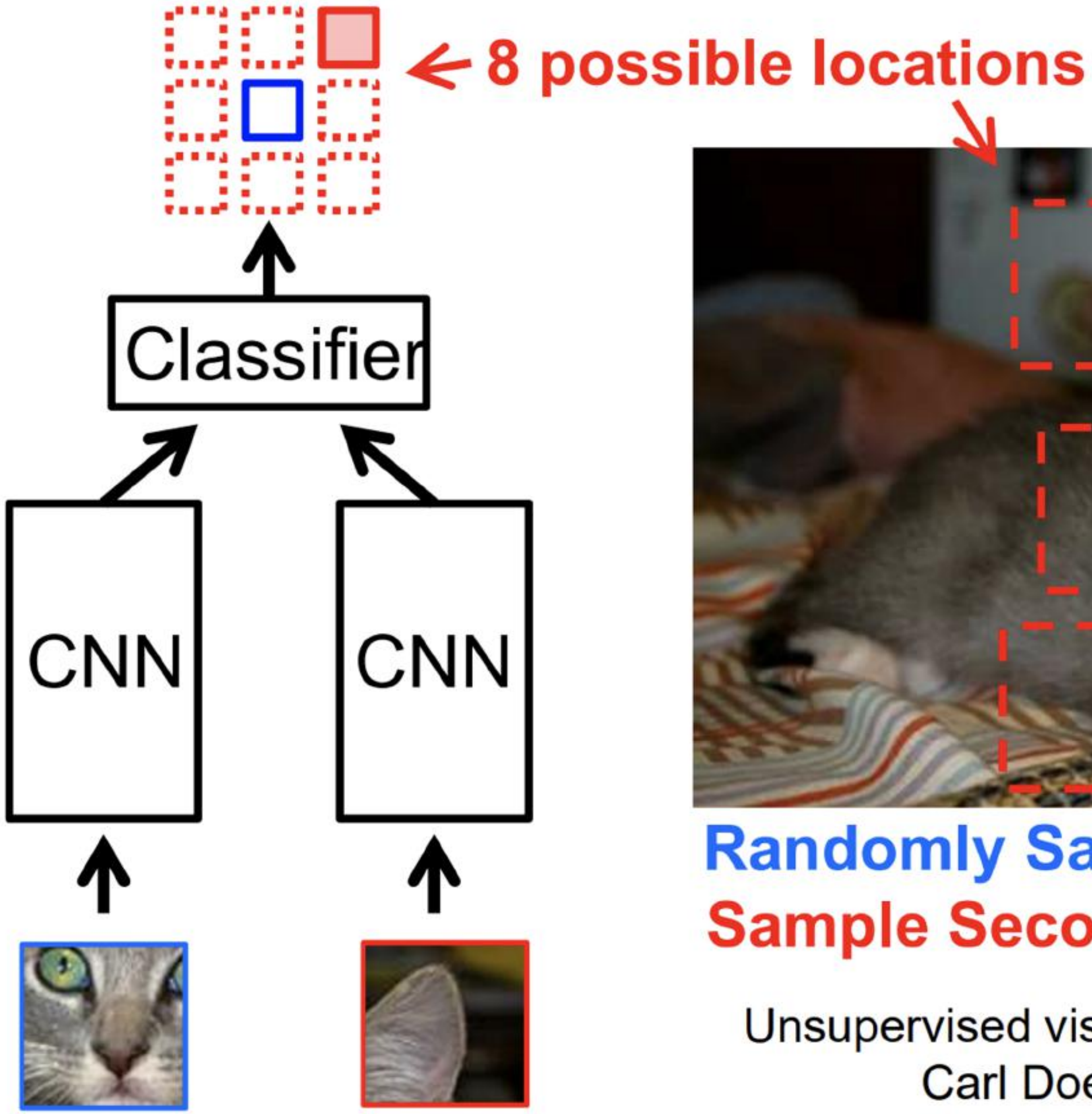
Relative Position of Image Patches

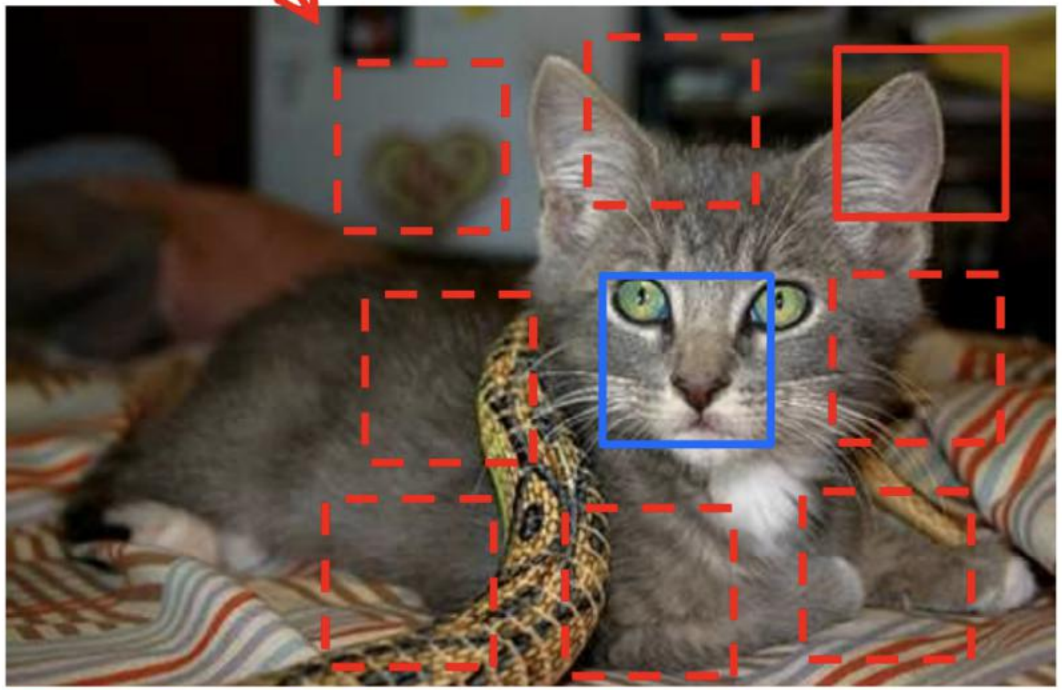


Doersch, Gupta, Efros

Slide credit: Andrew Zisserman

Relative Position of Image Patches

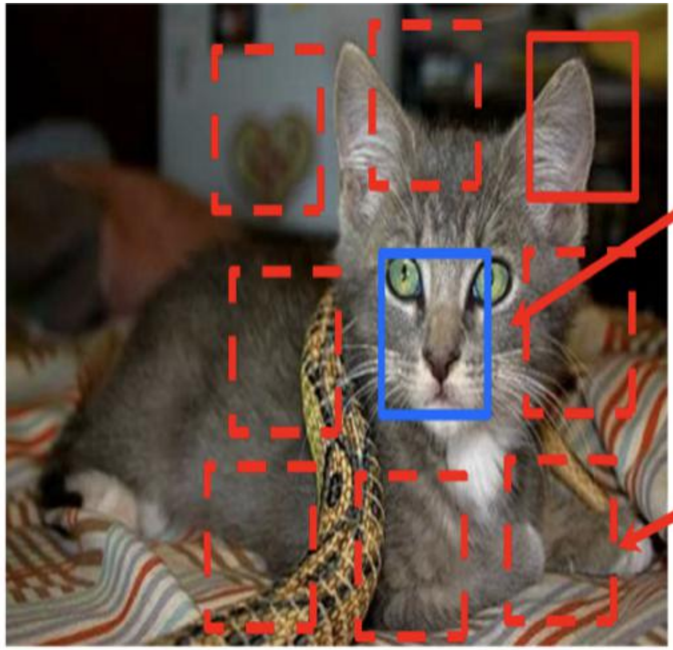




Randomly Sample Patch
Sample Second Patch

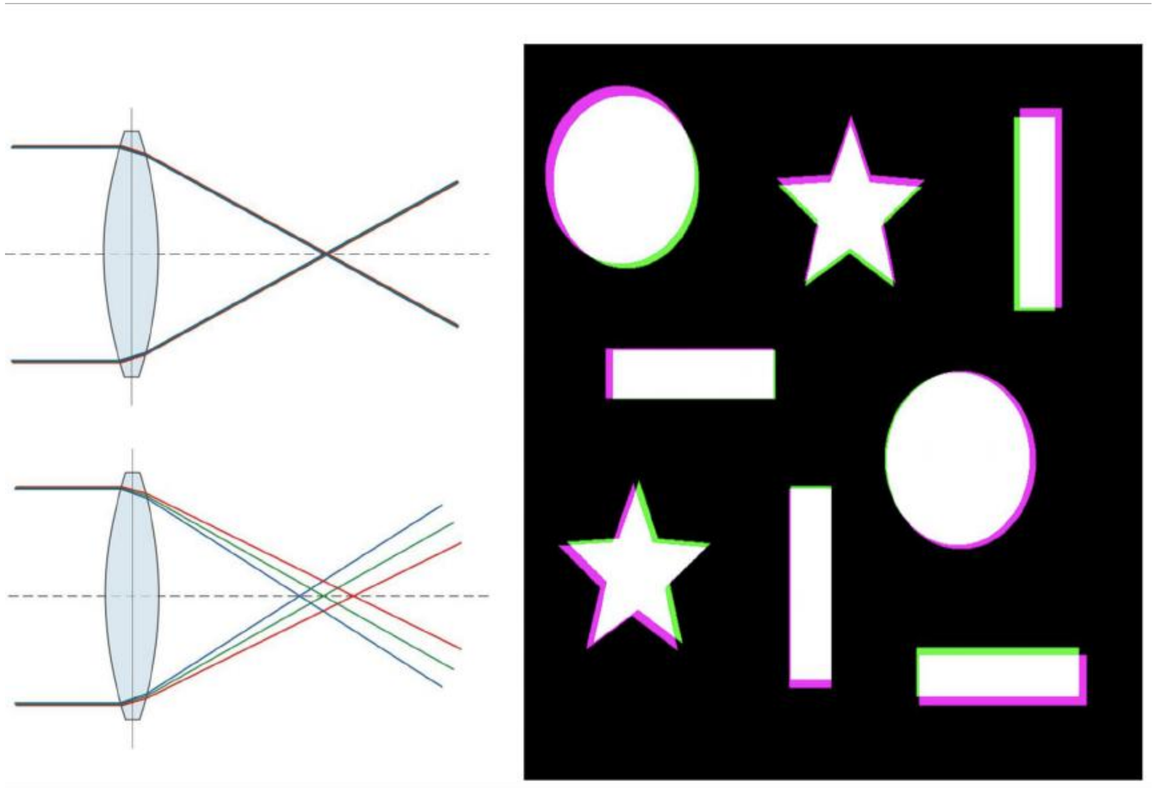
Unsupervised visual representation learning by context prediction,
Carl Doersch, Abhinav Gupta, Alexei A. Efros, ICCV 2015

Relative Position of Image Patches

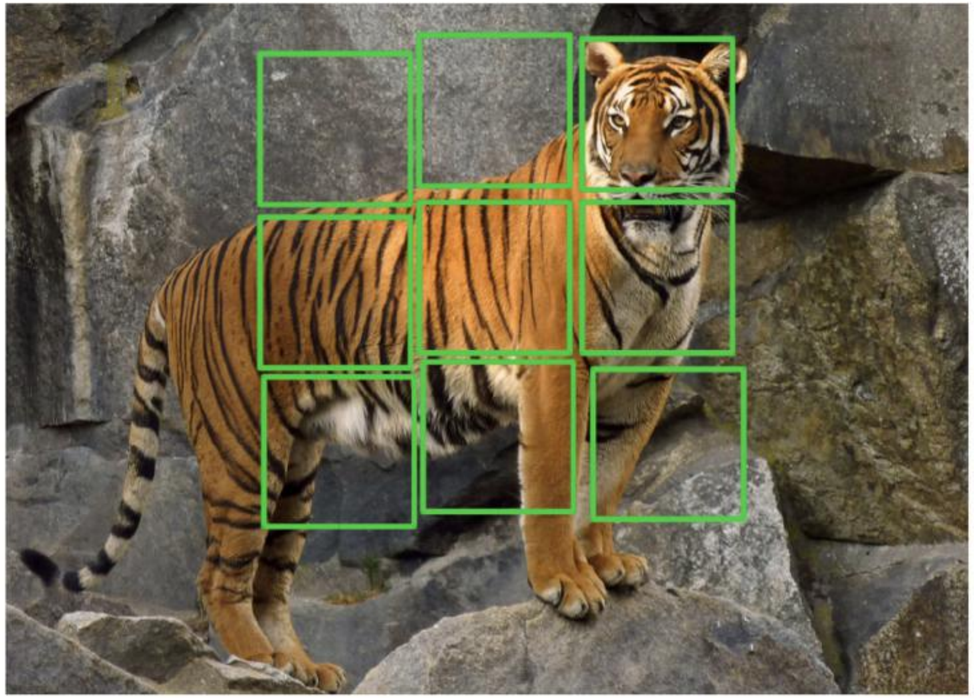


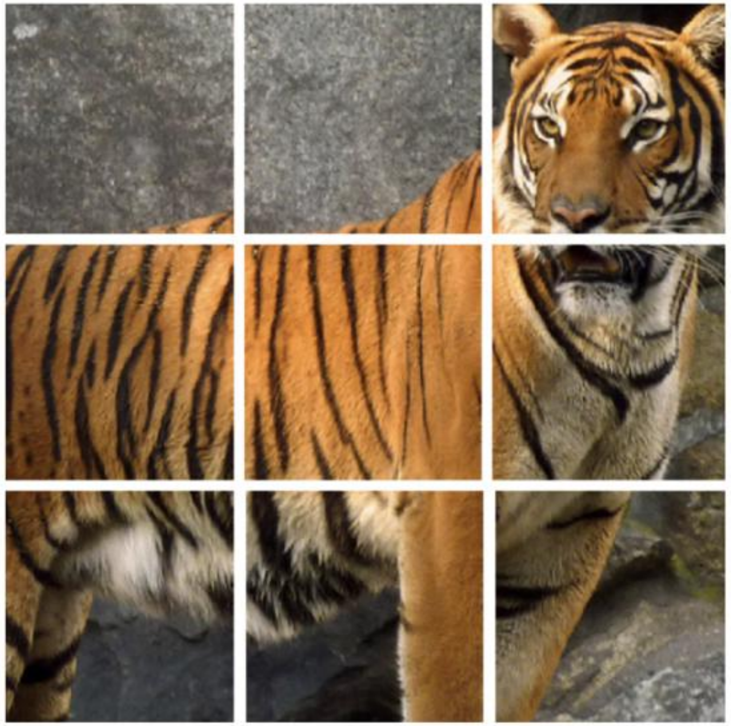
Include a gap

Jitter the patch locations



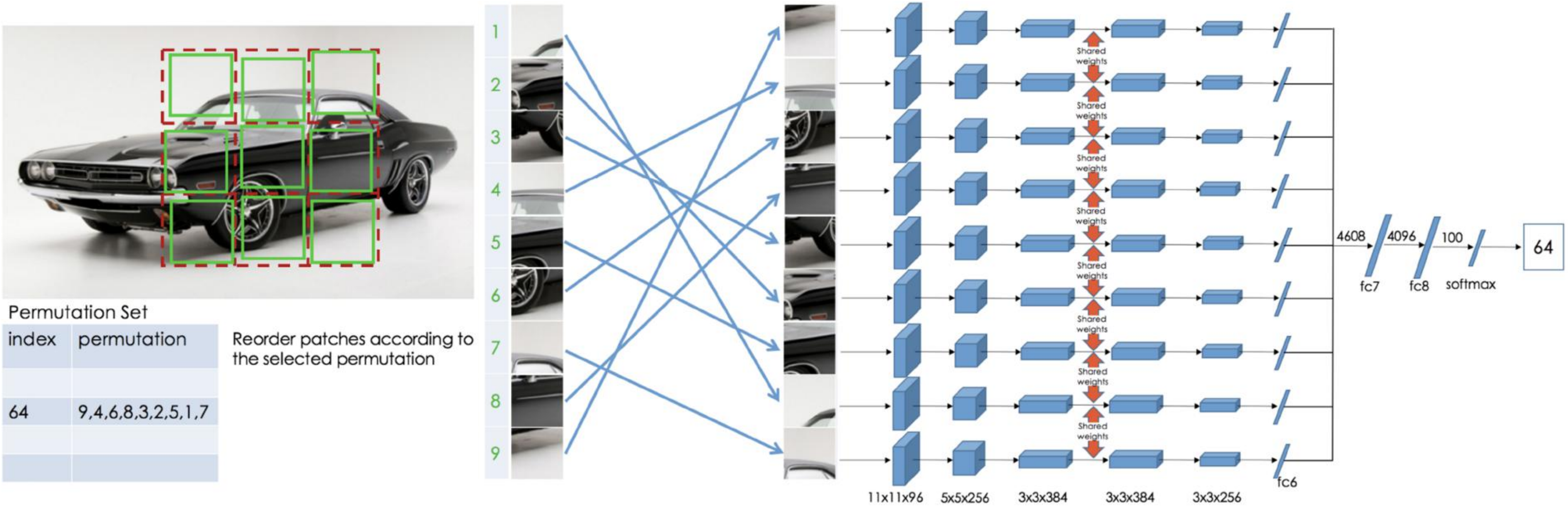
Solving Jigsaw Puzzles





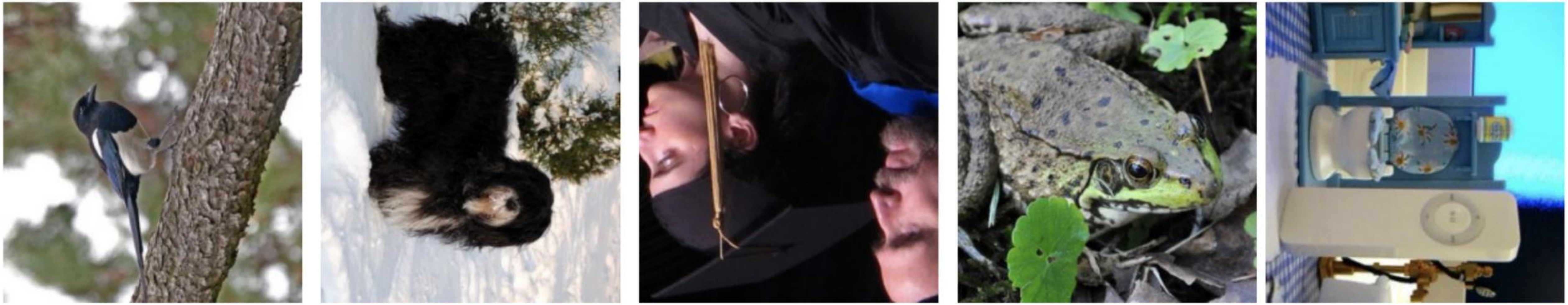
Noroozi and Favaro, 2016

Solving Jigsaw Puzzles



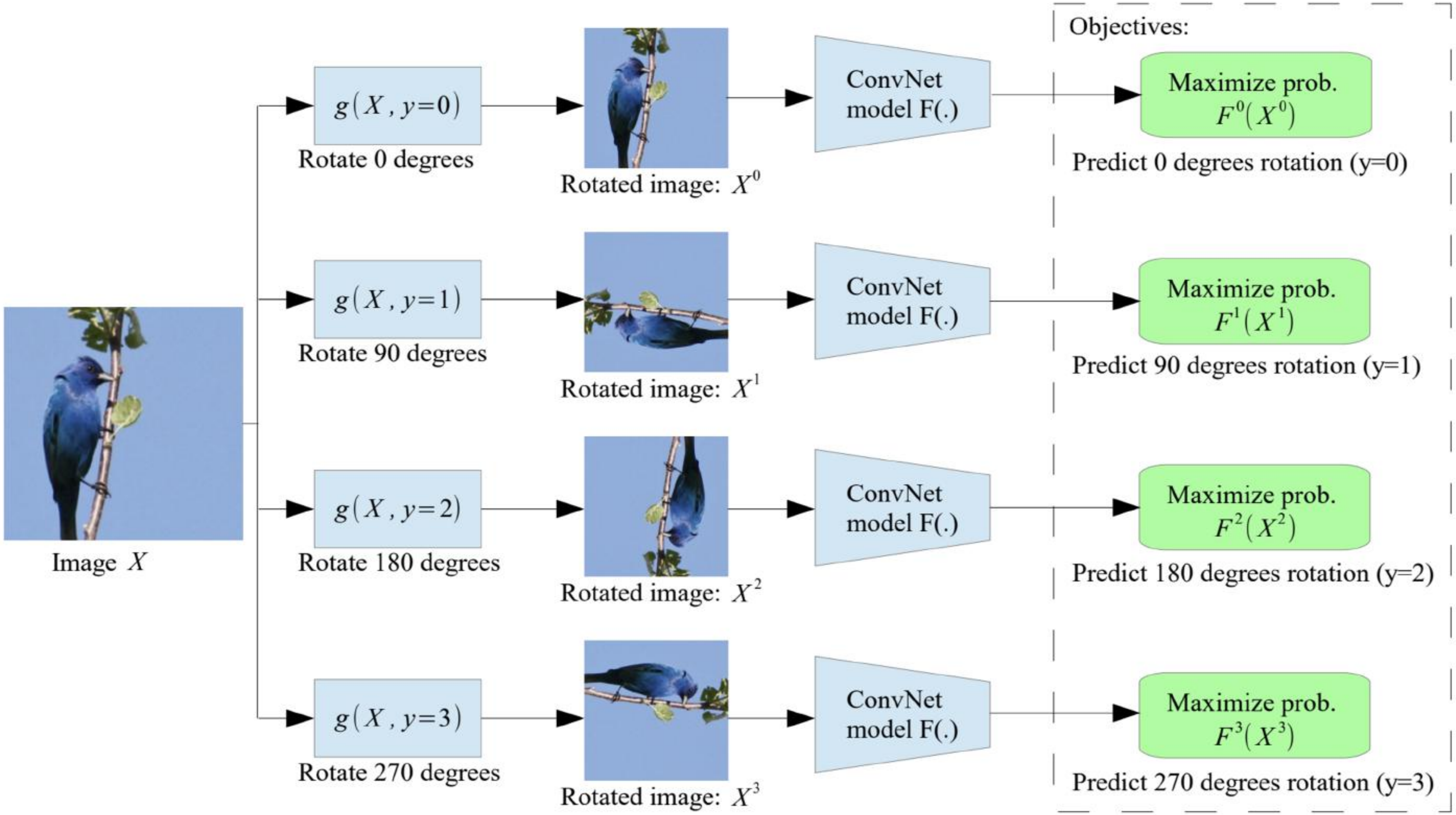
Noroozi and Favaro, 2016

Rotation



90° rotation | 270° rotation | 180° rotation | 0° rotation | 270° rotation

Rotation



Gidaris, Singh, and Komodakis. Unsupervised representation learning by predicting image rotations. ICLR 2018

Rotation

# 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

Method	Conv4	Conv5
ImageNet labels from (Bojanowski & Joulin, 2017)	59.7	59.7
Random from (Noroozi & Favaro, 2016)	27.1	12.0
Tracking Wang & Gupta (2015)	38.8	29.8
Context (Doersch et al., 2015)	45.6	30.4
Colorization (Zhang et al., 2016a)	40.7	35.2
Jigsaw Puzzles (Noroozi & Favaro, 2016)	45.3	34.6
BIGAN (Donahue et al., 2016)	41.9	32.2
NAT (Bojanowski & Joulin, 2017)	-	36.0
(Ours) RotNet	50.0	43.8

Rotation

Method	Conv1	Conv2	Conv3	Conv4	Conv5
ImageNet labels	19.3	36.3	44.2	48.3	50.5
Random	11.6	17.1	16.9	16.3	14.1
Random rescaled Krähenbühl et al. (2015)	17.5	23.0	24.5	23.2	20.6
Context (Doersch et al., 2015)	16.2	23.3	30.2	31.7	29.6
Context Encoders (Pathak et al., 2016b)	14.1	20.7	21.0	19.8	15.5
Colorization (Zhang et al., 2016a)	12.5	24.5	30.4	31.5	30.3
Jigsaw Puzzles (Noroozi & Favaro, 2016)	18.2	28.8	34.0	33.9	27.1
BIGAN (Donahue et al., 2016)	17.7	24.5	31.0	29.9	28.0
Split-Brain (Zhang et al., 2016b)	17.7	29.3	35.4	35.2	32.8
Counting (Noroozi et al., 2017)	18.0	30.6	34.3	32.5	25.7
(Ours) RotNet	**18.8**	**31.7**	**38.7**	**38.2**	**36.5**

Rotation

	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**

Temporal coherence of color

Task: given a color video …
Colorize all frames of a gray scale version using a reference frame

reference frame

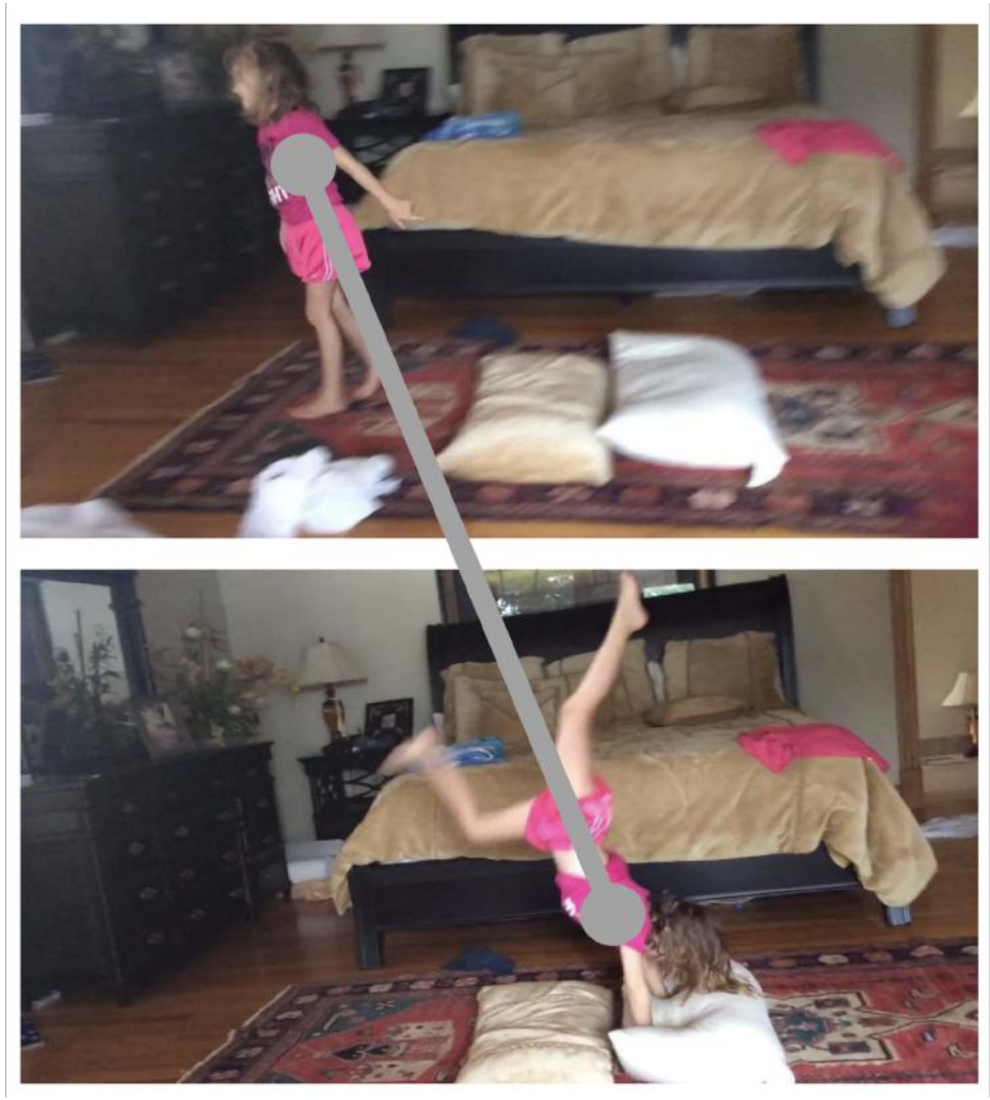
gray-scale video

Slide credit: Andrew Zisserman

Temporal coherence of color

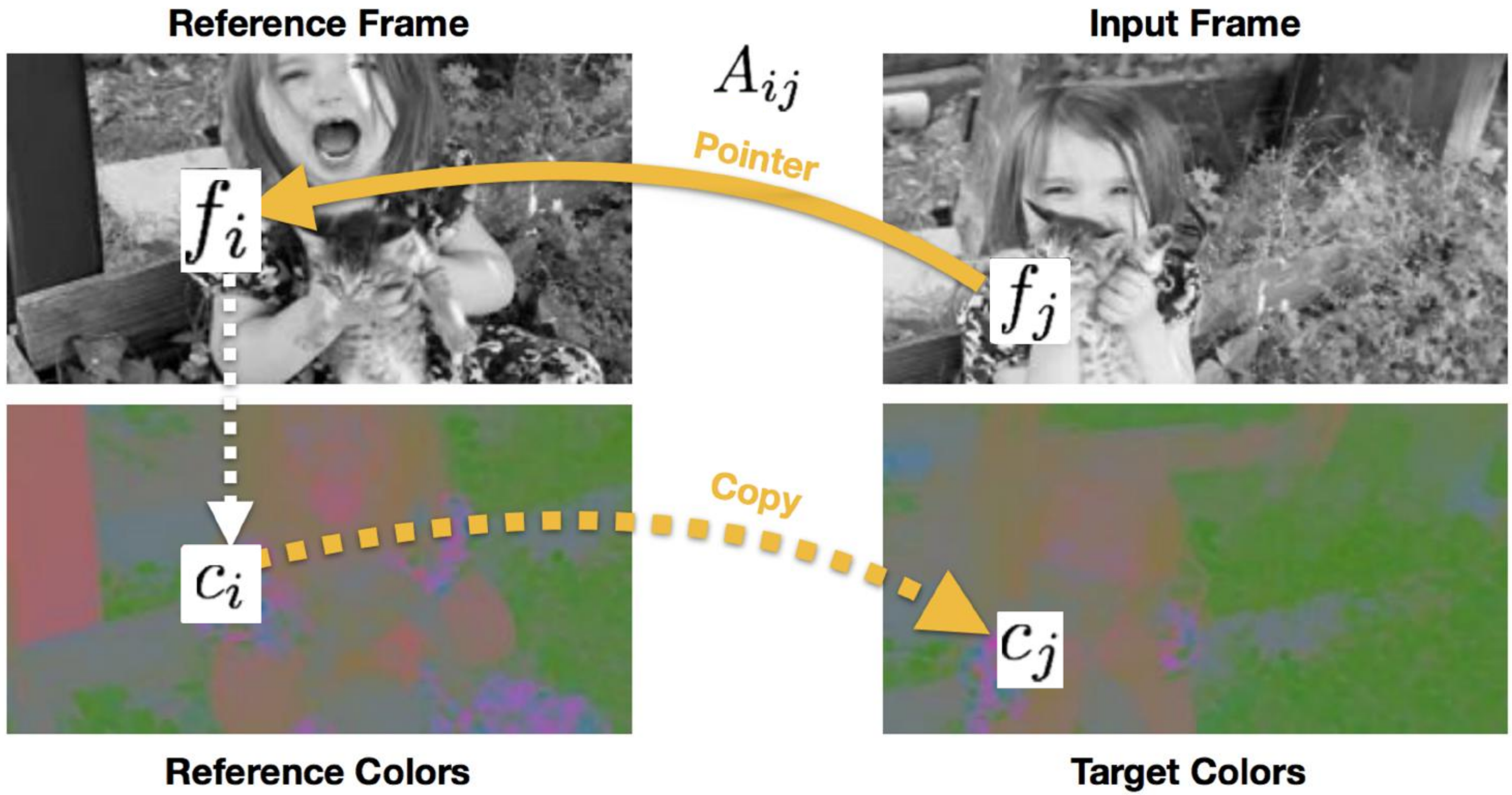
Slide credit: Andrew Zisserman

Temporal coherence of color

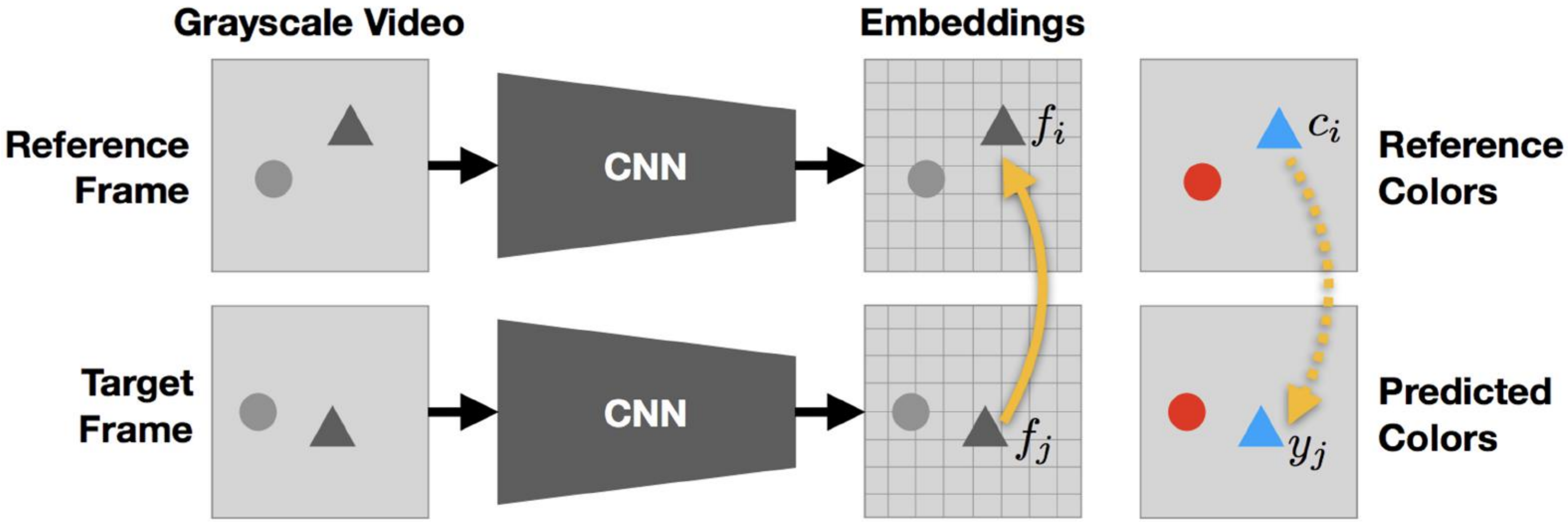


Slide credit: Andrew Zisserman

Tracking emerges from colorization

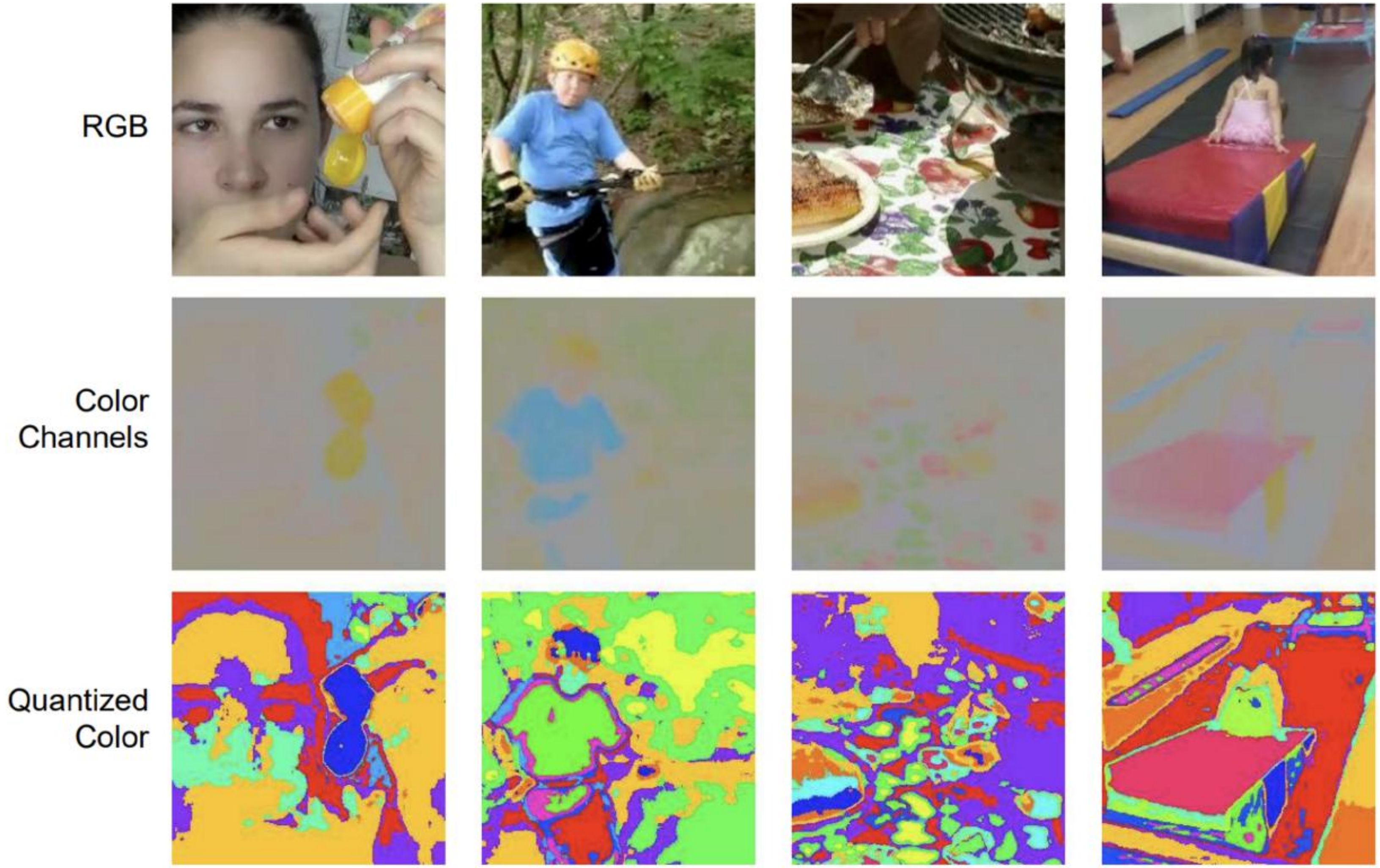


Tracking emerges from colorization



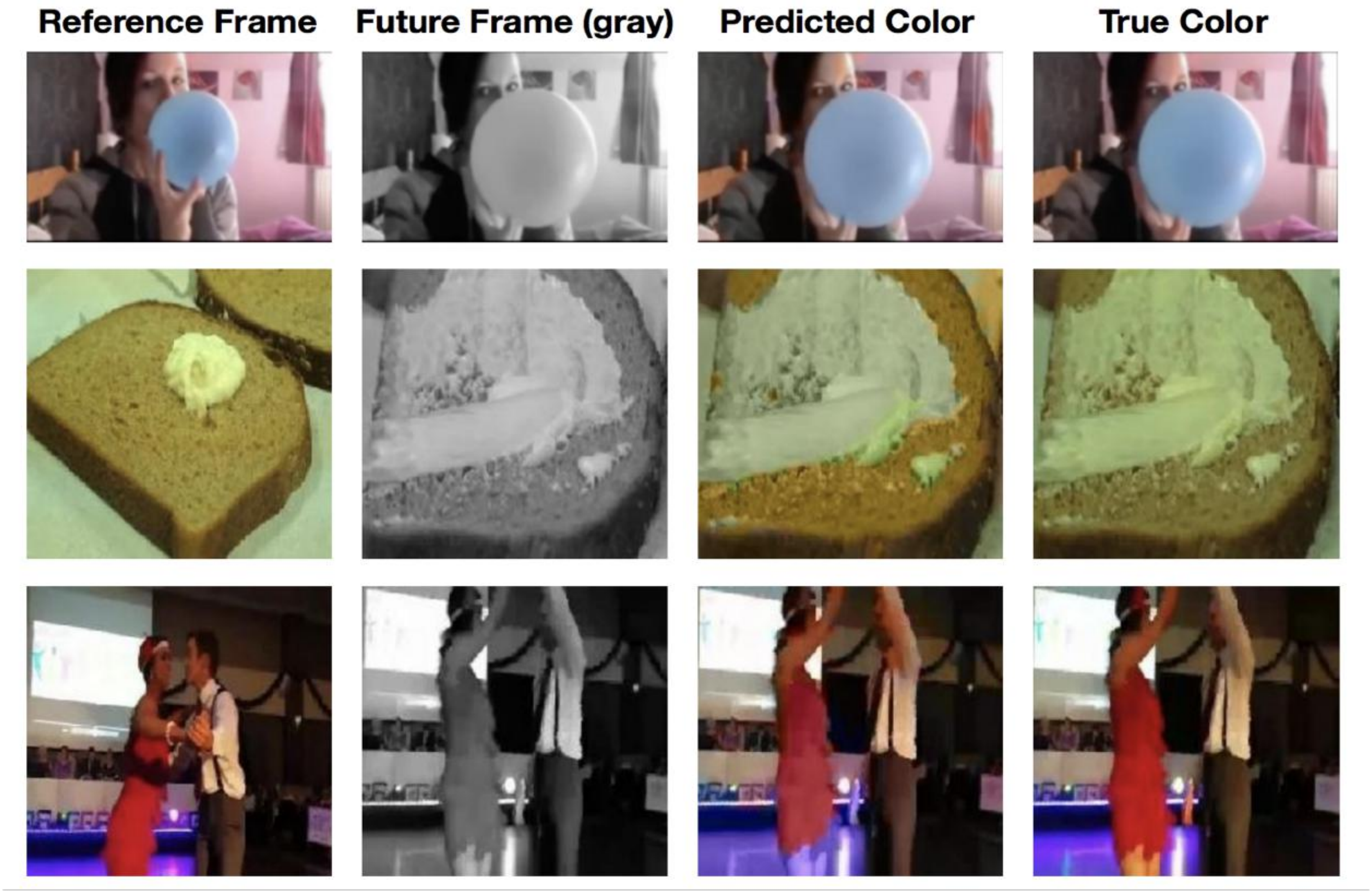
$$A_{ij} = \frac{\exp(f_i^T f_j)}{\sum_k \exp(f_k^T f_j)} \qquad \hat{c}_j = \sum_i A_{ij} c_i \qquad \min_f \mathcal{L}\left(c_j, \sum_i A_{ij} c_i\right)$$

Tracking emerges from colorization

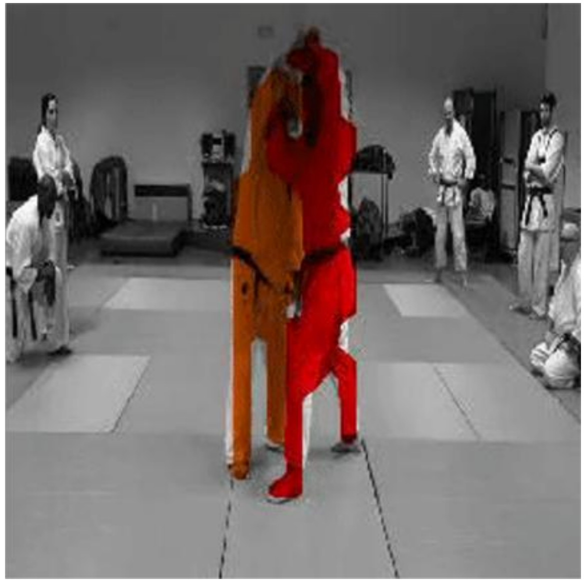


Slide credit: Andrew Zisserman

Tracking emerges from colorization



Tracking emerges from colorization

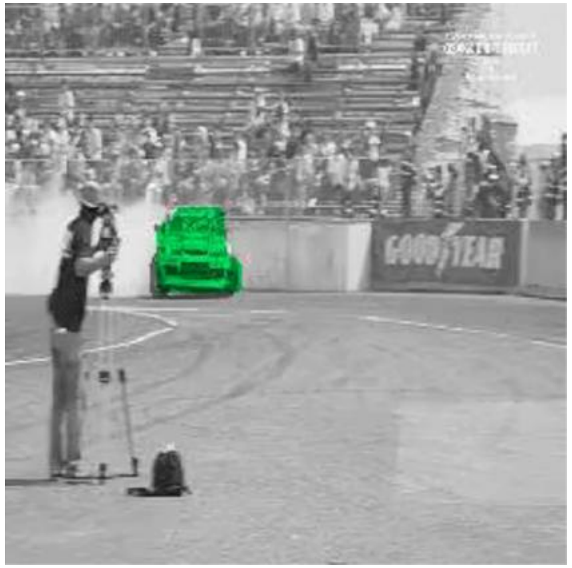


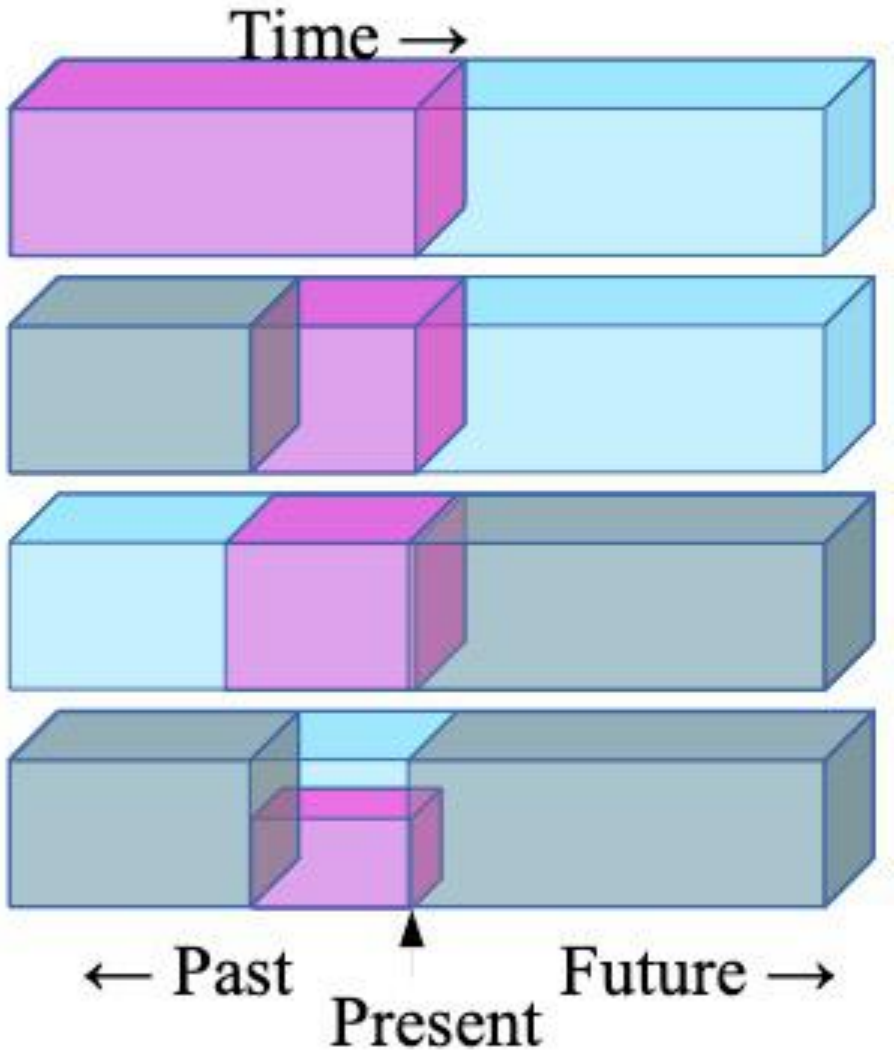
Image credit: Google AI Blog post

Lecture overview

- Motivation
- Reconstruct from a corrupted (or partial) version
- Proxy tasks in computer vision
- Contrastive Learning
  - Word2vec
  - Contrastive Predictive Coding (CPC)
  - Instance Discrimination
  - Recent State-of-the-art progress

Predicting neighbouring context

- Predict any part of the input from any other part.
- Predict the future from the past.
- Predict the future from the recent past.
- Predict the past from the present.
- Predict the top from the bottom.
- Predict the occluded from the visible
- **Pretend there is a part of the input you don't know and predict that.**



Word Embeddings

$$w^{aardvark} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{a} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{at} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \cdots \; w^{zebra} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

Slide credit: Stanford 224n

Word Embeddings

- The vast majority of rule-based or statistical NLP and IR work regarded words as atomic symbols: hotel, conference, walk

- In vector space terms, this is a vector with one 1 and a lot of zeroes

"hotel"

0 0 0 0 0 0 0 0 0 0 1 0 0 0 0

- We now call this a **one-hot** representation.

Slide credit: Stanford 224n

Word embeddings

- The size of word vectors are equal to the number of words in the dictionary
  - Vector size is proportional to the size of the dictionary
    20K (speech) – 50K (Pen Treebank) – 500K (A large dictionary) – 13M (Google 1T)
- One-hot vectors vectors are **orthogonal**
- There is no natural notion of similarity in a set of one-hot vectors

$$\begin{array}{r l} \text{“motel”} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^{T} \\ \text{“hotel”} & \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} = 0 \end{array}$$

Slide credit: Stanford 224n

Word embeddings

- You can get a lot of value by representing a word by means of its neighbors

- "You shall know a word by the company it keeps"

  (J. R. Firth 1957:11)

- One of the most successful ideas of modern NLP

government debt problems turning into	banking	crises as has happened in
saying that Europe needs unified	banking	regulation to replace the hodgepodge

These words will represent **"banking"**

Slide credit: Stanford 224n

Distributional hypothesis

- The meaning of a word is (can be approximated by, derived from) the set of contexts in which it occurs in texts

He filled the **wampimuk**, passed it around and we all drunk some

We found a little, hairy **wampimuk** sleeping behind the tree

Testing the distributional hypothesis: The influence of context on judgements of semantic similarity [McDonald & Ramscar'01]

Slide credit: Marco Baroni

Distributional semantics

```
he curtains open and the moon shining in on the barely
ars and the cold , close moon " . And neither of the w
rough the night with the moon shining so brightly , it
made in the light of the moon . It all boils down , wr
 surely under a crescent moon , thrilled by ice-white
sun , the seasons of the moon ? Home , alone , Jay pla
m is dazzling snow , the moon has risen full and cold
un and the temple of the moon , driving out of the hug
 in the dark and now the moon rises , full and amber a
bird on the shape of the moon over the trees in front
 But I could n't see the moon or the stars , only the
rning , with a sliver of moon hanging among the stars
 they love the sun , the moon and the stars . None of
the light of an enormous moon . The plash of flowing w
man 's first step on the moon ; various exhibits , aer
 the inevitable piece of moon rock . Housing The Airsh
oud obscured part of the moon . The Allied guns behind
```

A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge [Landauer and Dumais'97]
From frequency to meaning: Vector space models of semantics [Turney ve Pantel'10]
…

Slide credit: Marco Baroni

Word Embeddings

1. I enjoy flying.
2. I like NLP.
3. I like deep learning.

The resulting counts matrix will then be:

$$X = \begin{array}{c} \\ I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{array} \begin{array}{c} \begin{array}{cccccccc} I & like & enjoy & deep & learning & NLP & flying & . \end{array} \\ \begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

Slide credit: Stanford 224n

Word Embeddings

Applying SVD to X:

$$
\begin{array}{cc} & |V| \\ |V| & \begin{bmatrix} & & \\ & X & \\ & & \end{bmatrix} \end{array}
=
\begin{array}{cc} & |V| \\ |V| & \begin{bmatrix} | & | & \\ u_1 & u_2 & \cdots \\ | & | & \end{bmatrix} \end{array}
\begin{array}{cc} & |V| \\ |V| & \begin{bmatrix} \sigma_1 & 0 & \cdots \\ 0 & \sigma_2 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \end{array}
\begin{array}{cc} & |V| \\ |V| & \begin{bmatrix} - & v_1 & - \\ - & v_2 & - \\ & \vdots & \end{bmatrix} \end{array}
$$

Slide credit: Stanford 224n

Word Embeddings

SVD approach suffers from:

- Sparsity
- SVD computation costs
- Infrequent words
- Noise from frequent words
- There are hacks to fix some of these (ex TF-IDF) but still not very reliable

Slide credit: Stanford 224n

n-gram Language Models

Unigram

$$P(w_1, w_2, \cdots, w_n) = \prod_{i=1}^{n} P(w_i)$$

Bigram

$$P(w_1, w_2, \cdots, w_n) = \prod_{i=2}^{n} P(w_i | w_{i-1})$$

Slide credit: Stanford 224n

word2vec

Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

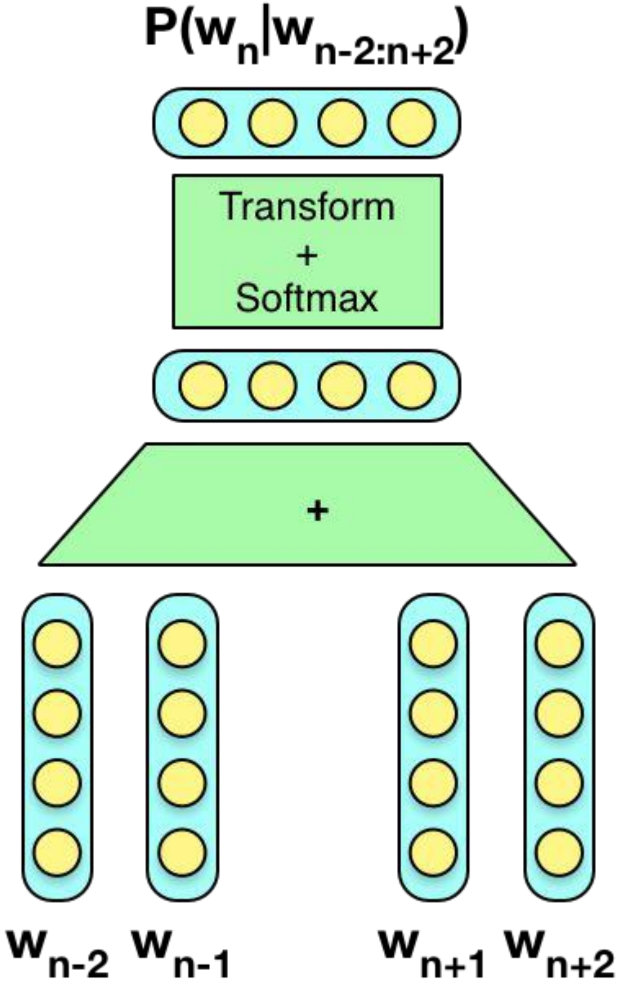
Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

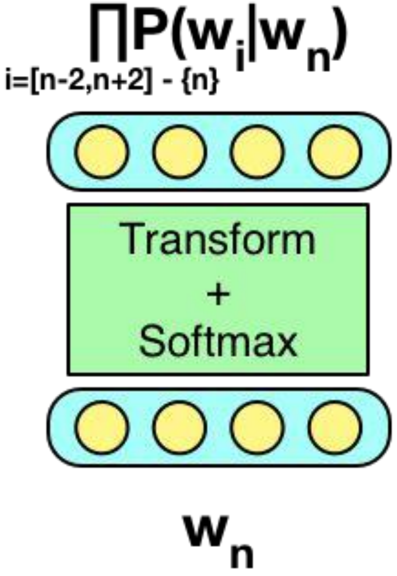
Abstract

We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing techniques based on different types of neural networks. We observe large improvements in accuracy at much lower computational cost, i.e. it takes less than a day to learn high quality word vectors from a 1.6 billion words data set. Furthermore, we show that these vectors provide state-of-the-art performance on our test set for measuring syntactic and semantic word similarities.

word2vec



CBoW model



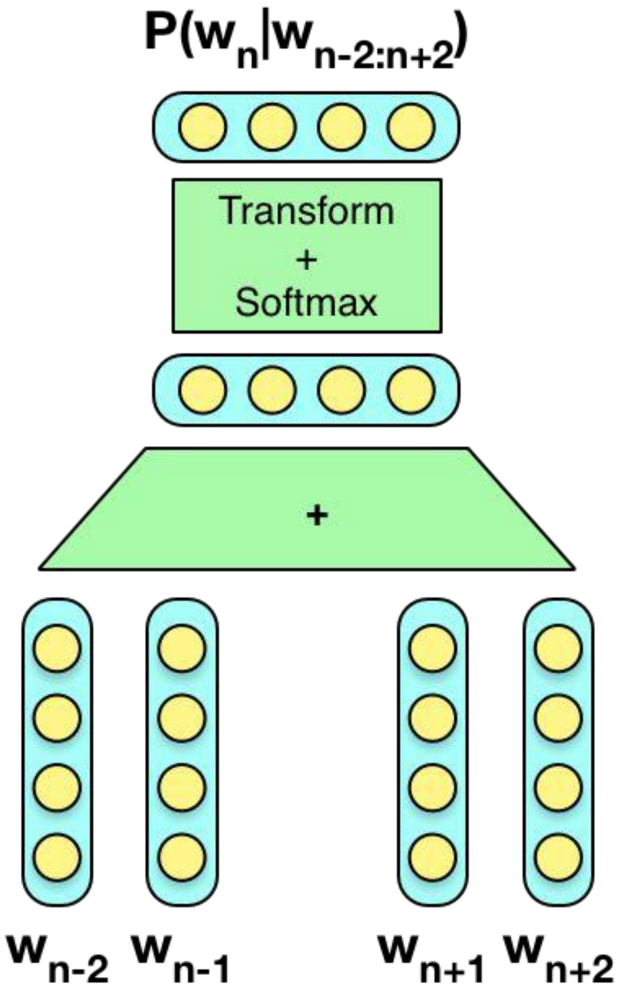
Skip-gram model

Distributed representations of words and phrases and their compositionality [Mikolov vd.'13]

Image credit: Ed Grefenstette

word2vec - CBOW

Continuous Bag Of
Words (CBOW)

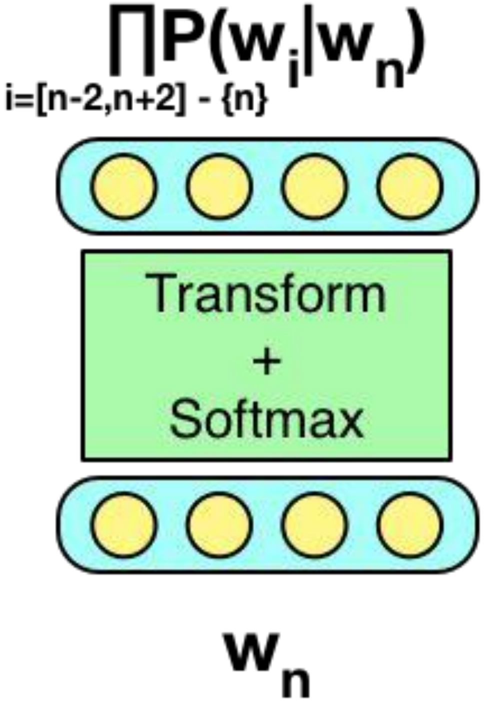


$$
\begin{aligned}
\text{minimize } J &= -\log P(w_c | w_{c-m}, \ldots, w_{c-1}, w_{c+1}, \ldots, w_{c+m}) \\
&= -\log P(u_c | \hat{v}) \\
&= -\log \frac{\exp(u_c^T \hat{v})}{\sum_{j=1}^{|V|} \exp(u_j^T \hat{v})} \\
&= -u_c^T \hat{v} + \log \sum_{j=1}^{|V|} \exp(u_j^T \hat{v})
\end{aligned}
$$

Slide credit: Stanford 224n

word2vec - Skip Gram

Skip Gram



$$
\begin{aligned}
\text{minimize } J &= -\log P(w_{c-m}, \ldots, w_{c-1}, w_{c+1}, \ldots, w_{c+m} | w_c) \\
&= -\log \prod_{j=0, j\neq m}^{2m} P(w_{c-m+j}|w_c) \\
&= -\log \prod_{j=0, j\neq m}^{2m} P(u_{c-m+j}|v_c) \\
&= -\log \prod_{j=0, j\neq m}^{2m} \frac{\exp(u_{c-m+j}^T v_c)}{\sum_{k=1}^{|V|} \exp(u_k^T v_c)} \\
&= -\sum_{j=0, j\neq m}^{2m} u_{c-m+j}^T v_c + 2m \log \sum_{k=1}^{|V|} \exp(u_k^T v_c)
\end{aligned}
$$

$$
\begin{aligned}
J &= -\sum_{j=0, j\neq m}^{2m} \log P(u_{c-m+j}|v_c) \\
&= \sum_{j=0, j\neq m}^{2m} H(\hat{y}, y_{c-m+j})
\end{aligned}
$$

Slide credit: Stanford 224n

word2vec - Skip Gram

Skip-gram model

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c,j\neq 0}\log p(w_{t+j}|w_t)$$

$$p(w_O|w_I)=\frac{\exp\left({v'_{w_O}}^\top v_{w_I}\right)}{\sum_{w=1}^{W}\exp\left({v'_w}^\top v_{w_I}\right)}$$

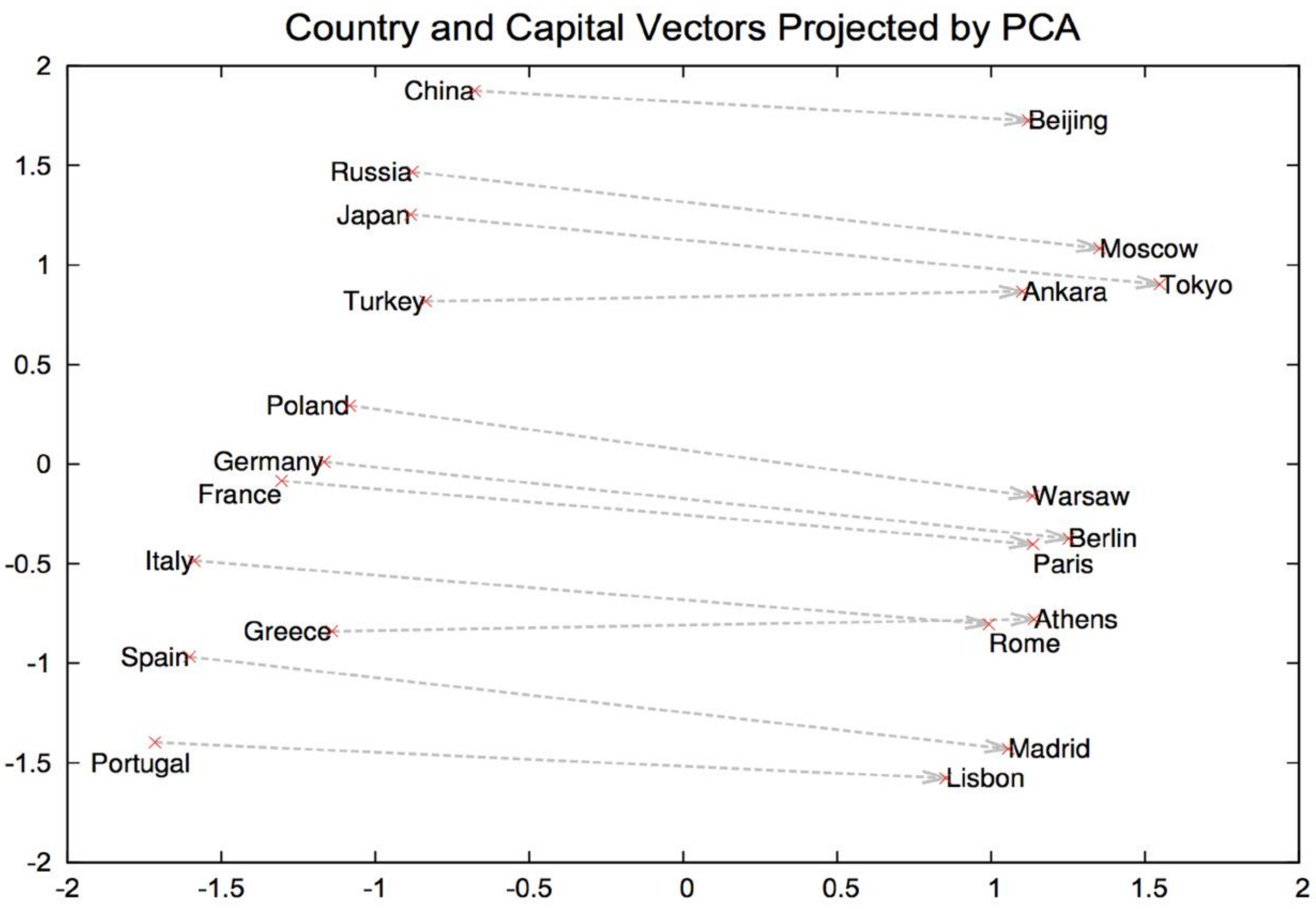
Don't have to have the denominator over all words in the vocabulary

- Can use negative sampling

$$\log\sigma({v'_{w_O}}^\top v_{w_I})+\sum_{i=1}^{k}\mathbb{E}_{w_i\sim P_n(w)}\left[\log\sigma(-{v'_{w_i}}^\top v_{w_I})\right]$$

$$P(w_i)=1-\sqrt{\frac{t}{f(w_i)}}$$

word2vec


Country and Capital Vectors Projected by PCA
China
Beijing
Russia
Japan
Moscow
Turkey
Ankara
Tokyo
Poland
Germany
France
Warsaw
Berlin
Paris
Italy
Greece
Spain
Athens
Rome
Portugal
Madrid
Lisbon
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

word2vec

Newspapers			
New York San Jose	New York Times San Jose Mercury News	Baltimore Cincinnati	Baltimore Sun Cincinnati Enquirer
NHL Teams			
Boston Phoenix	Boston Bruins Phoenix Coyotes	Montreal Nashville	Montreal Canadiens Nashville Predators
NBA Teams			
Detroit Oakland	Detroit Pistons Golden State Warriors	Toronto Memphis	Toronto Raptors Memphis Grizzlies
Airlines			
Austria Belgium	Austrian Airlines Brussels Airlines	Spain Greece	Spainair Aegean Airlines
Company executives			
Steve Ballmer Samuel J. Palmisano	Microsoft IBM	Larry Page Werner Vogels	Google Amazon

word2vec

	NEG-15 with 10^{-5} subsampling	HS with 10^{-5} subsampling
Vasco de Gama	Lingsugur	Italian explorer
Lake Baikal	Great Rift Valley	Aral Sea
Alan Bean	Rebbeca Naomi	moonwalker
Ionian Sea	Ruegen	Ionian Islands
chess master	chess grandmaster	Garry Kasparov

Table 4: Examples of the closest entities to the given short phrases, using two different models.

Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

Table 5: Vector compositionality using element-wise addition. Four closest tokens to the sum of two vectors are shown, using the best Skip-gram model.

word2vec

Model (training time)	Redmond	Havel	ninjutsu	graffiti	capitulate
Collobert (50d) (2 months)	conyers lubbock keene	plauen dzerzhinsky osterreich	reiki kohona karate	cheesecake gossip dioramas	abdicate accede rearm
Turian (200d) (few weeks)	McCarthy Alston Cousins	Jewell Arzu Ovitz	- - -	gunfire emotion impunity	- - -
Mnih (100d) (7 days)	Podhurst Harlang Agarwal	Pontiff Pinochet Rodionov	- - -	anaesthetics monkeys Jews	Mavericks planning hesitated
Skip-Phrase (1000d, 1 day)	Redmond Wash. Redmond Washington Microsoft	Vaclav Havel president Vaclav Havel Velvet Revolution	ninja martial arts swordsmanship	spray paint grafitti taggers	capitulation capitulated capitulating

Table 6: Examples of the closest tokens given various well known models and the Skip-gram model trained on phrases using over 30 billion training words. An empty cell means that the word was not in the vocabulary.

Deep InfoMax

LEARNING DEEP REPRESENTATIONS BY MUTUAL INFORMATION ESTIMATION AND MAXIMIZATION

R Devon Hjelm
MSR Montreal, MILA, UdeM, IVADO
devon.hjelm@microsoft.com

Alex Fedorov
MRN, UNM

Samuel Lavoie-Marchildon
MILA, UdeM

Karan Grewal
U Toronto

Phil Bachman
MSR Montreal

Adam Trischler
MSR Montreal

Yoshua Bengio
MILA, UdeM, IVADO, CIFAR

ABSTRACT

This work investigates unsupervised learning of representations by maximizing mutual information between an input and the output of a deep neural network encoder. Importantly, we show that structure matters: incorporating knowledge about locality in the input into the objective can significantly improve a representation's suitability for downstream tasks. We further control characteristics of the representation by matching to a prior distribution adversarially. Our method, which we call Deep InfoMax (DIM), outperforms a number of popular unsupervised learning methods and compares favorably with fully-supervised learning on several classification tasks in with some standard architectures. DIM opens new avenues for unsupervised learning of representations and is an important step towards flexible formulations of representation learning objectives for specific end-goals.

Deep InfoMax

- Network encodes the input
- The discriminator estimates mutual information (batch-wise)
- Estimate is used to maximize the MI between encoder input and output

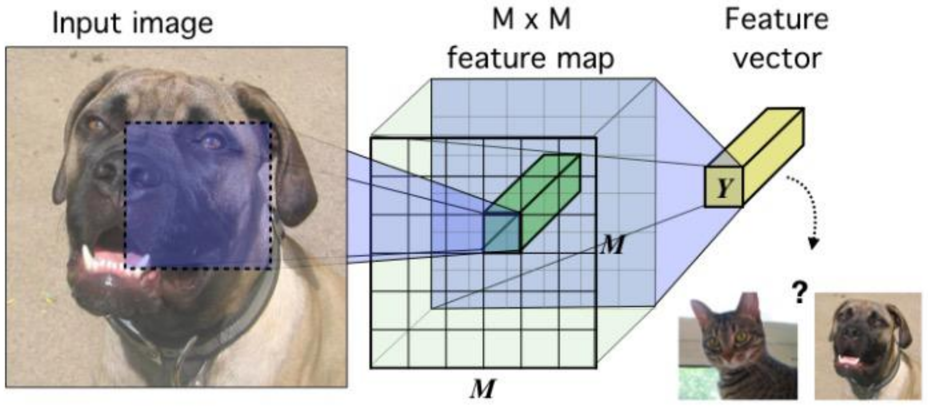


Figure 1: **The base encoder model in the context of image data.** An image (in this case) is encoded using a convnet until reaching a feature map of $M \times M$ feature vectors corresponding to $M \times M$ input patches. These vectors are summarized into a single feature vector, Y. Our goal is to train this network such that useful information about the input is easily extracted from the high-level features.

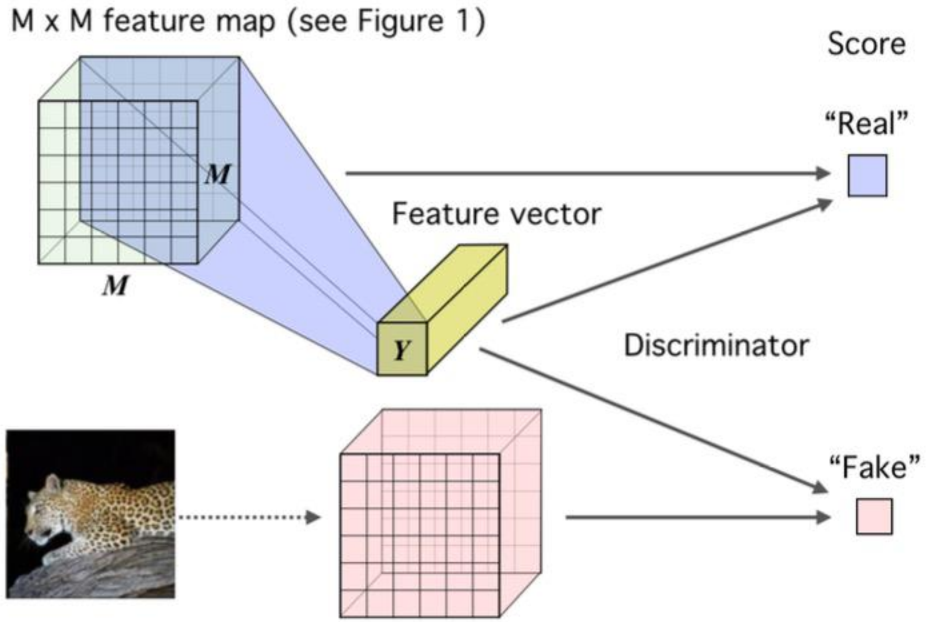


Figure 2: **Deep InfoMax (DIM) with a global MI$(X;Y)$ objective.** Here, we pass both the high-level feature vector, Y, and the lower-level $M \times M$ feature map (see Figure 1) through a discriminator to get the score. Fake samples are drawn by combining the same feature vector with a $M \times M$ feature map from another image.

Deep InfoMax

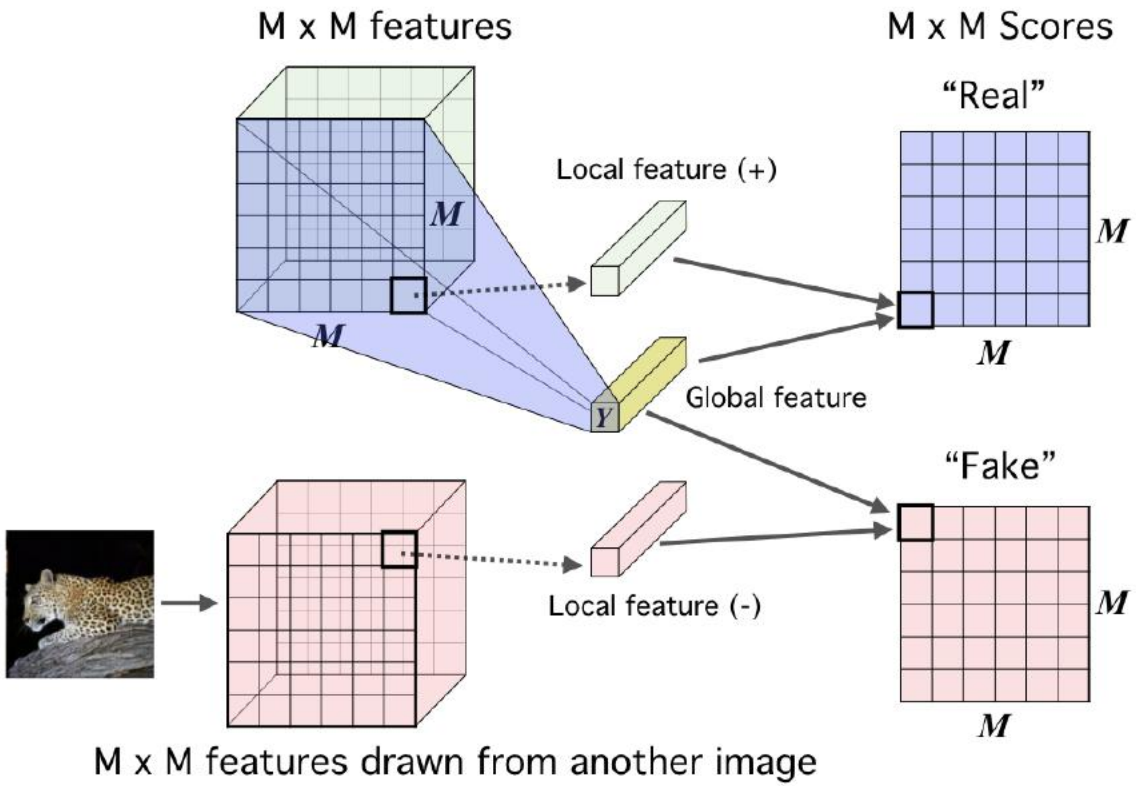


Figure 3: **Maximizing mutual information between local features and global features.** First we encode the image to a feature map that reflects some structural aspect of the data, e.g. spatial locality, and we further summarize this feature map into a global feature vector (see Figure 1). We then concatenate this feature vector with the lower-level feature map *at every location*. A score is produced for each local-global pair through an additional function (see the Appendix A.2 for details).

Deep InfoMax

Table 1: Classification accuracy (top 1) results on CIFAR10 and CIFAR100. DIM(L) (i.e., with the local-only objective) outperforms all other unsupervised methods presented by a wide margin. In addition, DIM(L) approaches or even surpasses a fully-supervised classifier with similar architecture. DIM with the global-only objective is competitive with some models across tasks, but falls short when compared to generative models and DIM(L) on CIFAR100. Fully-supervised classification results are provided for comparison.

Model	CIFAR10			CIFAR100		
	conv	fc (1024)	$Y(64)$	conv	fc (1024)	$Y(64)$
Fully supervised		75.39			42.27	
VAE	60.71	60.54	54.61	37.21	34.05	24.22
AE	62.19	55.78	54.47	31.50	23.89	27.44
β-VAE	62.4	57.89	55.43	32.28	26.89	28.96
AAE	59.44	57.19	52.81	36.22	33.38	23.25
BiGAN	62.57	62.74	52.54	37.59	33.34	21.49
NAT	56.19	51.29	31.16	29.18	24.57	9.72
DIM(G)	52.2	52.84	43.17	27.68	24.35	19.98
DIM(L) (DV)	**72.66**	**70.60**	**64.71**	**48.52**	**44.44**	**39.27**
DIM(L) (JSD)	**73.25**	**73.62**	**66.96**	**48.13**	**45.92**	**39.60**
DIM(L) (infoNCE)	**75.21**	**75.57**	**69.13**	**49.74**	**47.72**	**41.61**

Deep InfoMax

Table 2: Classification accuracy (top 1) results on Tiny ImageNet and STL-10. For Tiny ImageNet, DIM with the local objective outperforms all other models presented by a large margin, and approaches accuracy of a fully-supervised classifier similar to the Alexnet architecture used here.

	Tiny ImageNet			STL-10 (random crop pretraining)			
	conv	fc (4096)	$Y(64)$	conv	fc (4096)	$Y(64)$	SS
Fully supervised	36.60			68.7			
VAE	18.63	16.88	11.93	58.27	56.72	46.47	68.65
AE	19.07	16.39	11.82	58.19	55.57	46.82	70.29
β-VAE	19.29	16.77	12.43	57.15	55.14	46.87	70.53
AAE	18.04	17.27	11.49	59.54	54.47	43.89	64.15
BiGAN	24.38	20.21	13.06	71.53	67.18	58.48	74.77
NAT	13.70	11.62	1.20	64.32	61.43	48.84	70.75
DIM(G)	11.32	6.34	4.95	42.03	30.82	28.09	51.36
DIM(L) (DV)	30.35	29.51	28.18	69.15	63.81	61.92	71.22
DIM(L) (JSD)	**33.54**	**36.88**	**31.66**	**72.86**	**70.85**	**65.93**	**76.96**
DIM(L) (infoNCE)	**34.21**	**38.09**	**33.33**	**72.57**	**70.00**	**67.08**	**76.81**

Contrastive Predictive Coding

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

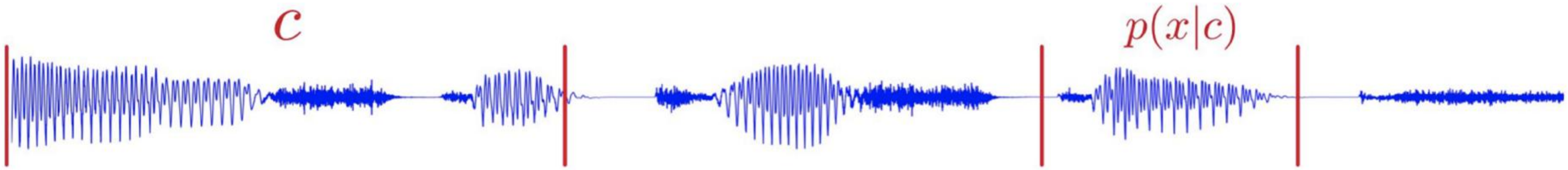


Figure credit: Alex Graves

Contrastive Predictive Coding

Figure credit: Alex Graves

Contrastive Predictive Coding

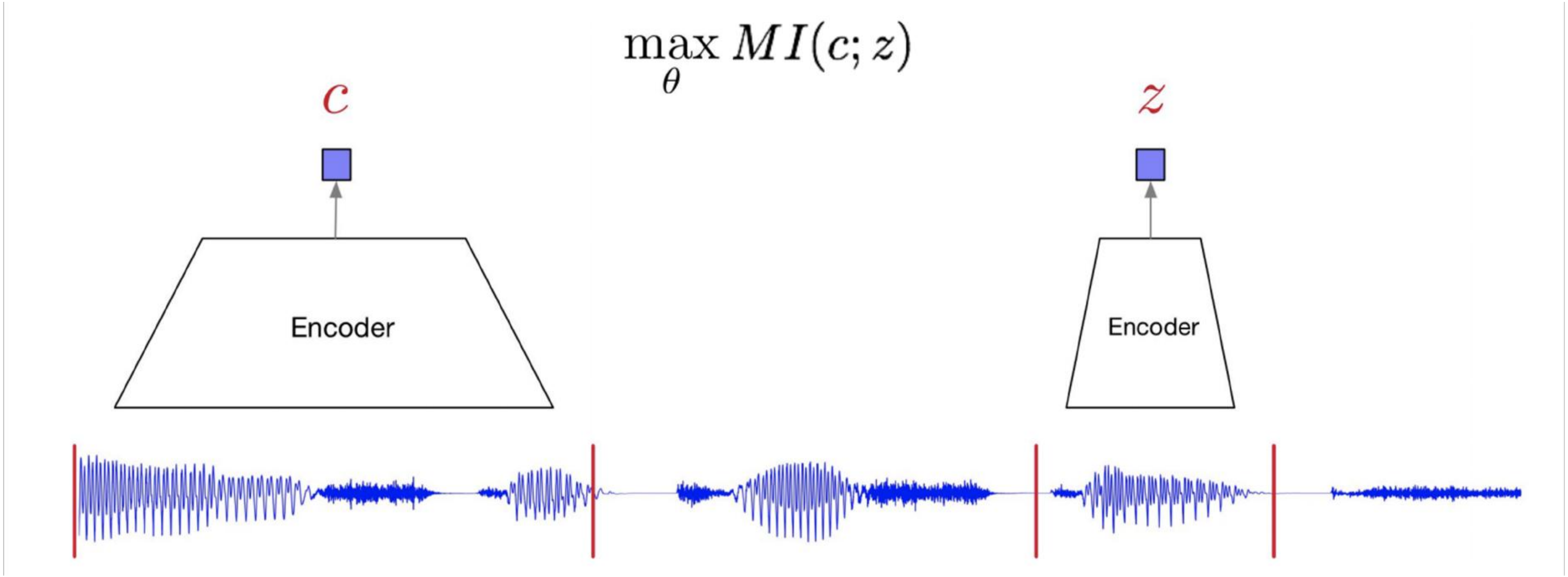


Figure credit: Alex Graves

Contrastive Predictive Coding

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

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

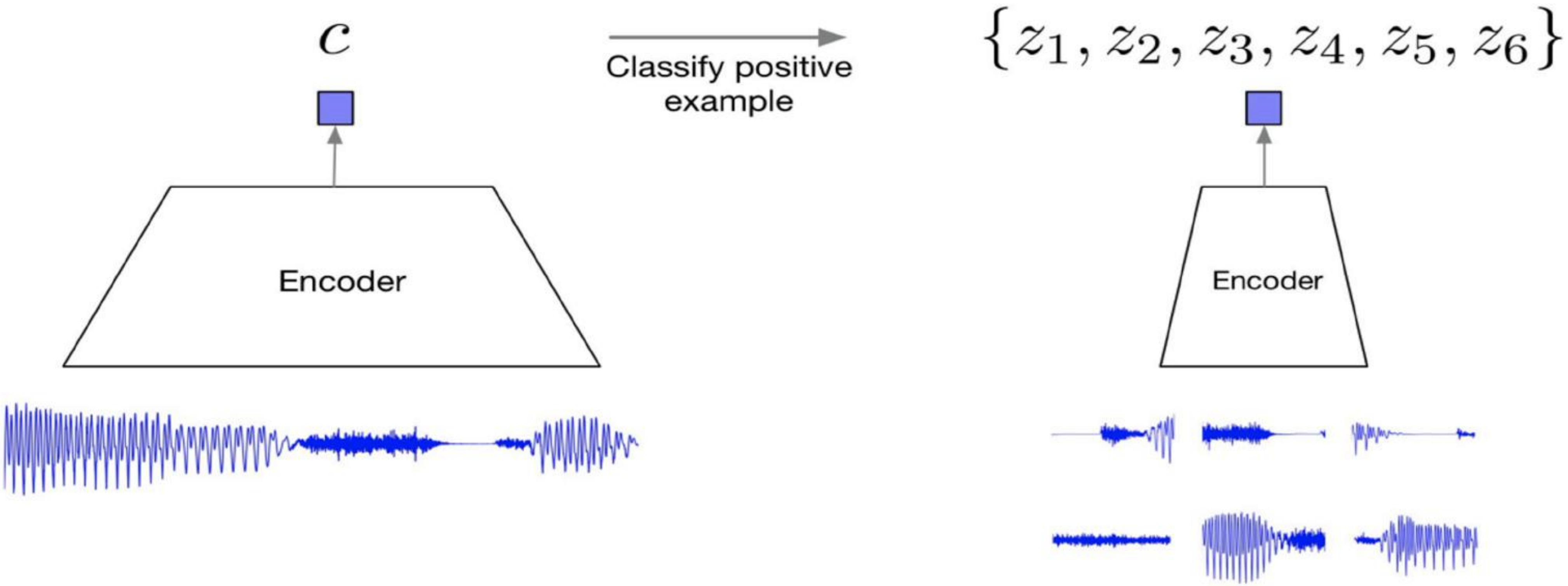


Figure credit: Alex Graves

Contrastive Predictive Coding

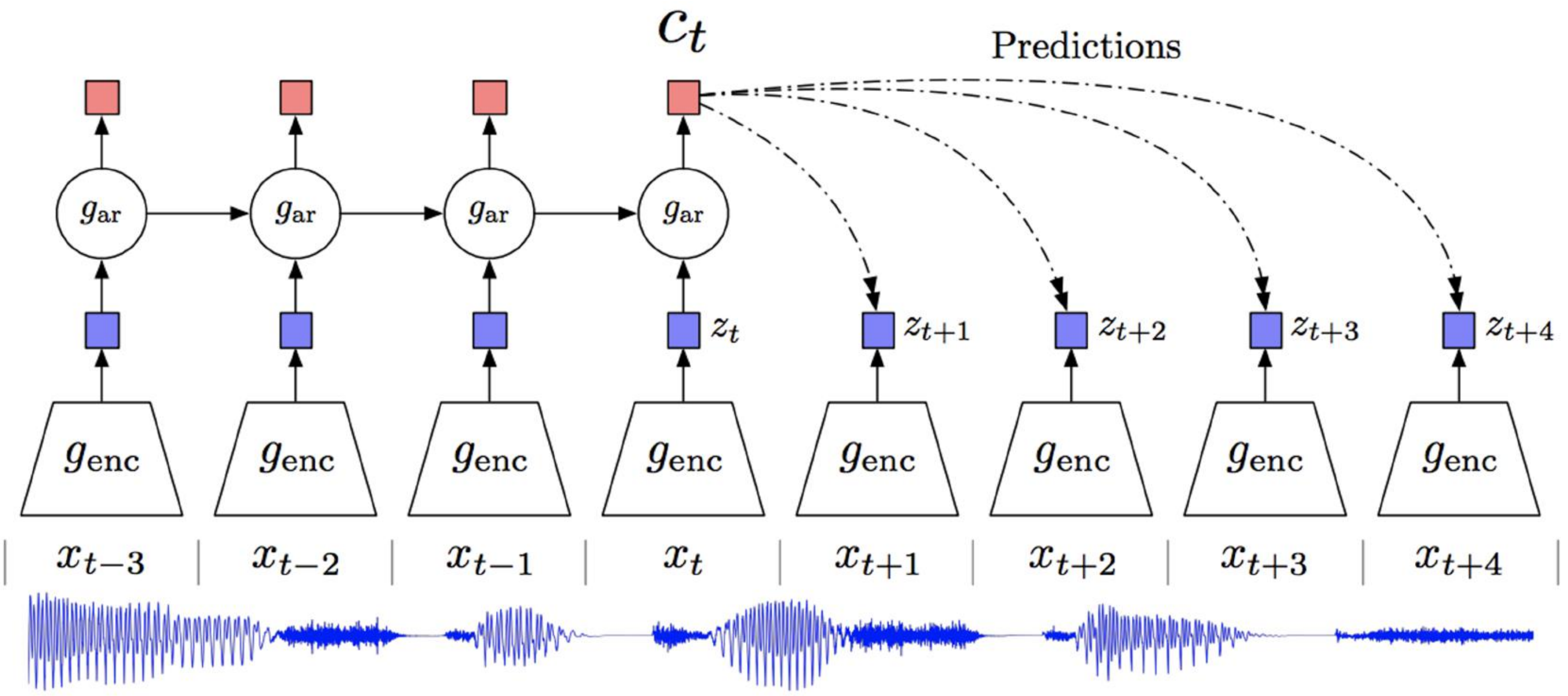
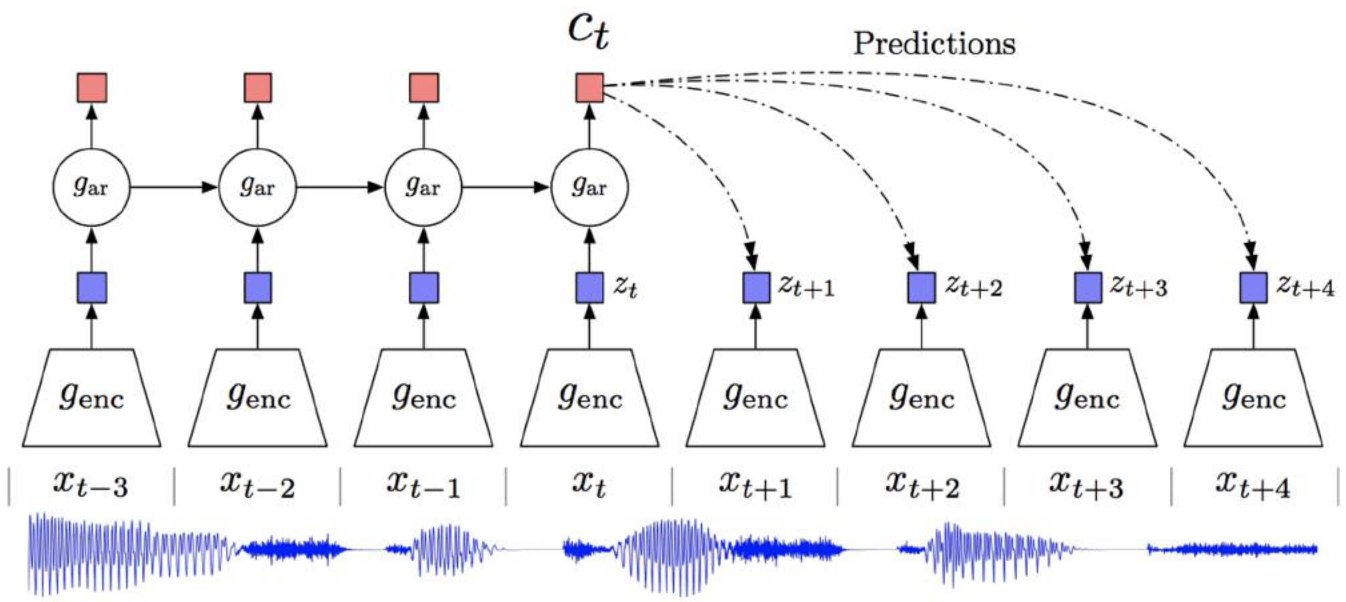


Figure credit: Alex Graves

Contrastive Predictive Coding



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

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

Figure credit: Alex Graves

Contrastive Predictive Coding

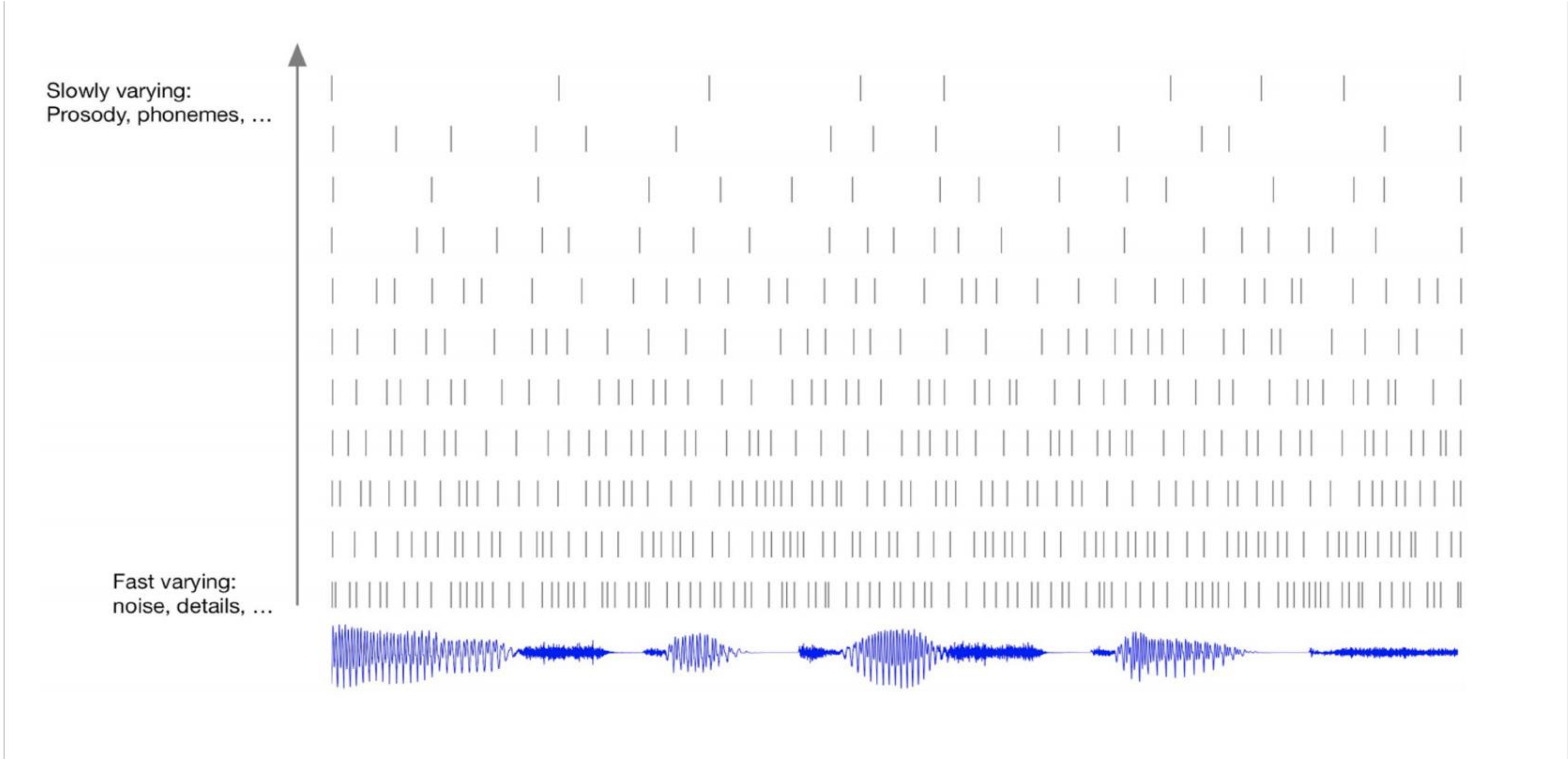


Figure credit: Alex Graves

Contrastive Predictive Coding

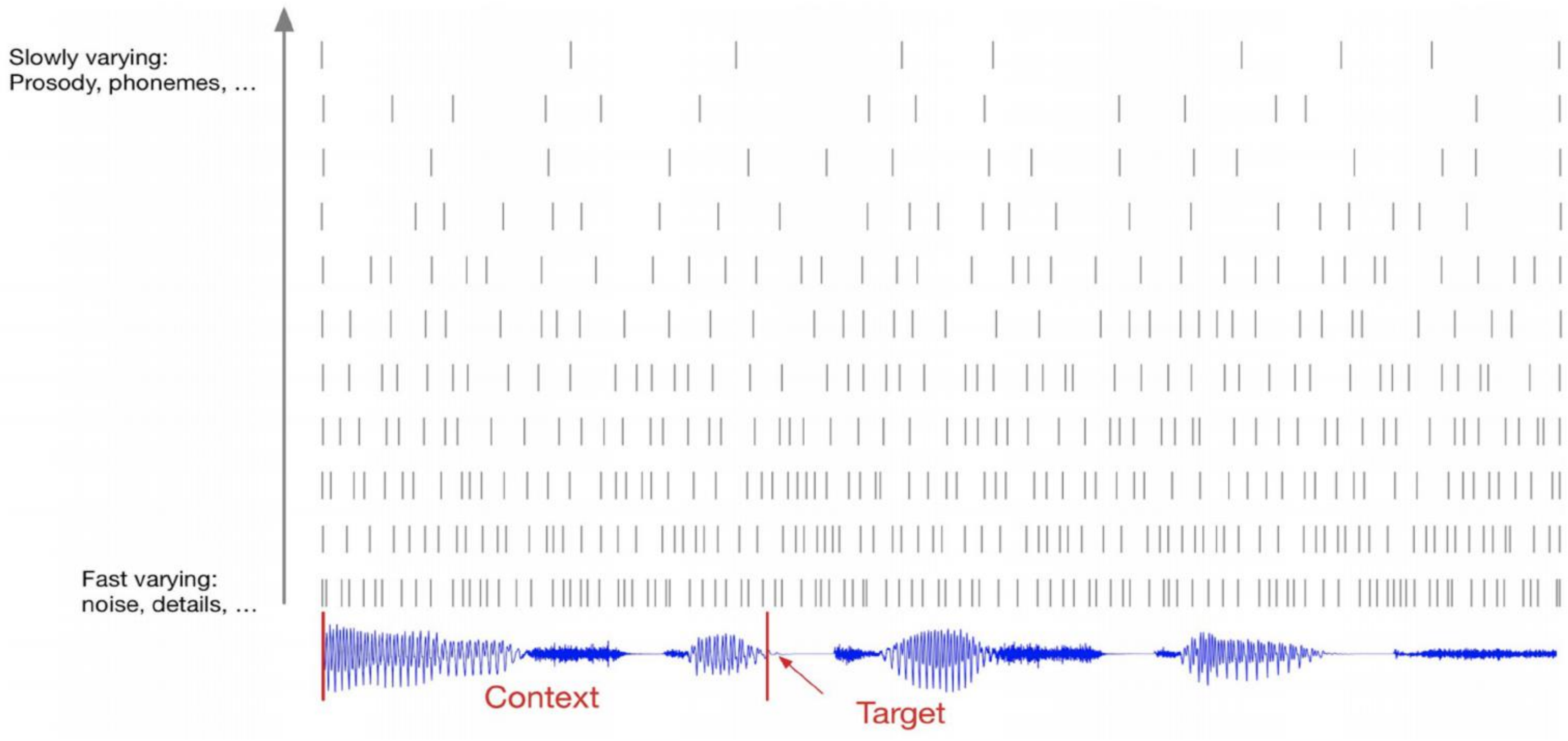


Figure credit: Alex Graves

Contrastive Predictive Coding

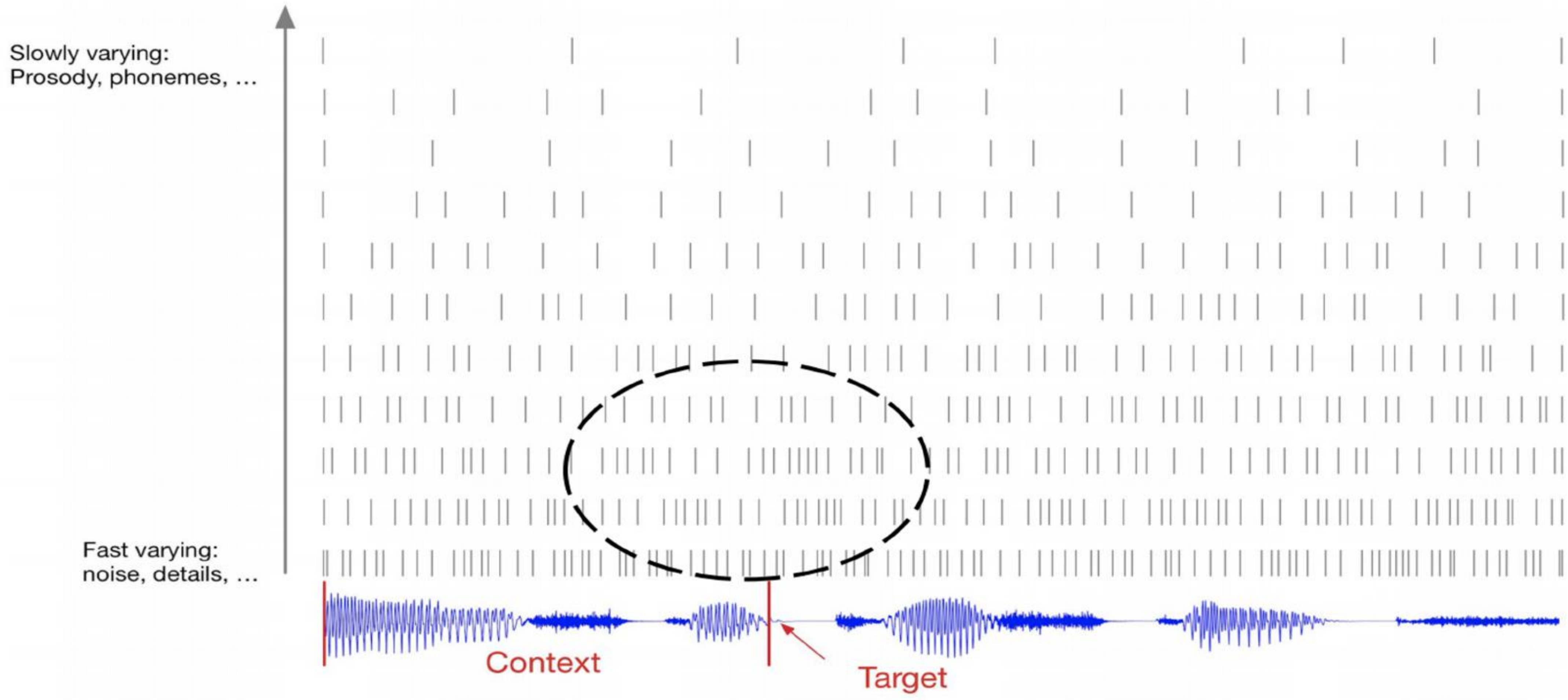


Figure credit: Alex Graves

Contrastive Predictive Coding

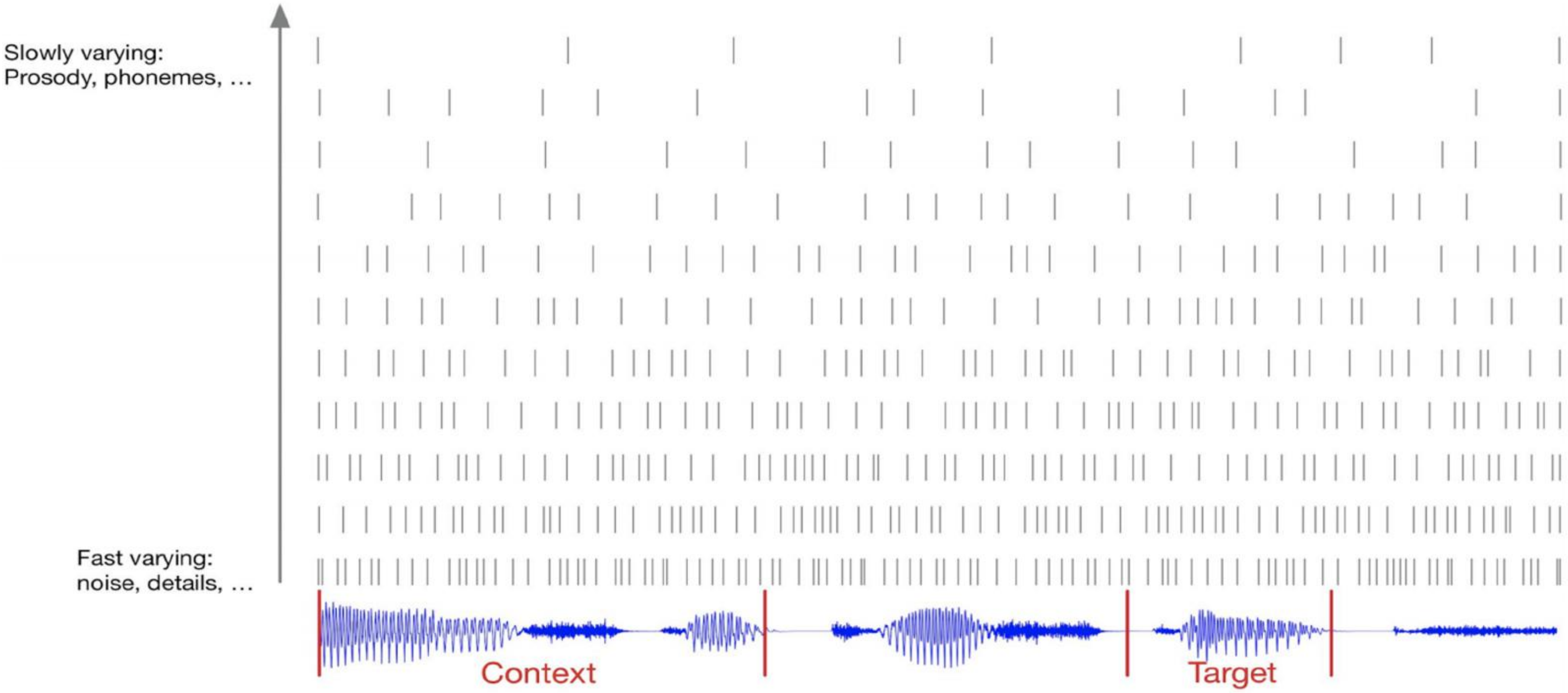


Figure credit: Alex Graves

Contrastive Predictive Coding

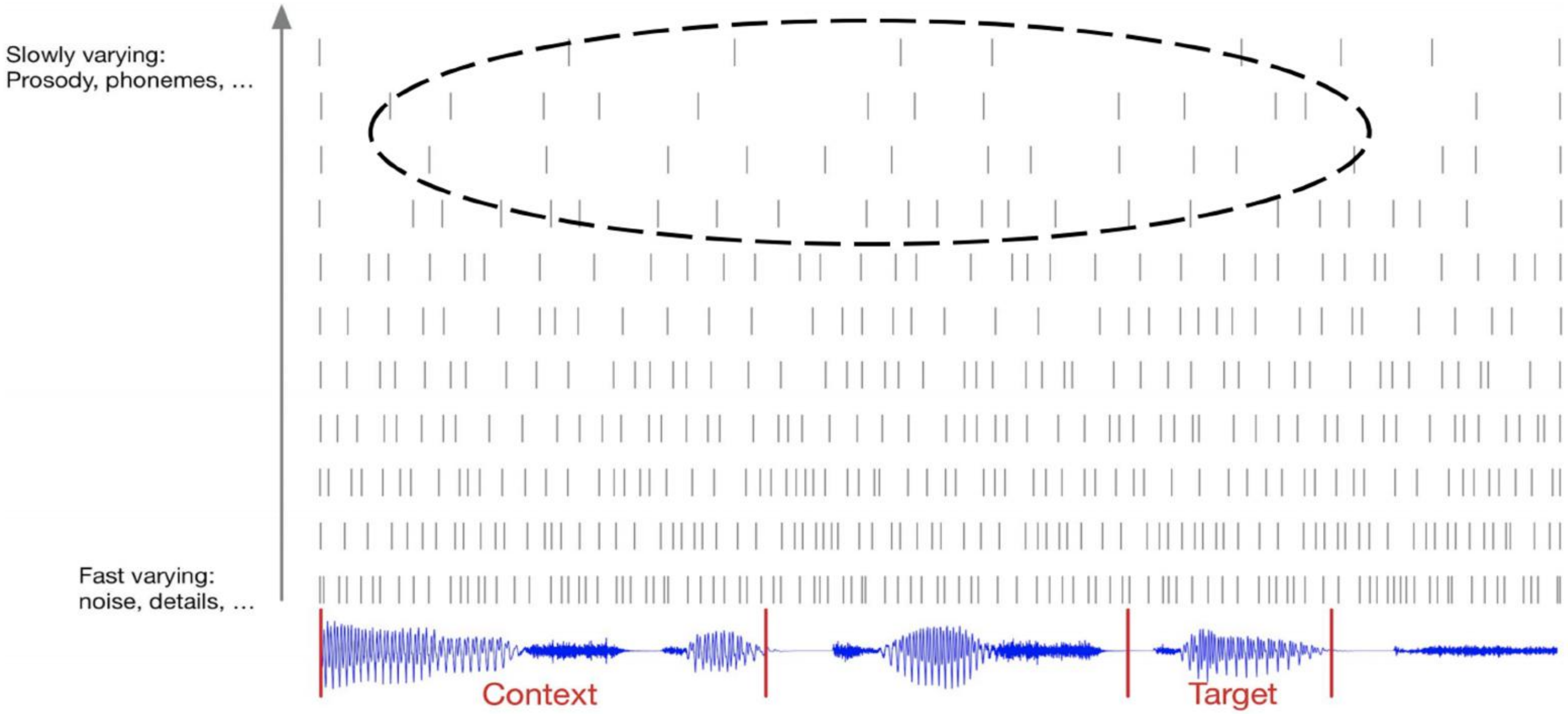


Figure credit: Alex Graves

Contrastive Predictive Coding

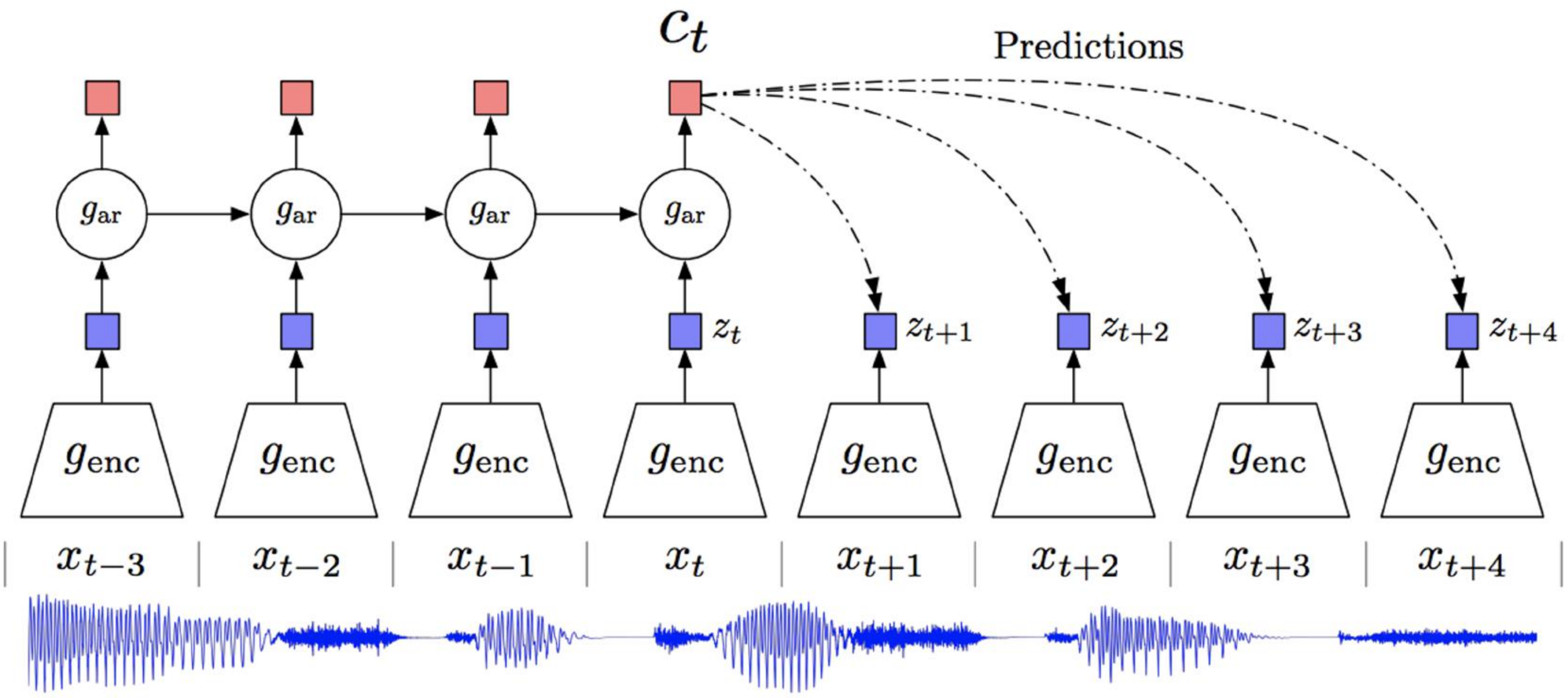


Figure credit: Alex Graves

CPC - Speech

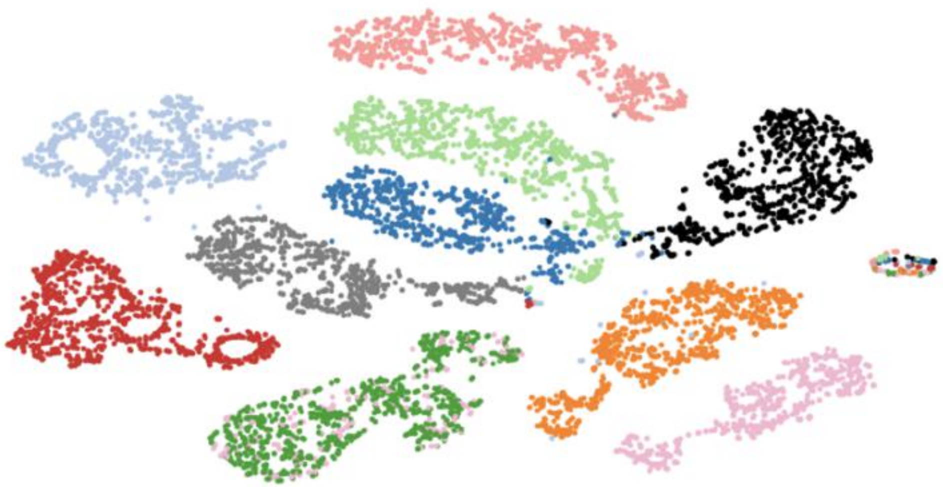


Figure 2: t-SNE visualization of audio (speech) representations for a subset of 10 speakers (out of 251). Every color represents a different speaker.

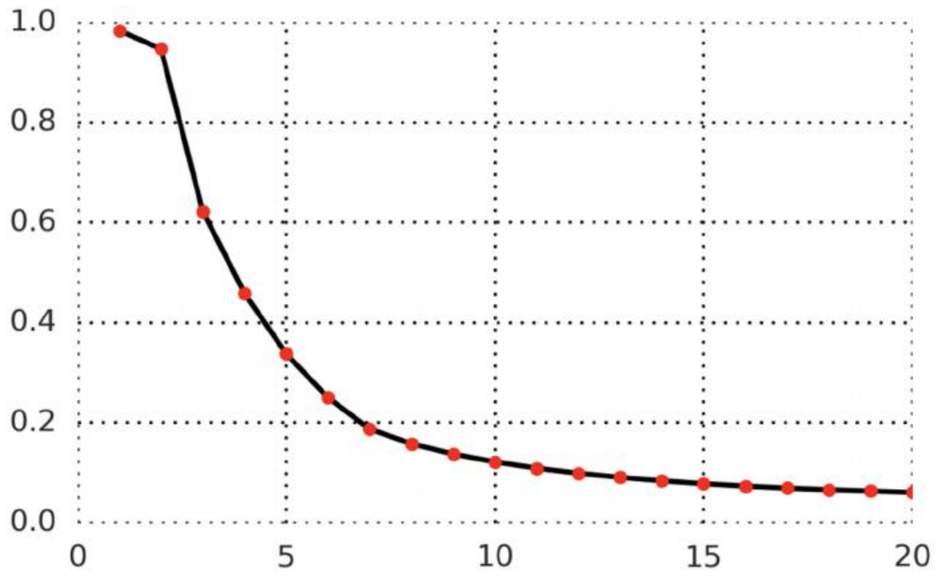


Figure 3: Average accuracy of predicting the positive sample in the contrastive loss for 1 to 20 latent steps in the future of a speech waveform. The model predicts up to 200ms in the future as every step consists of 10ms of audio.

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

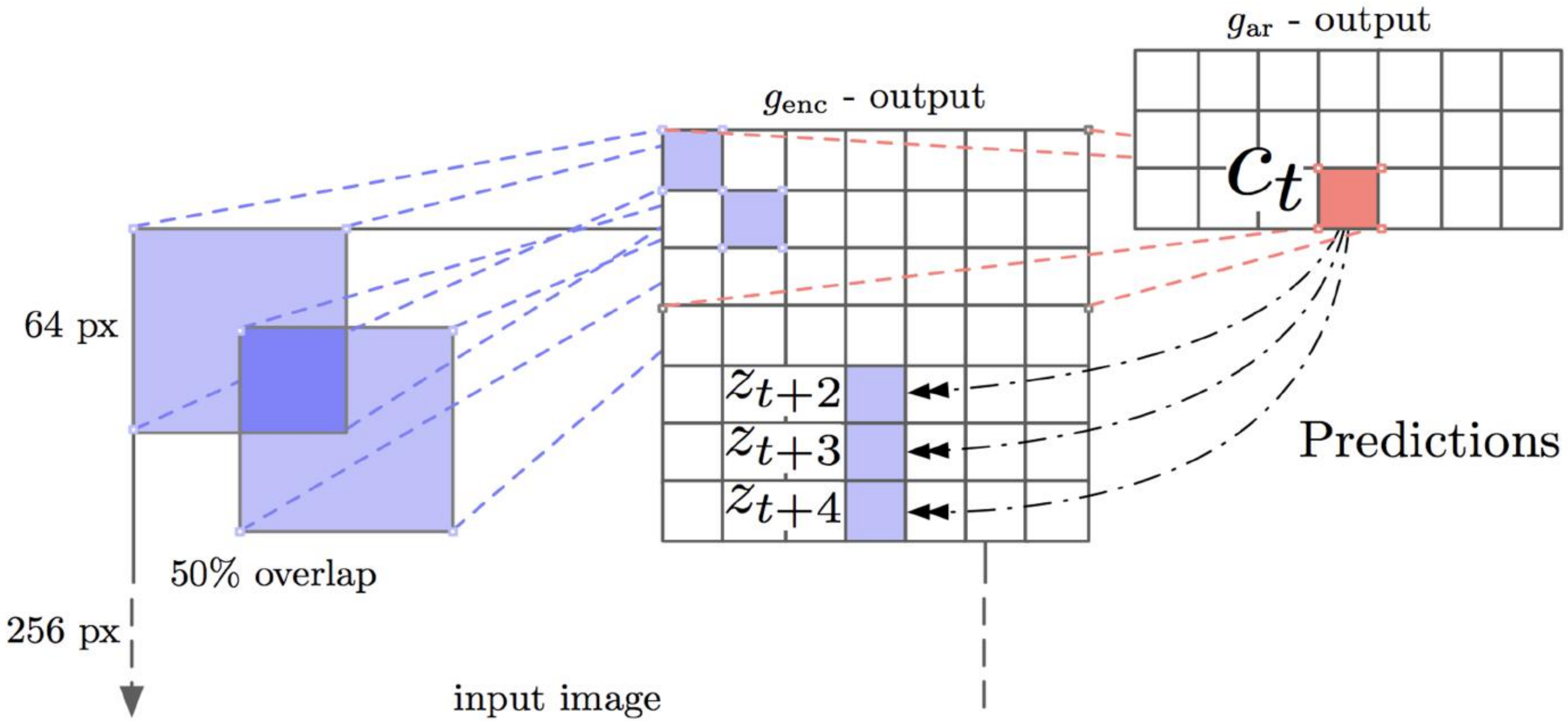


Figure 4: Visualization of Contrastive Predictive Coding for images (2D adaptation of Figure 1).

CPC - ImageNet

CPC - ImageNet

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

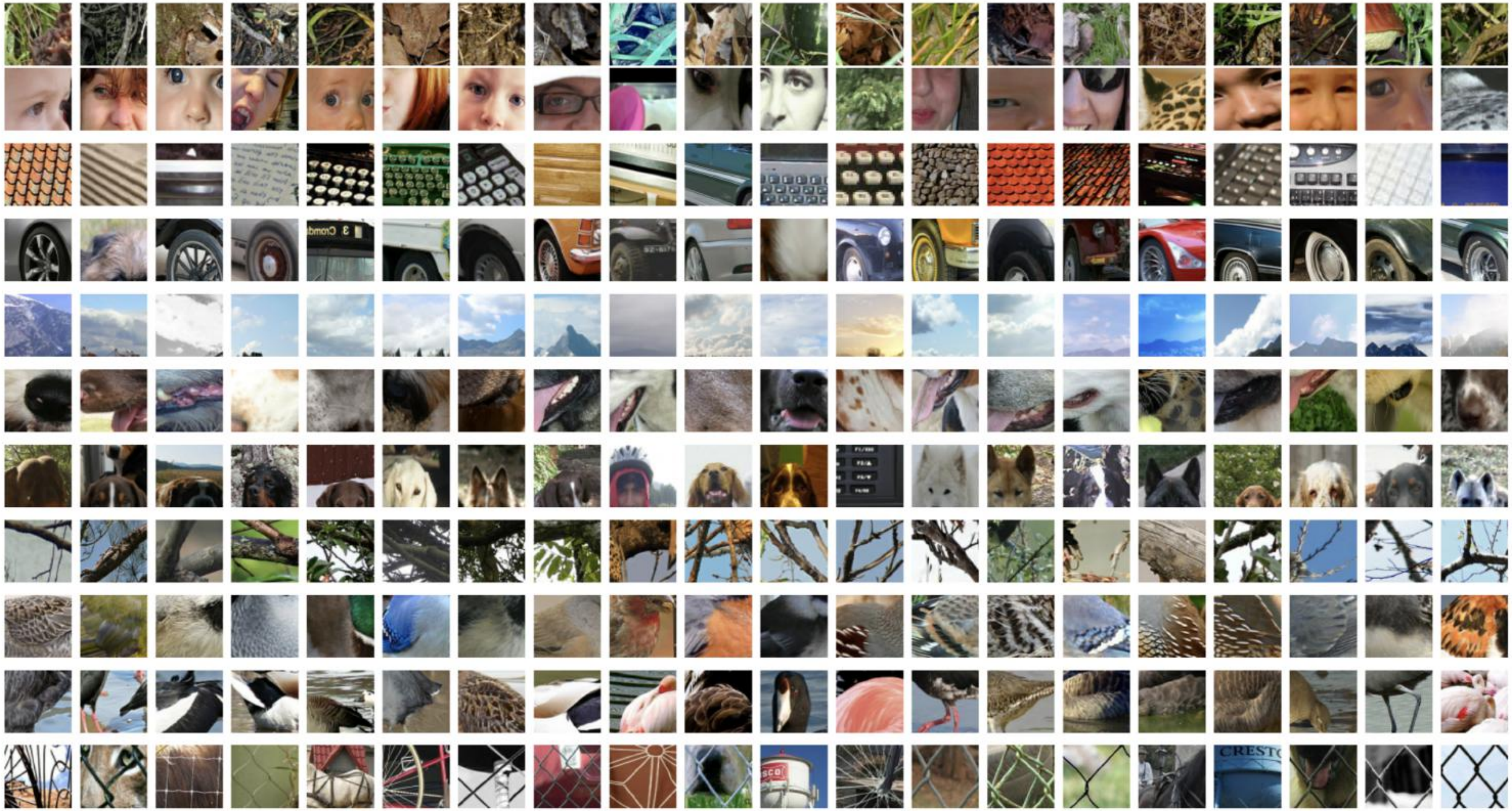


Figure 5: Every row shows image patches that activate a certain neuron in the CPC architecture.

CPC - Natural Language Processing

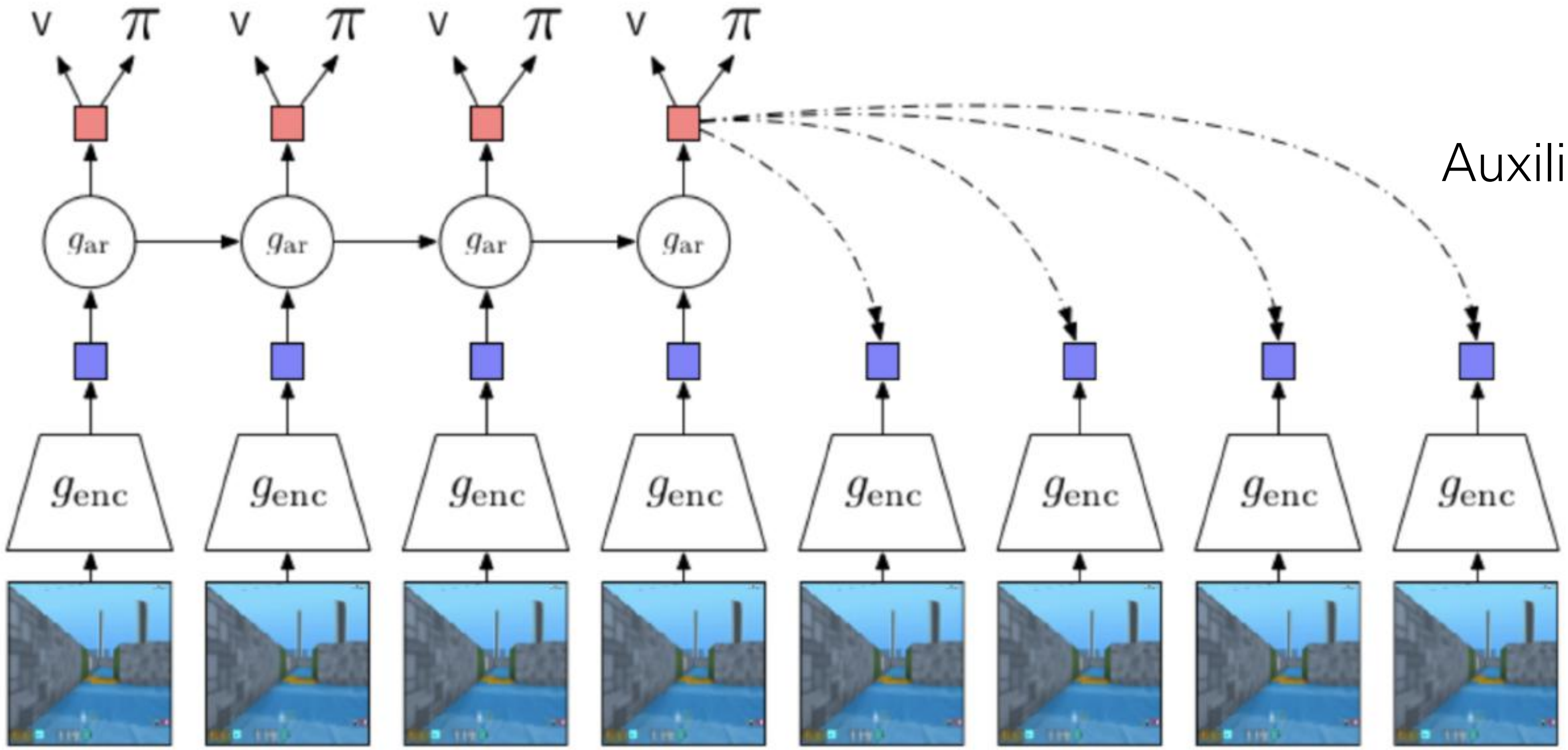
Method	MR	CR	Subj	MPQA	TREC
Paragraph-vector [40]	74.8	78.1	90.5	74.2	91.8
Skip-thought vector [26]	75.5	79.3	92.1	86.9	91.4
Skip-thought + LN [41]	79.5	82.6	93.4	89.0	-
CPC	76.9	80.1	91.2	87.7	96.8

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.

Oord, Li, Vinyals, 2018

CPC - Reinforcement Learning

Auxiliary loss is on policy
Predict 30 steps in the future



Auxiliary Losses

Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

Data-Efficient Image Recognition with Contrastive Predictive Coding

Olivier J. Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi,
Carl Doersch, S. M. Ali Eslami, Aaron van den Oord
DeepMind
London, UK

Abstract

Human observers can learn to recognize new categories of images from a handful of examples, yet doing so with machine perception remains an open challenge. We hypothesize that data-efficient recognition is enabled by representations which make the variability in natural signals more predictable. We therefore revisit and improve Contrastive Predictive Coding, an unsupervised objective for learning such representations. This new implementation produces features which support state-of-the-art linear classification accuracy on the ImageNet dataset. When used as input for non-linear classification with deep neural networks, this representation allows us to use 2–5× less labels than classifiers trained directly on image pixels. Finally, this unsupervised representation substantially improves transfer learning to object detection on PASCAL VOC-2007, surpassing fully supervised pre-trained ImageNet classifiers.

CPCv2 - Large Scale CPC on ImageNet

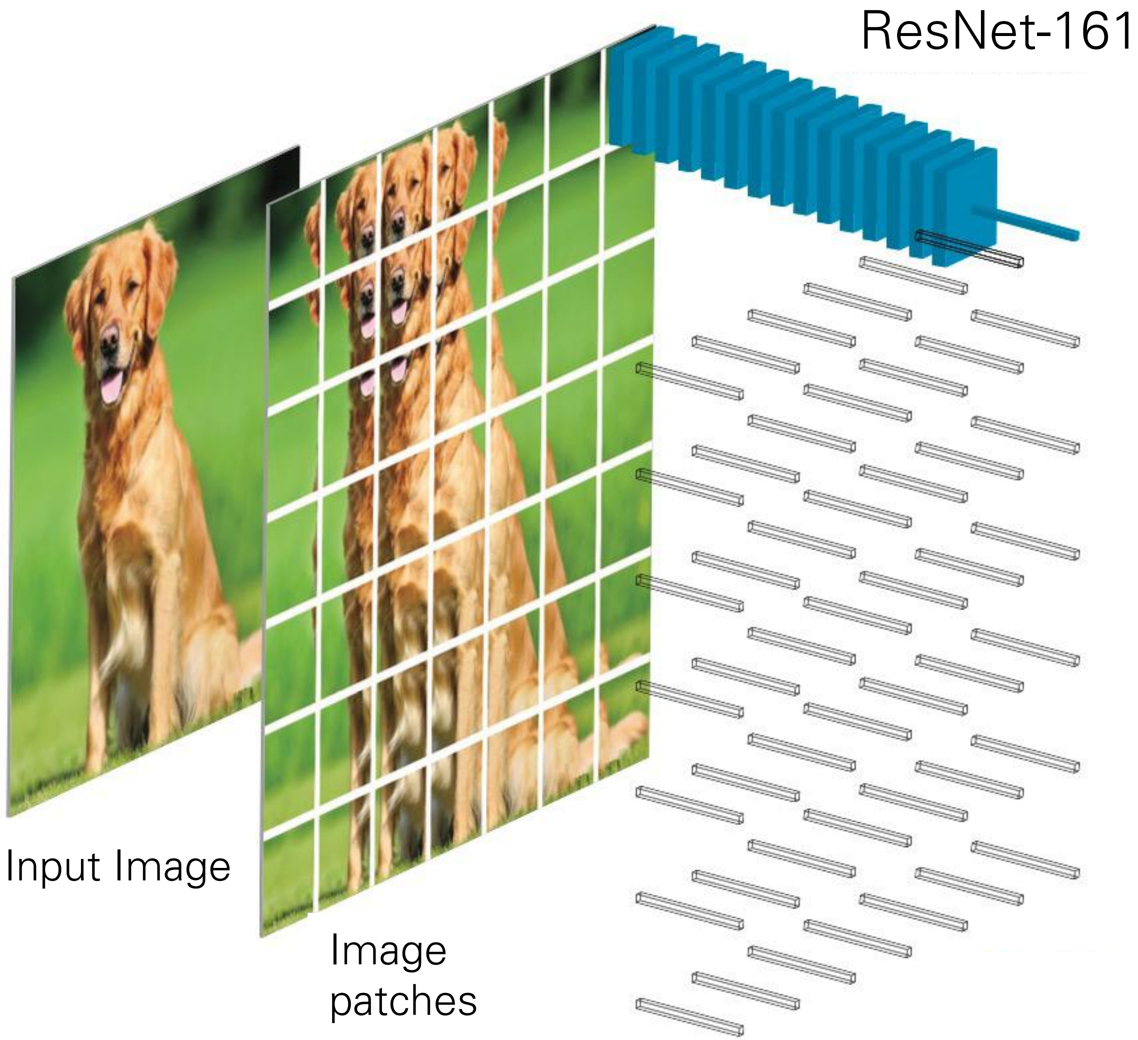


Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

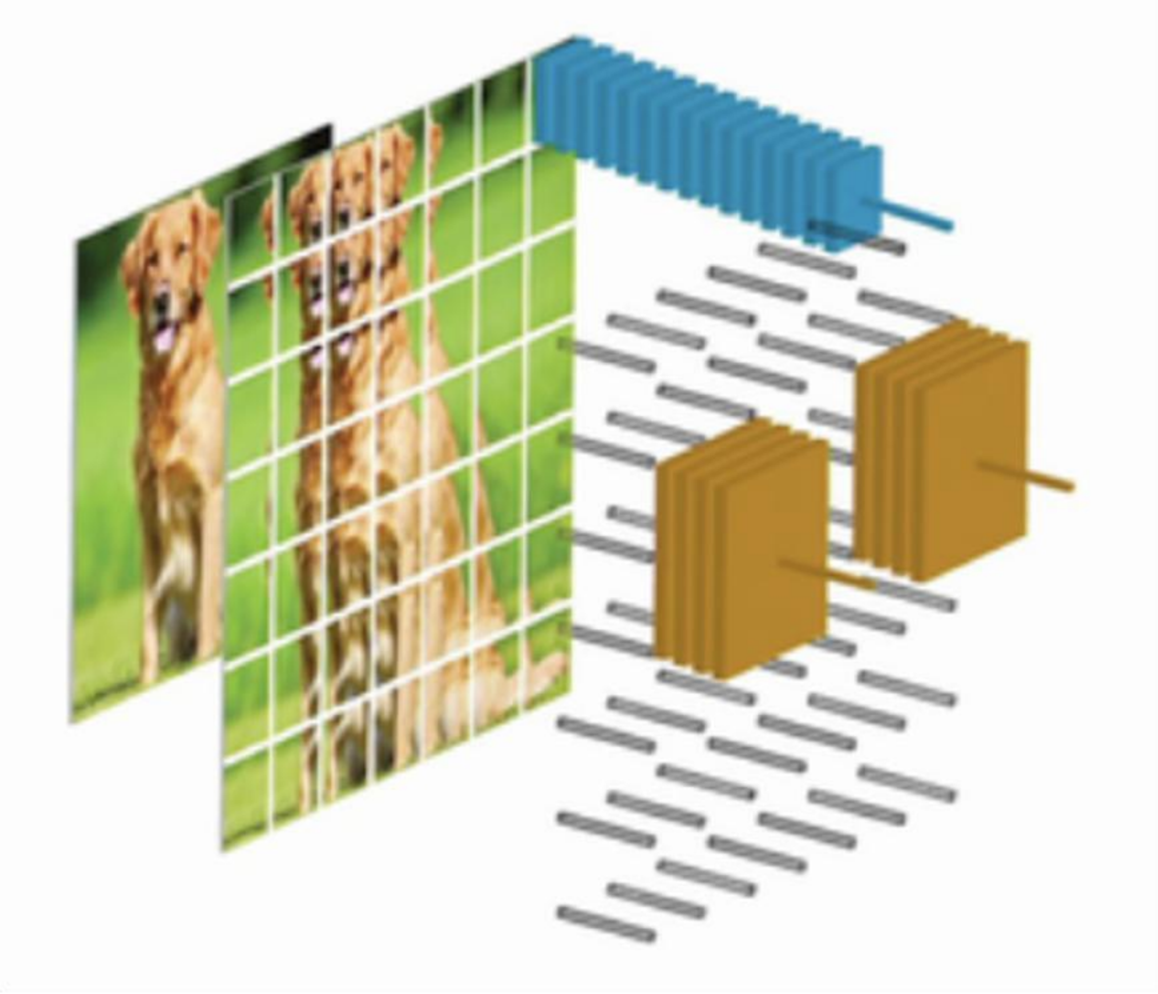


Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

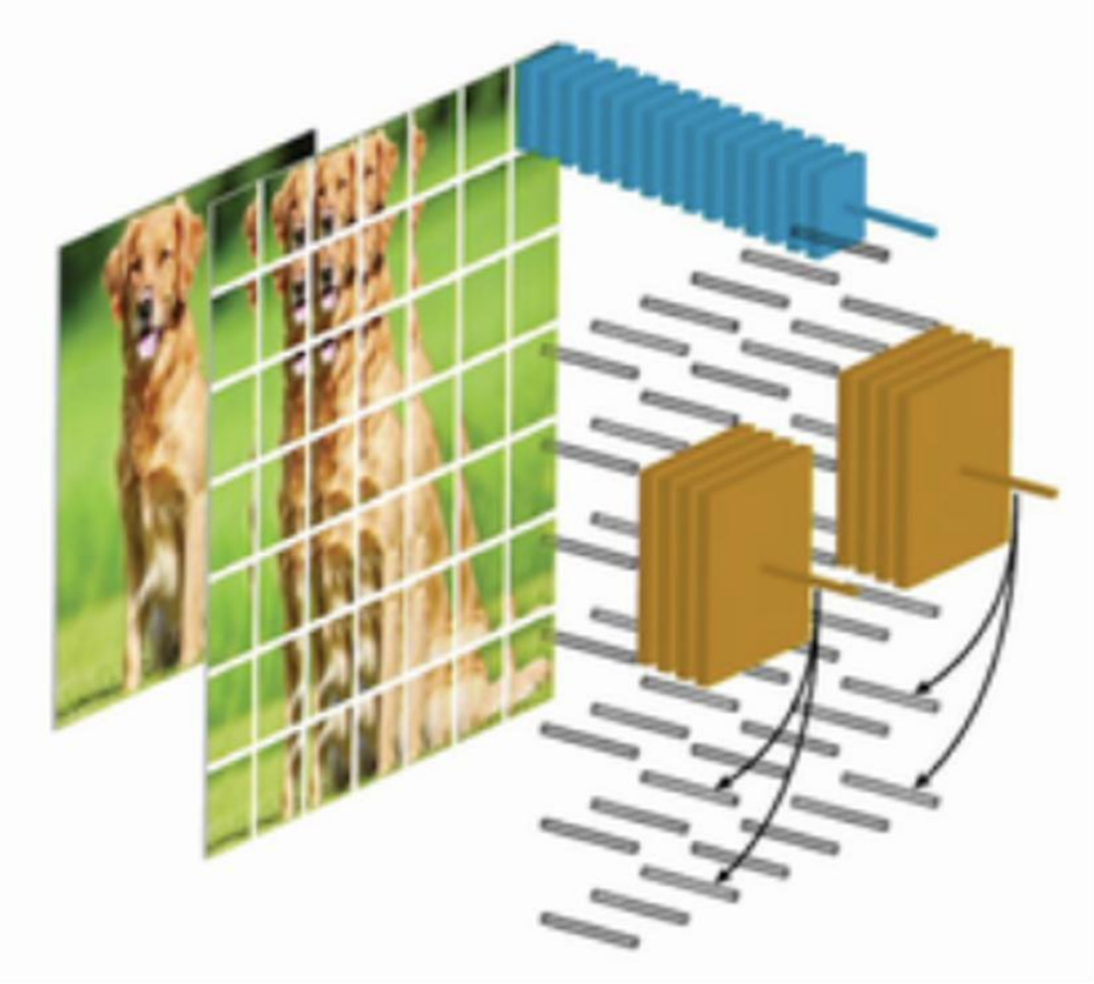


Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

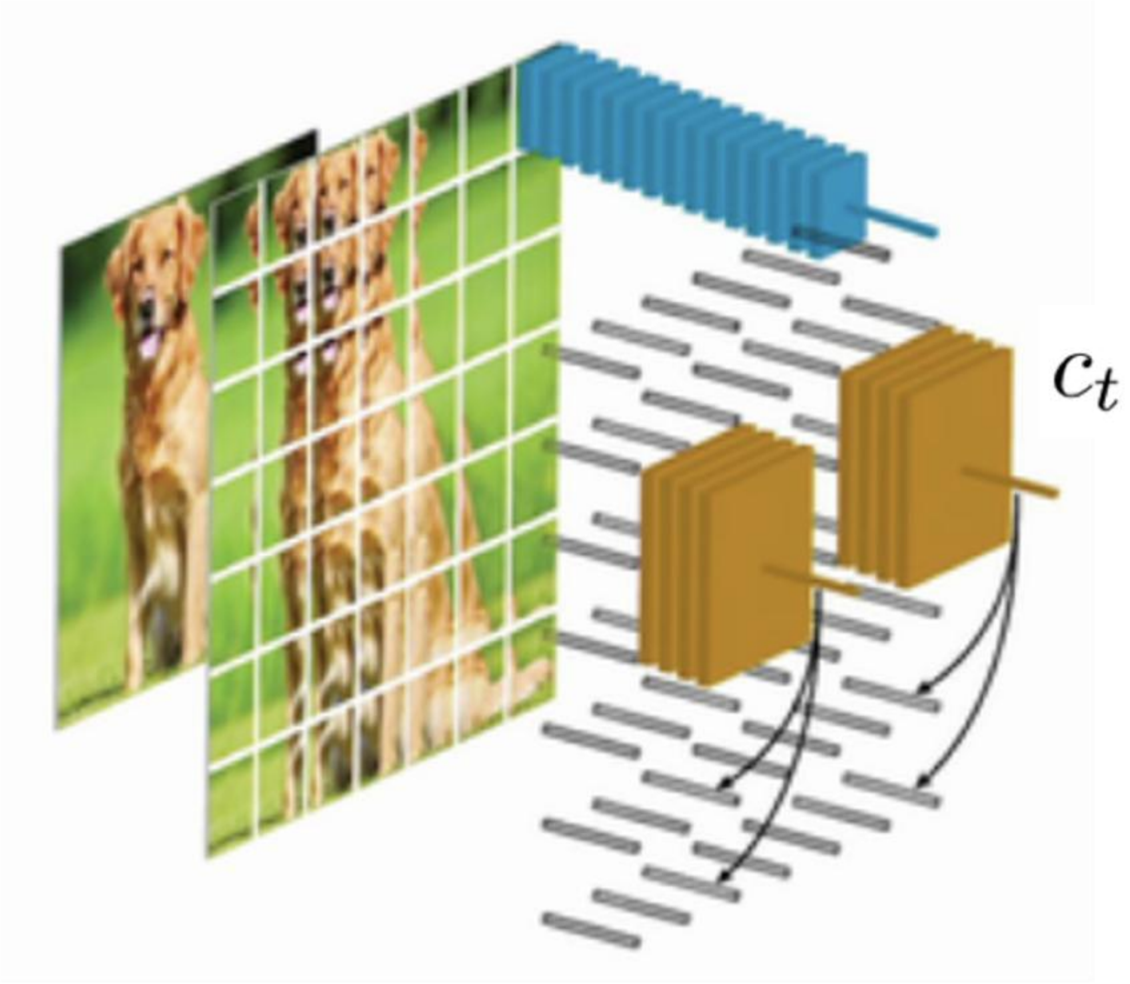


Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

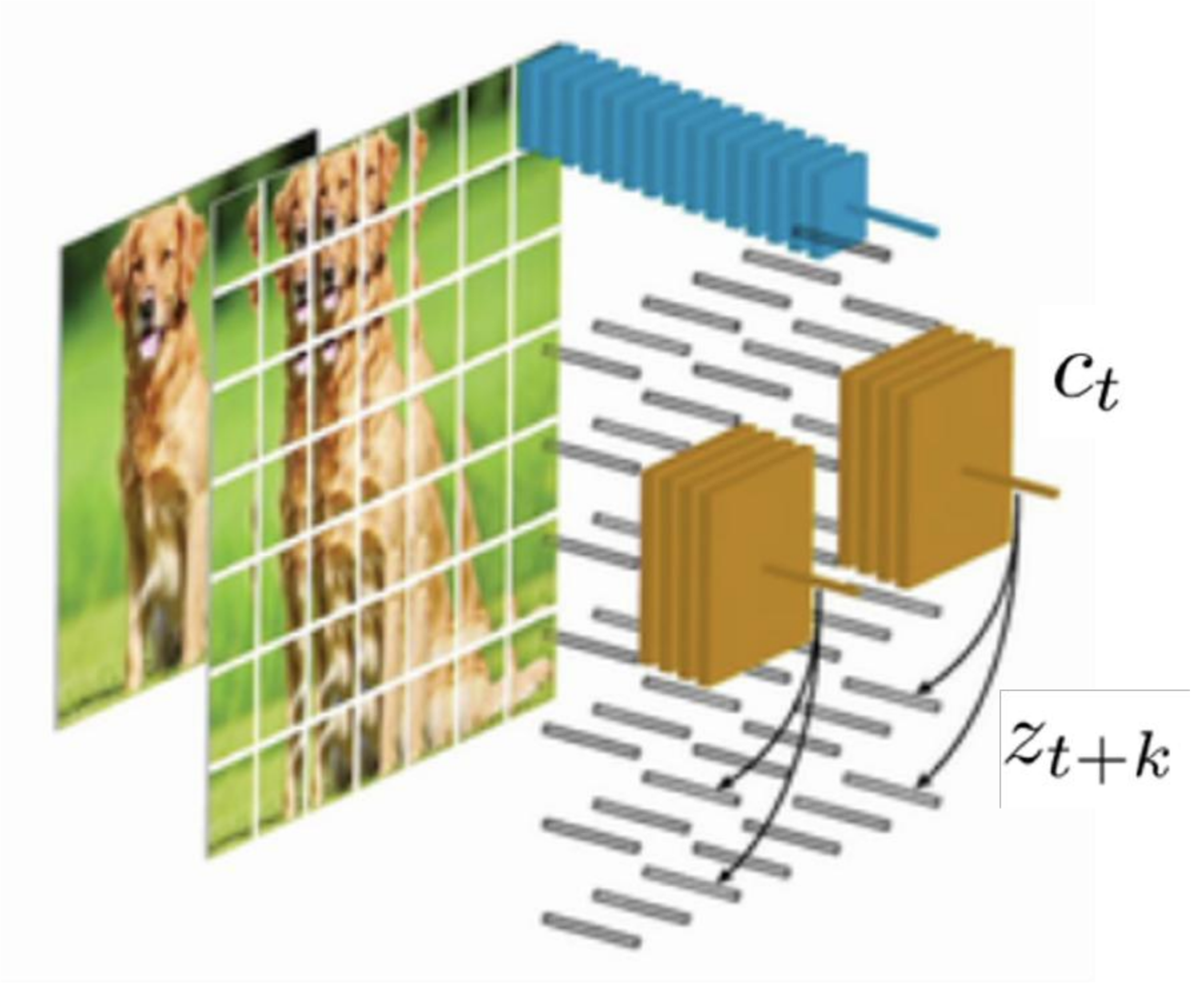
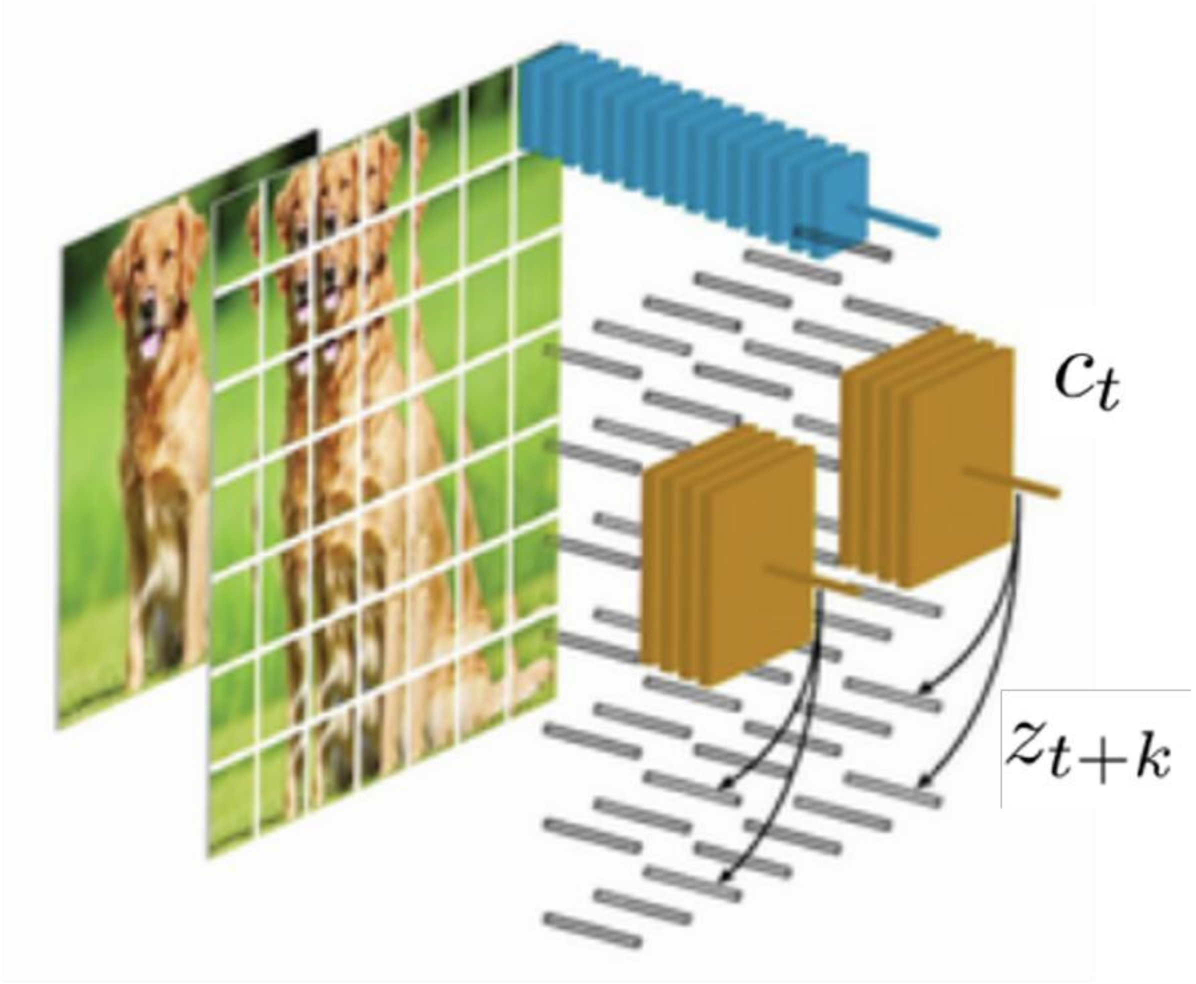


Figure credit: Aaron Van den Oord

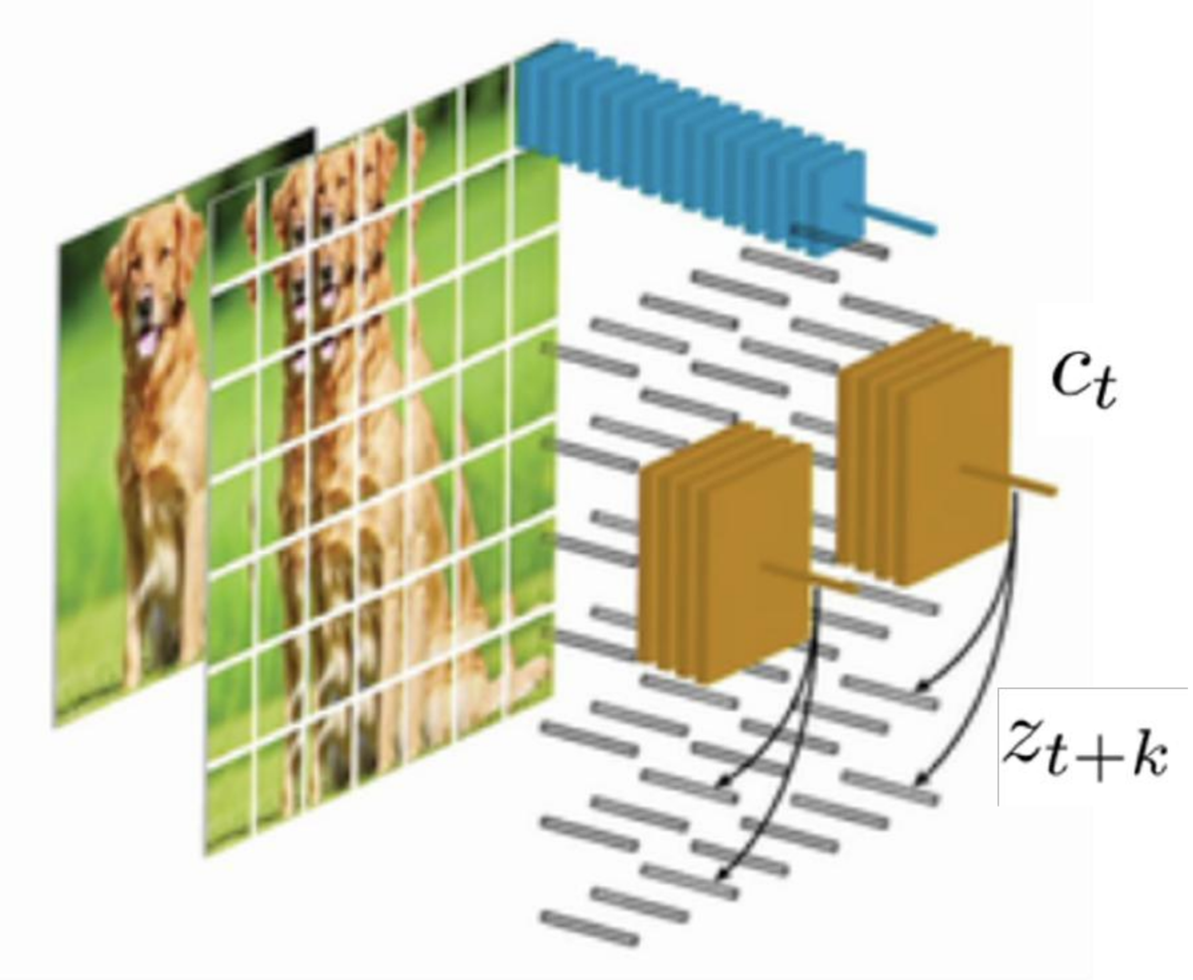
CPCv2 - Large Scale CPC on ImageNet



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

Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

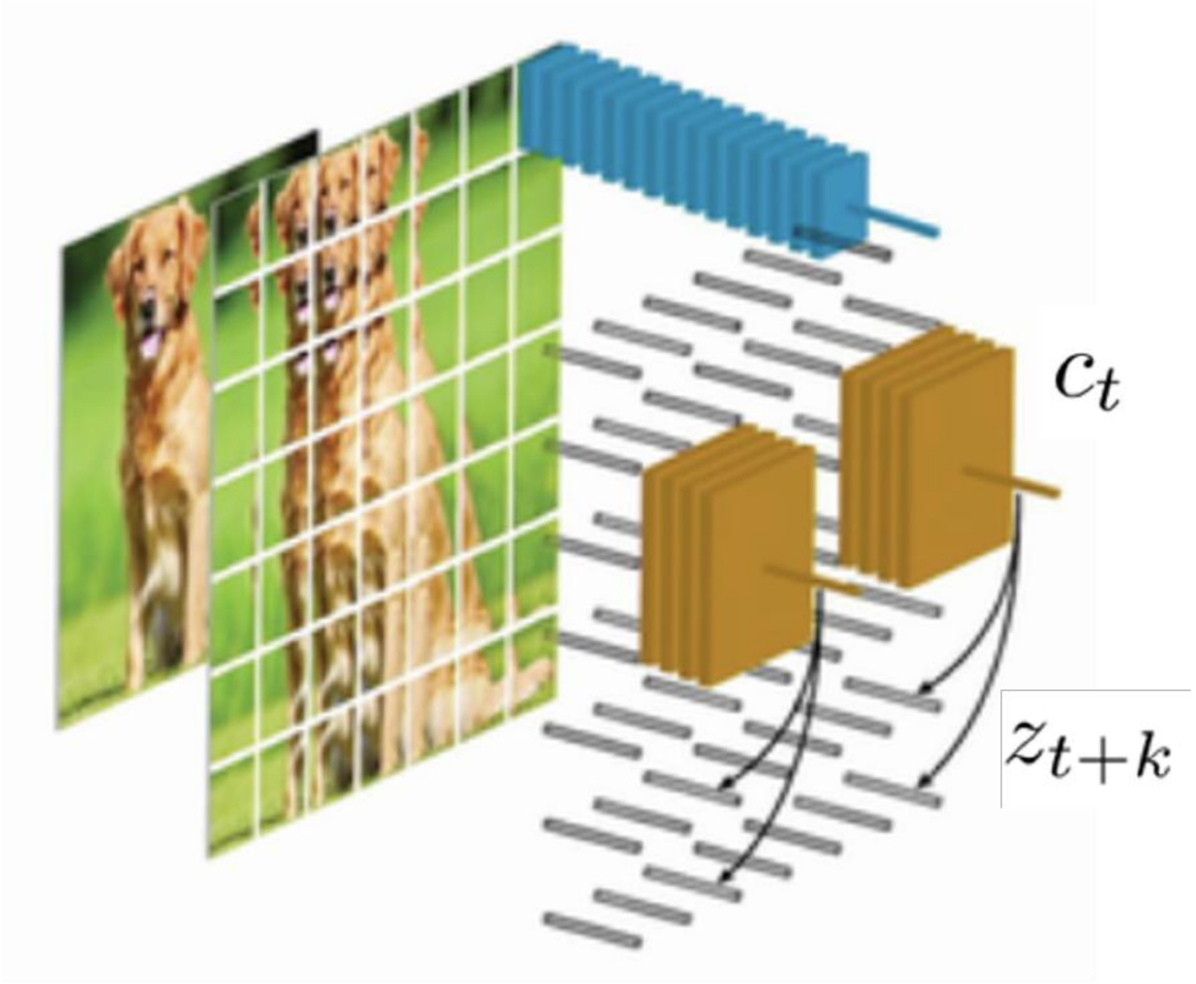


$$f_k(x_{t+k}, c_t) = \exp\left(z_{t+k}^T W_k c_t\right)$$

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

Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet



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

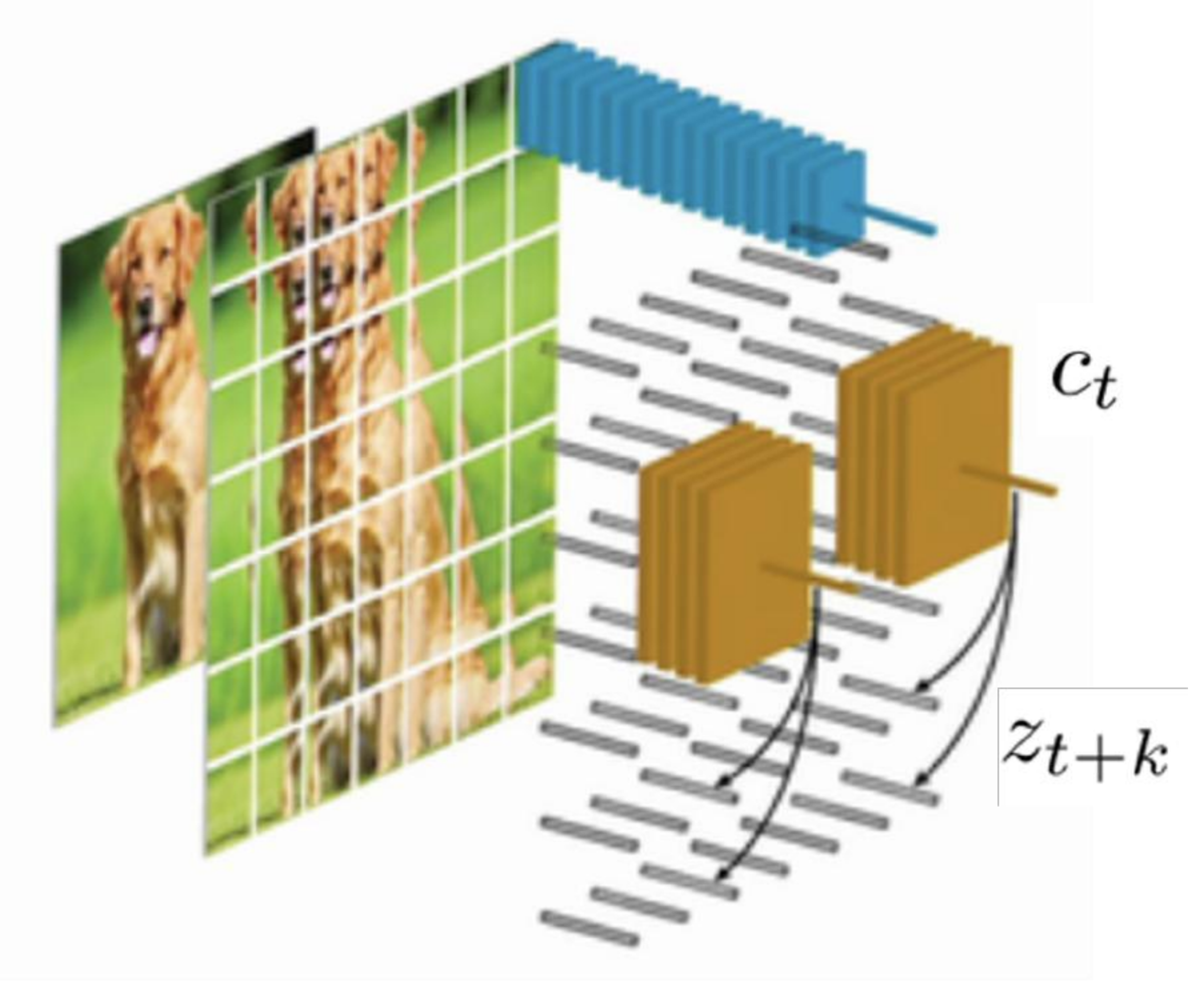
$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

Negatives

1. Other patches within image
2. Patches from other images

Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet



InfoNCE Loss

NCE: Noise-Contrastive Estimation

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

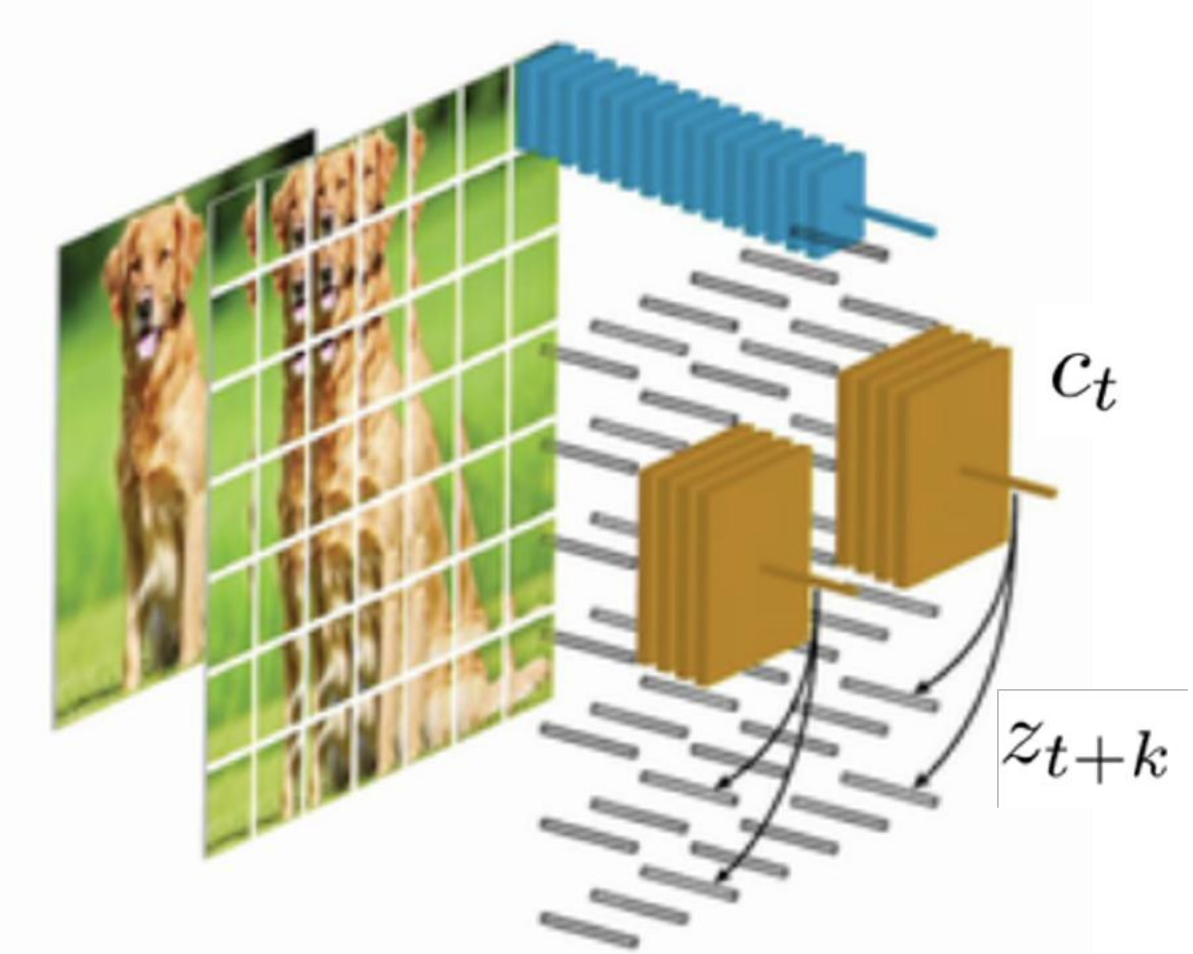
$$\mathcal{L}_{\mathrm{N}} = -\mathop{\mathbb{E}}_{X}\left[\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}\right]$$

Negatives

1. Other patches within image
2. Patches from other images

Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet



Parallel Implementation with PixelCNN (masked conv) and 1x1 conv

InfoNCE Loss

NCE: Noise-Contrastive Estimation

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

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

Negatives

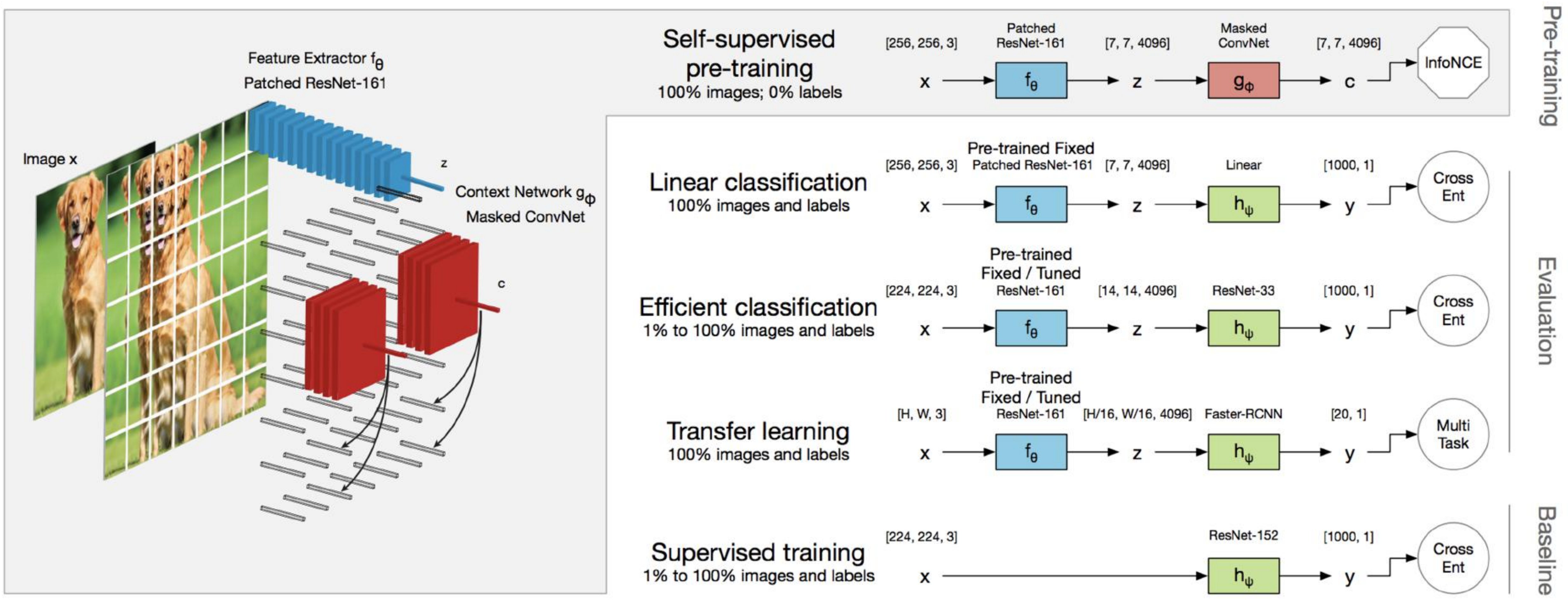
1. Other patches within image
2. Patches from other images

Figure credit: Aaron Van den Oord

CPCv2 - Large Scale CPC on ImageNet

- Train CPC on unlabeled ImageNet
- Train as long as possible (500 epochs) – 1 week
- Augment every patch with a lot of spatial and color augmentation [**extremely crucial**]
- Effective number of negatives = number of instances * number of patches per instance = 16 * 36 = 576

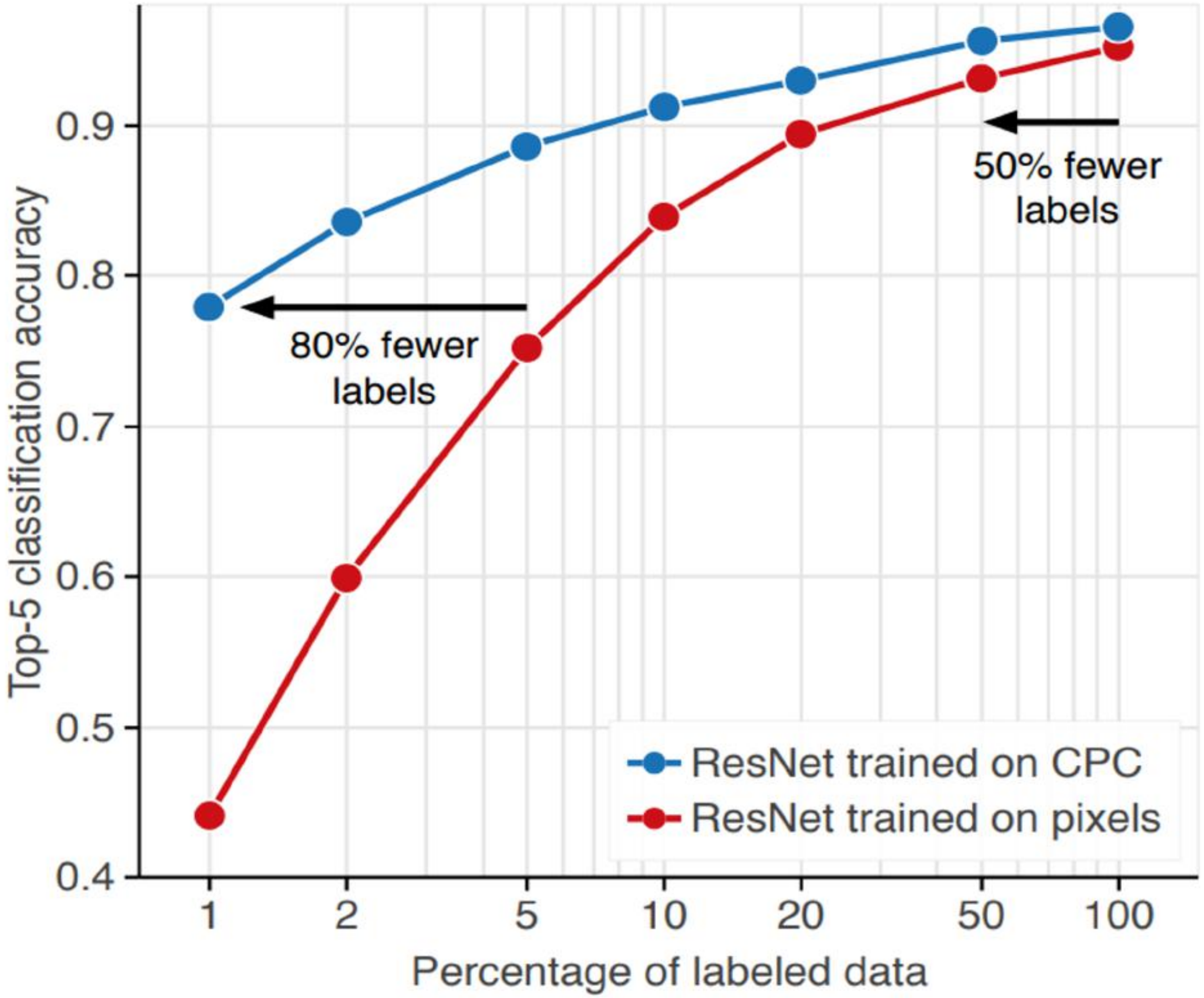
CPCv2 - Large Scale CPC on ImageNet



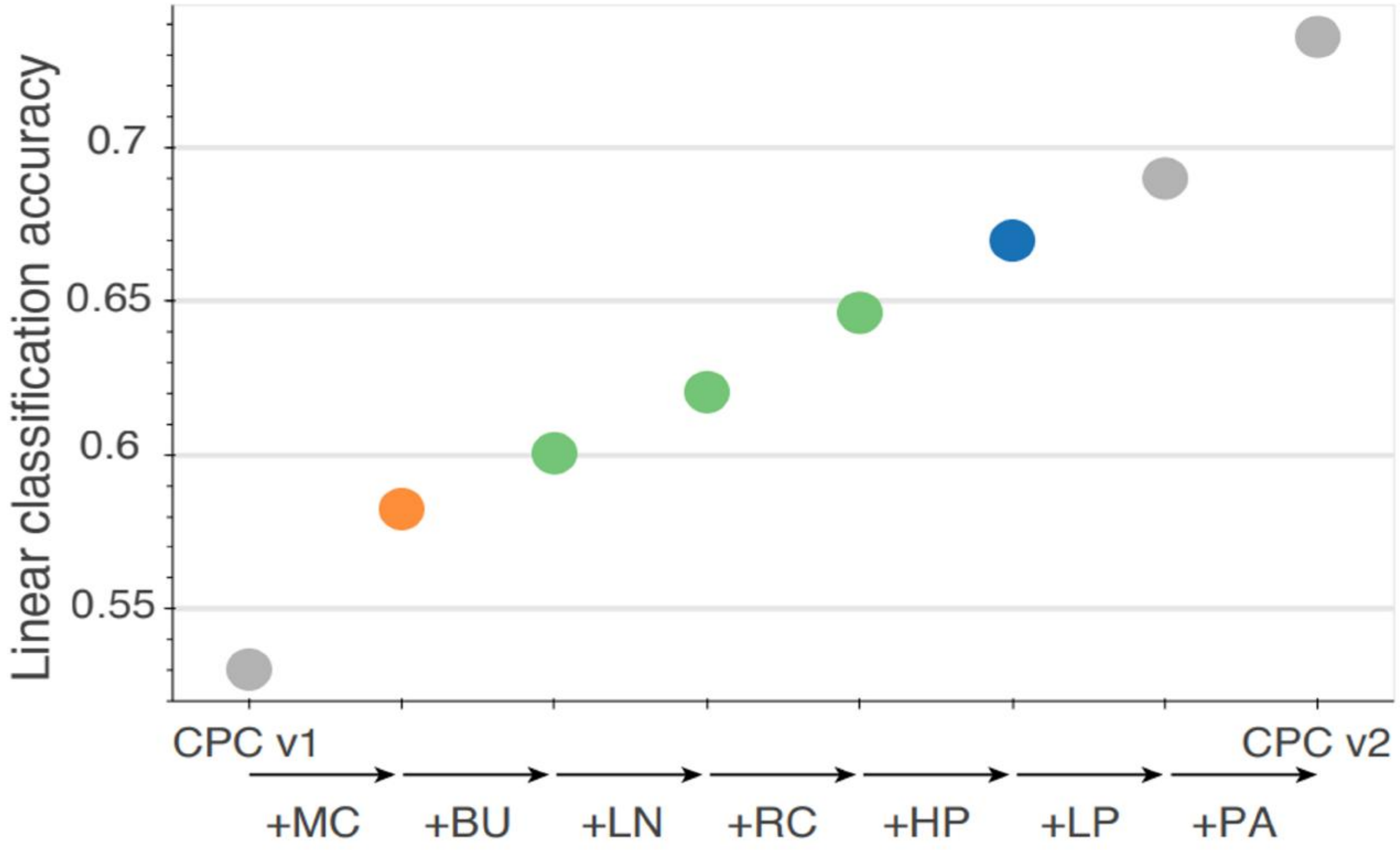
CPCv2 - Linear Classification

Method	Params (M)	Top-1	Top-5
Methods using ResNet-50:			
Instance Discr. [1]	24	54.0	-
Local Aggr. [2]	24	58.8	-
MoCo [3]	24	60.6	-
PIRL [4]	24	63.6	-
CPC v2 - ResNet-50	24	**63.8**	**85.3**
Methods using different architectures:			
Multi-task [5]	28	-	69.3
Rotation [6]	86	55.4	-
CPC v1 [7]	28	48.7	73.6
BigBiGAN [8]	86	61.3	81.9
AMDIM [9]	626	68.1	-
CMC [10]	188	68.4	88.2
MoCo [2]	375	68.6	-
CPC v2 - ResNet-161	305	**71.5**	**90.1**

CPCv2 - Data-Efficient Image Recognition



CPCv1 → CPCv2



MC: model capacity

BU: bottom-up spatial predictions

LN: layer normalization

RC: random color-dropping

HP: horizontal spatial predictions

LP: larger patches.

PA: further patch-based augmentation.

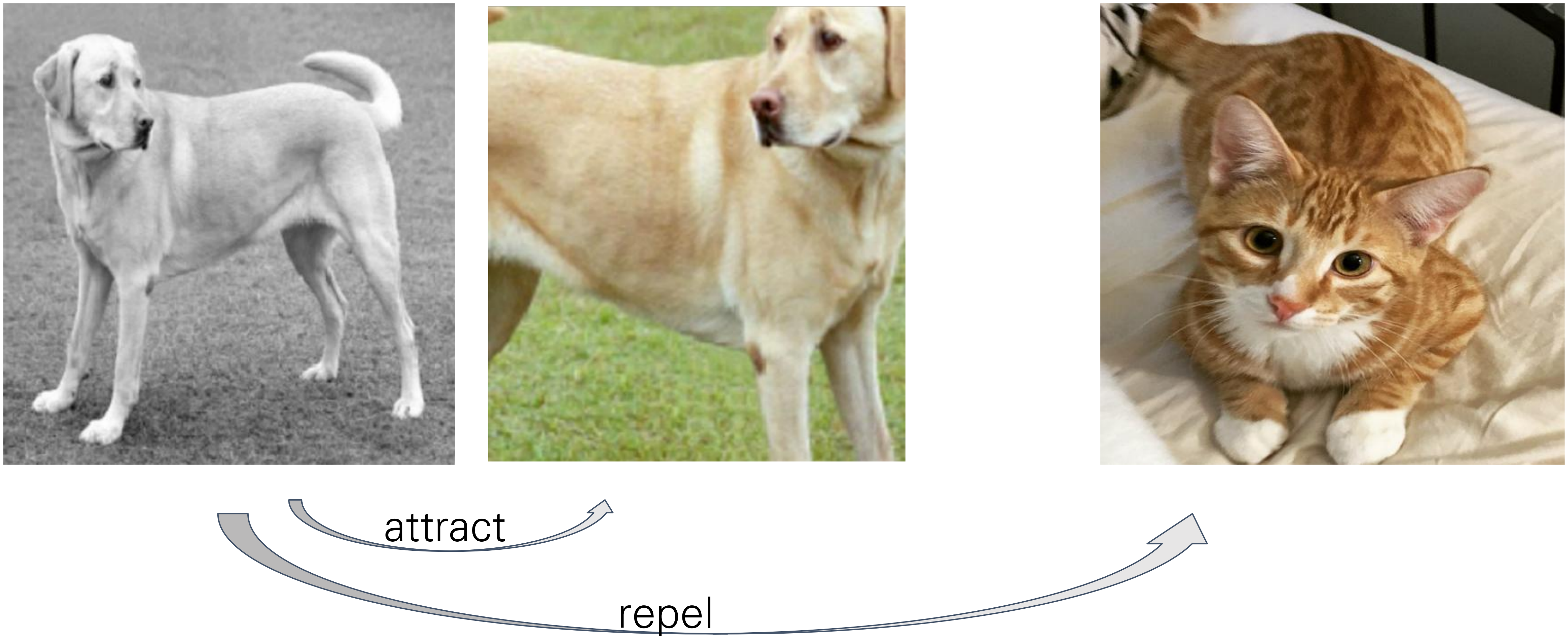
CPCv2 - Data-Efficient Supervised Learning

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 $\times 4$	-	-	**91.2**	-	95.0
Methods using representation learning only:						
Instance Discrimination [60]	ResNet-50	39.2	-	77.4	-	-
Rotation [63]	ResNet-152 $\times 2$	57.5	-	86.4	-	-
ResNet on BigBiGAN (fixed)	RevNet-50 $\times 4$	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**

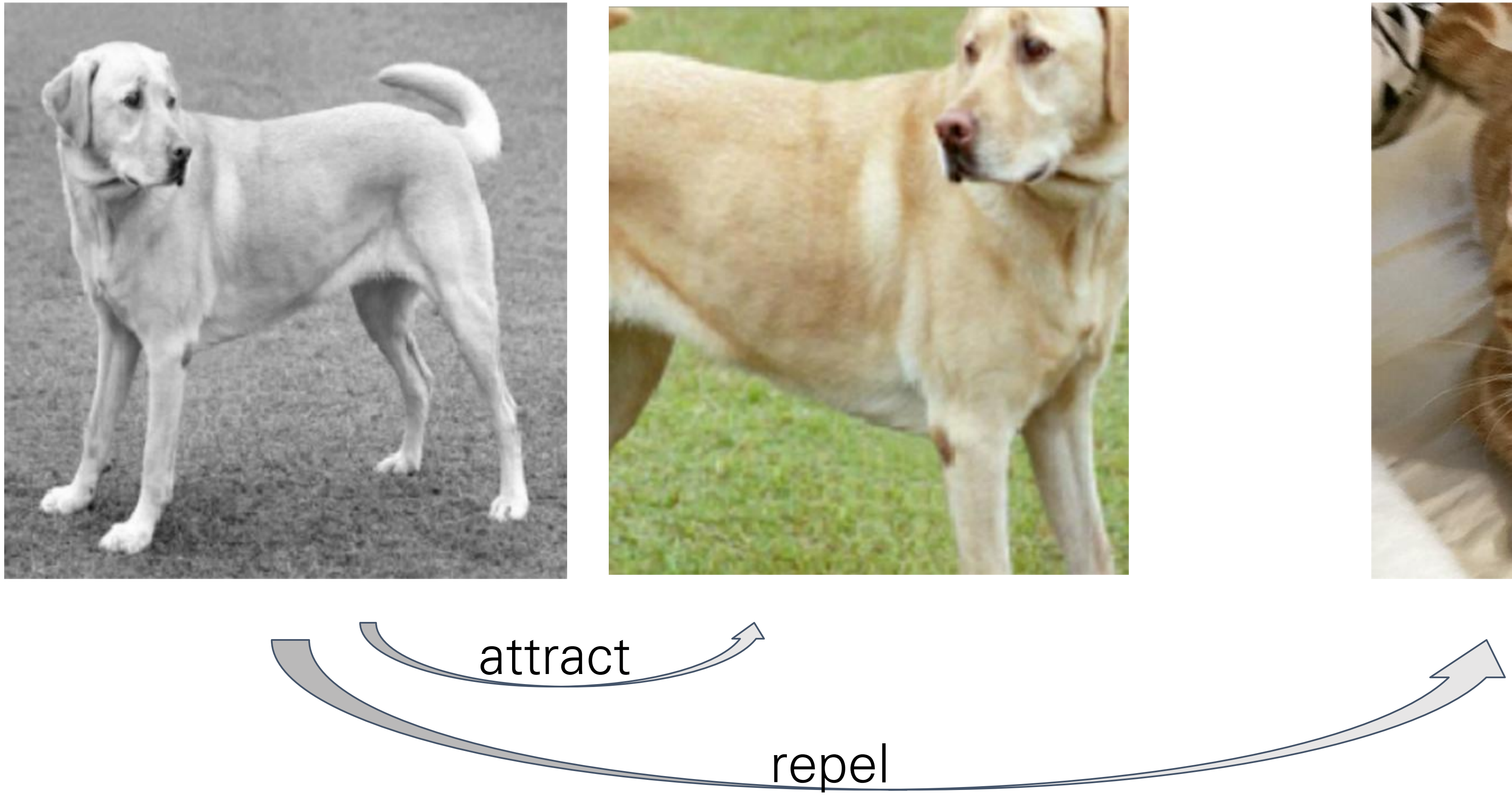
CPCv2 - PASCAL VOC-07 Detection

Method	Architecture	mAP
Transfer from labeled data:		
Supervised baseline	ResNet-152	74.7
Transfer from unlabeled data:		
Exemplar [17] by [13]	ResNet-101	60.9
Motion Segmentation [47] by [13]	ResNet-101	61.1
Colorization [64] by [13]	ResNet-101	65.5
Relative Position [14] by [13]	ResNet-101	66.8
Multi-task [13]	ResNet-101	70.5
Instance Discrimination [60]	ResNet-50	65.4
Deep Cluster [7]	VGG-16	65.9
Deeper Cluster [8]	VGG-16	67.8
Local Aggregation [66]	ResNet-50	69.1
Momentum Contrast [25]	ResNet-50	74.9
Faster-RCNN trained on CPC v2	ResNet-161	**76.6**

Instance Discrimination



Instance Discrimination



1. MoCo
2. SimCLR

Momentum Contrast (MoCo)

Momentum Contrast for Unsupervised Visual Representation Learning

Kaiming He Haoqi Fan Yuxin Wu Saining Xie Ross Girshick

Abstract

We present Momentum Contrast (MoCo) for unsupervised visual representation learning. From a perspective on contrastive learning [29] as dictionary look-up, we build a dynamic dictionary with a queue and a moving-averaged encoder. This enables building a large and consistent dictionary on-the-fly that facilitates contrastive unsupervised learning. MoCo provides competitive results under the common linear protocol on ImageNet classification. More importantly, the representations learned by MoCo transfer well to downstream tasks. MoCo can ***outperform*** *its supervised pre-training counterpart in* ***7*** *detection/segmentation tasks on PASCAL VOC, COCO, and other datasets, sometimes surpassing it by large margins. This suggests that the gap between unsupervised and supervised representation learning has been largely closed in many vision tasks.*

Momentum Contrast (MoCo)

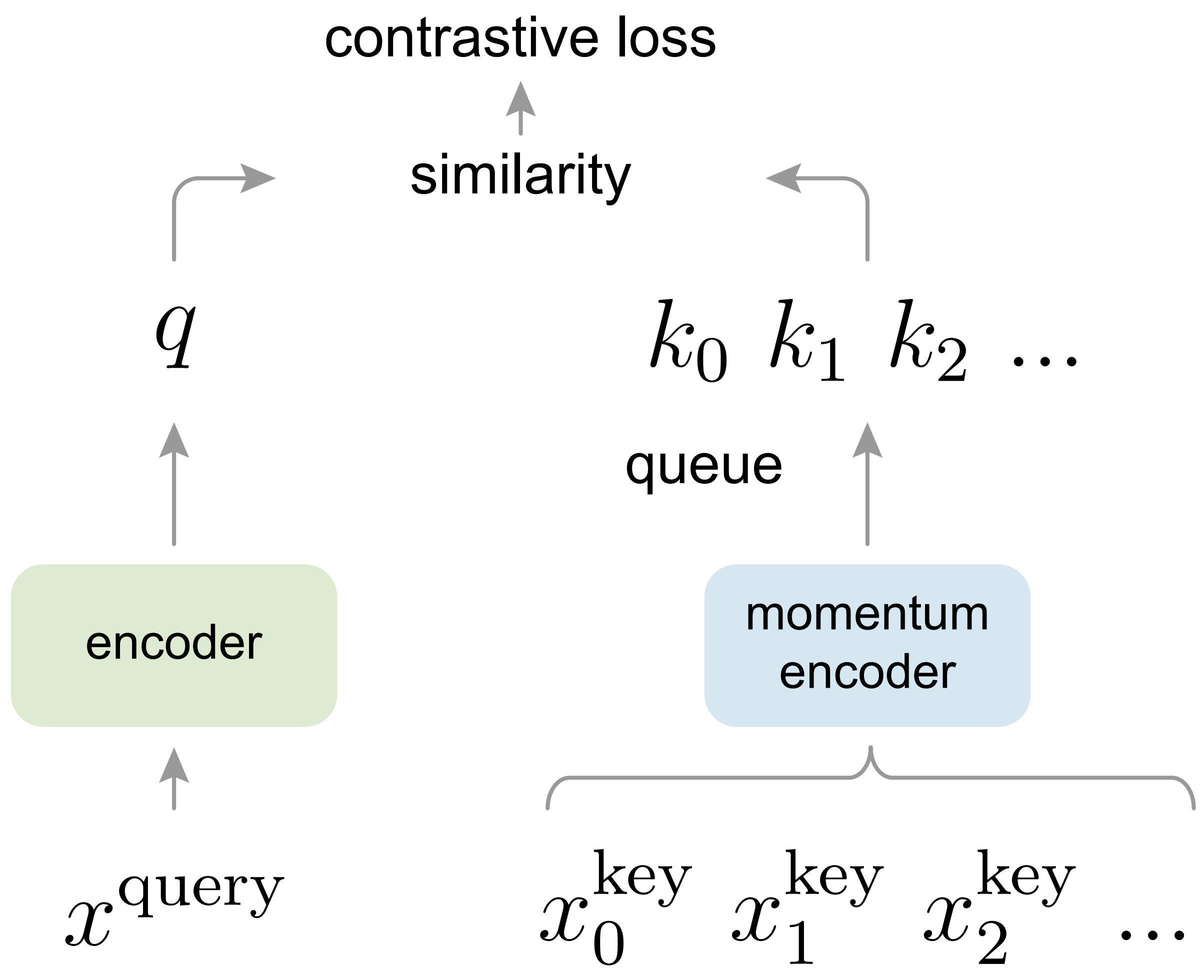
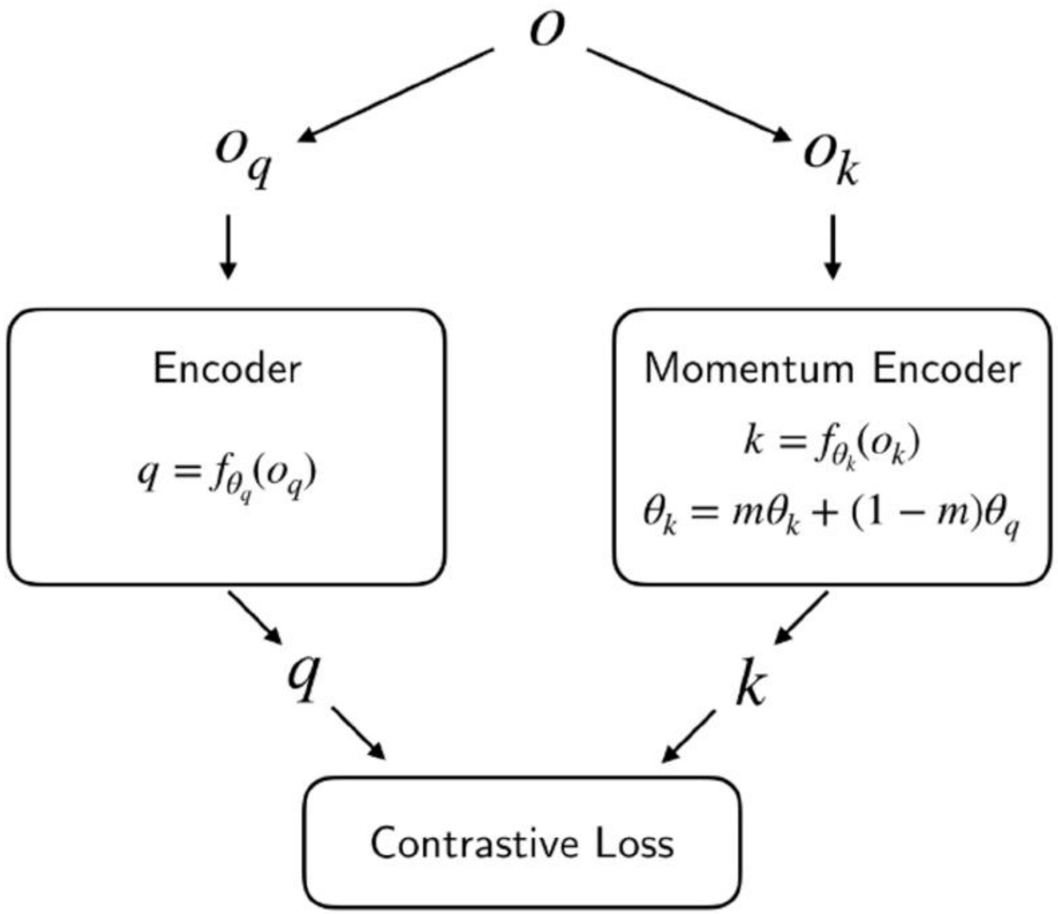


Figure 1. Momentum Contrast (MoCo) trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. The dictionary keys $\{k_0, k_1, k_2, ...\}$ are defined on-the-fly by a set of data samples. The dictionary is built as a queue, with the 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.

Momentum Contrast (MoCo)



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

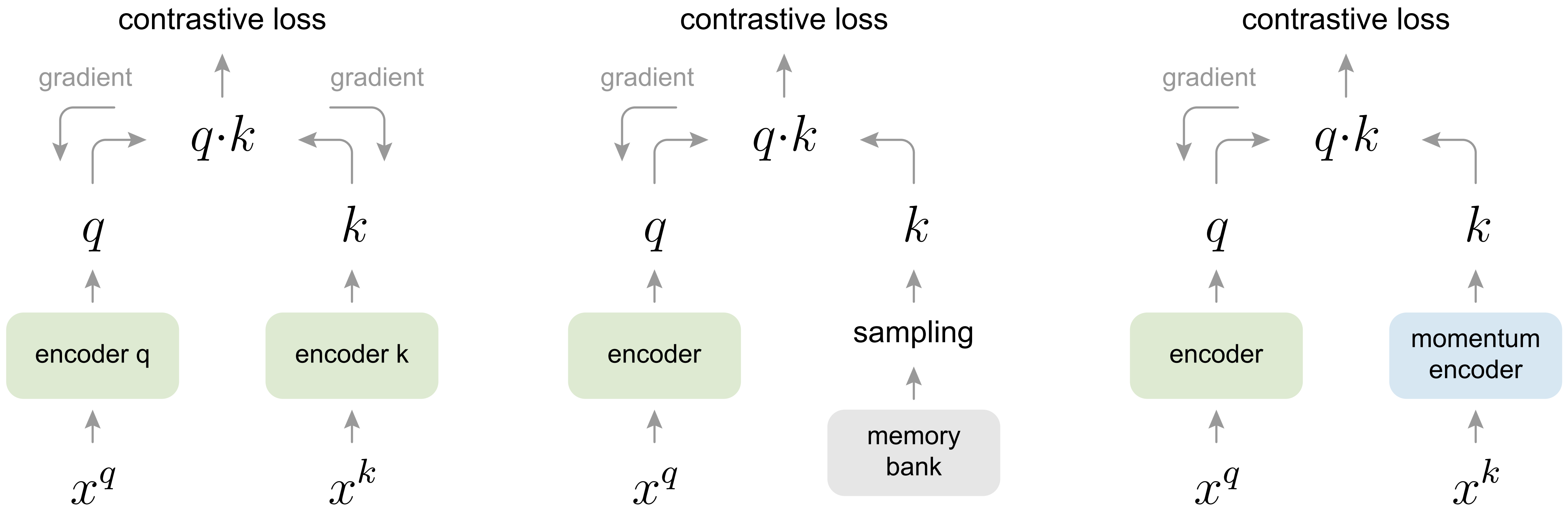
Momentum Contrast (MoCo)

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
```

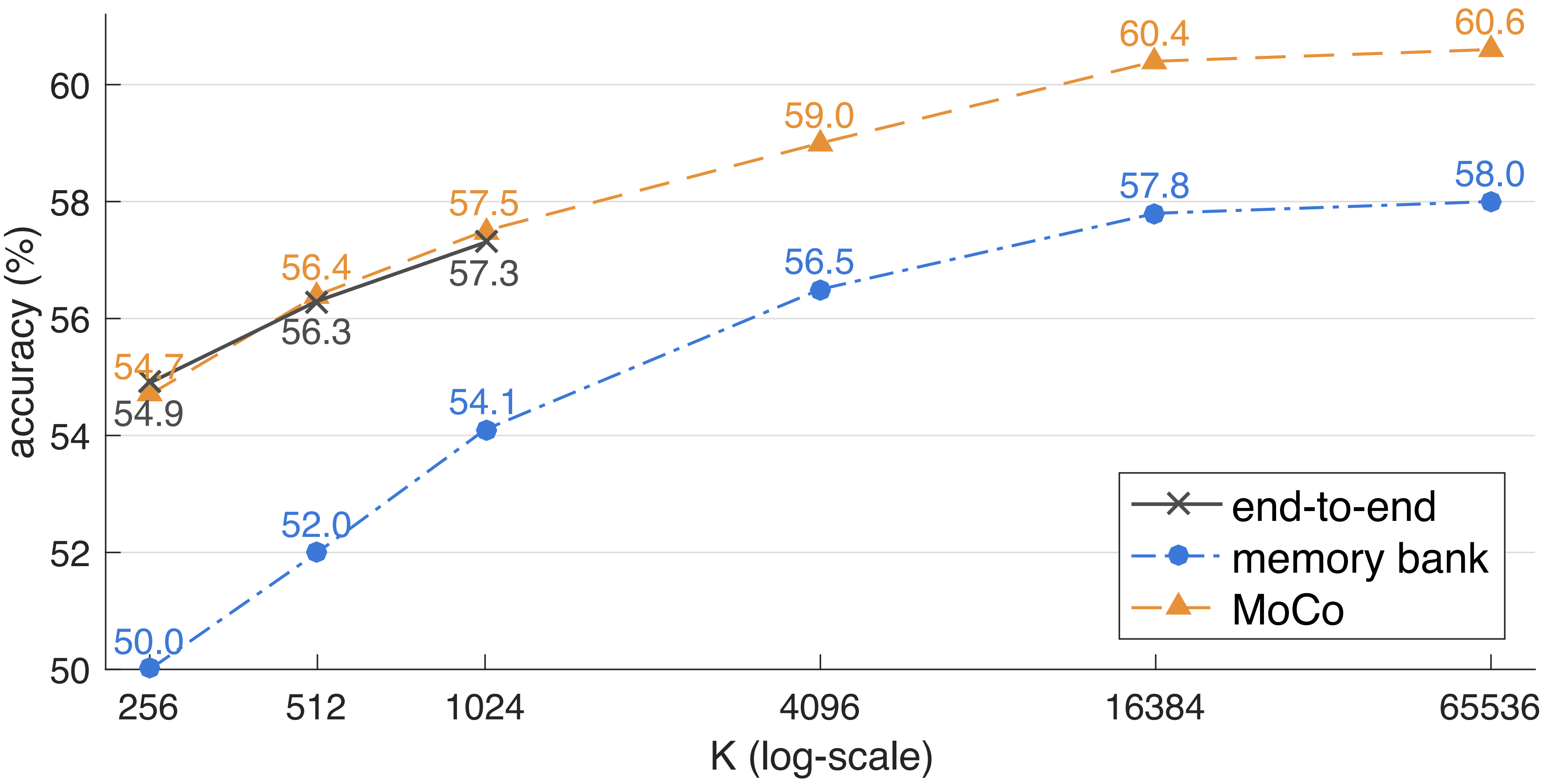
bmm: batch matrix multiplication; mm: matrix multiplication; cat: concatenation.

Momentum Contrast (MoCo)

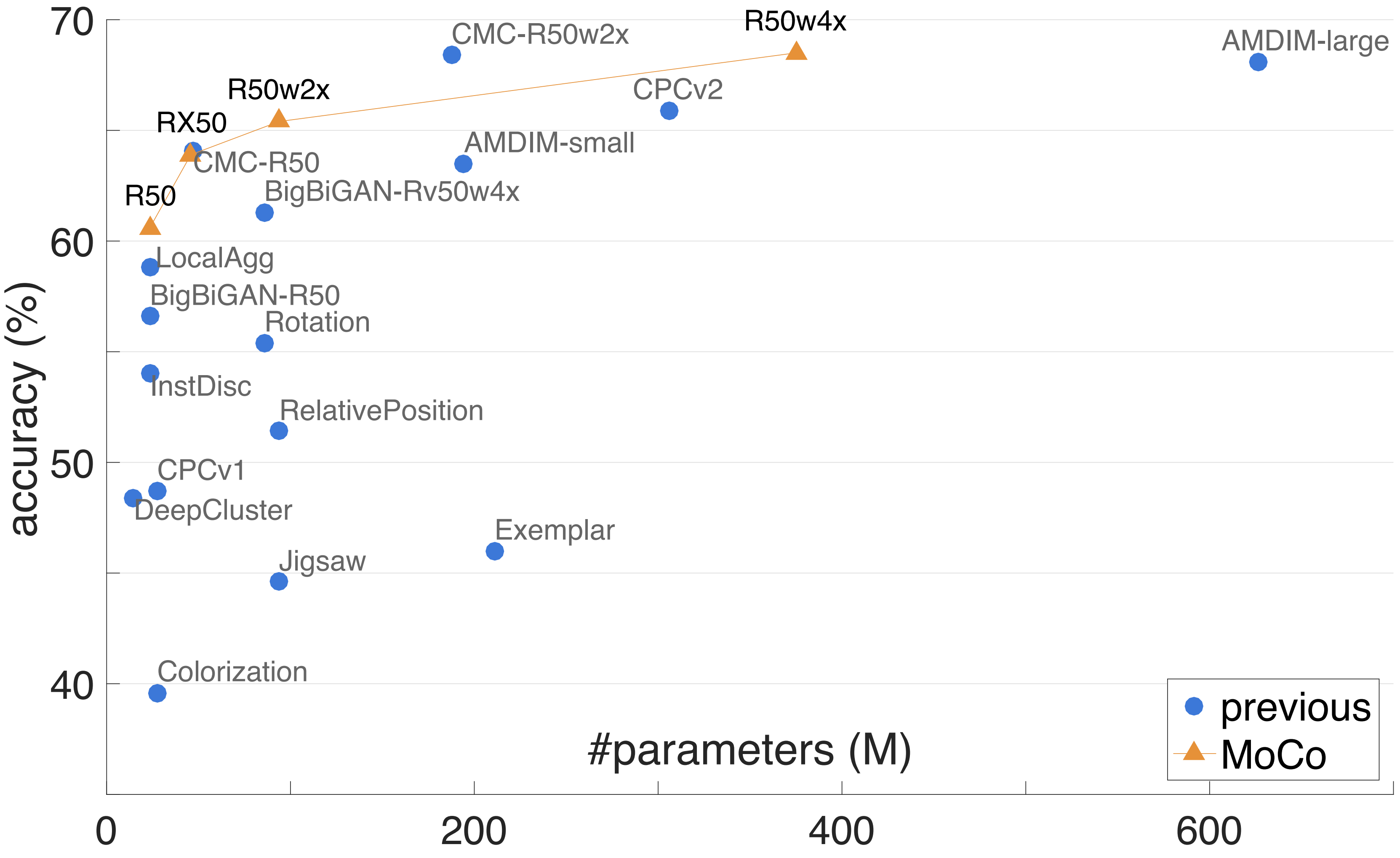


(a) end-to-end (b) memory bank (c) MoCo

Momentum Contrast (MoCo)



Momentum Contrast (MoCo)



SimCLR

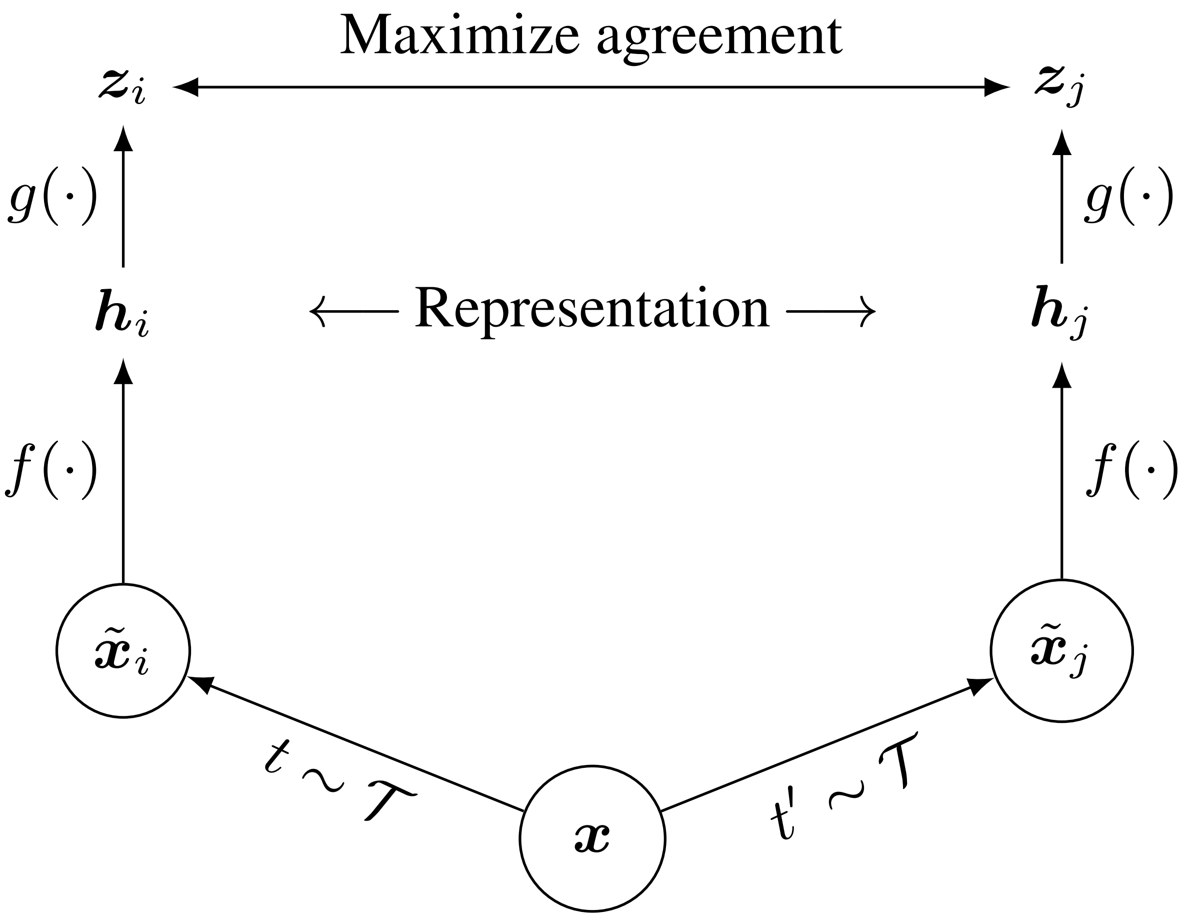
A Simple Framework for Contrastive Learning of Visual Representations

Ting Chen [1] **Simon Kornblith** [1] **Mohammad Norouzi** [1] **Geoffrey Hinton** [1]

Abstract

This paper presents *SimCLR*: a simple framework for contrastive learning of visual representations. We simplify recently proposed contrastive self-supervised learning algorithms without requiring specialized architectures or a memory bank. In order to understand what enables the contrastive prediction tasks to learn useful representations, we systematically study the major components of our framework. We show that (1) composition of data augmentations plays a critical role in defining effective predictive tasks, (2) introducing a learnable nonlinear transformation between the representation and the contrastive loss substantially improves the quality of the learned representations, and (3) contrastive learning benefits from larger batch sizes and more training steps compared to supervised learning. By combining these findings, we are able to considerably outperform previous methods for self-supervised and semi-supervised learning on ImageNet. A linear classifier trained on self-supervised representations learned by SimCLR achieves 76.5% top-1 accuracy, which is a 7% relative improvement over previous state-of- the-art, matching the performance of a supervised ResNet-50. When fine-tuned on only 1% of the labels, we achieve 85.8% top-5 accuracy, outperforming AlexNet with 100× fewer labels.

SimCLR



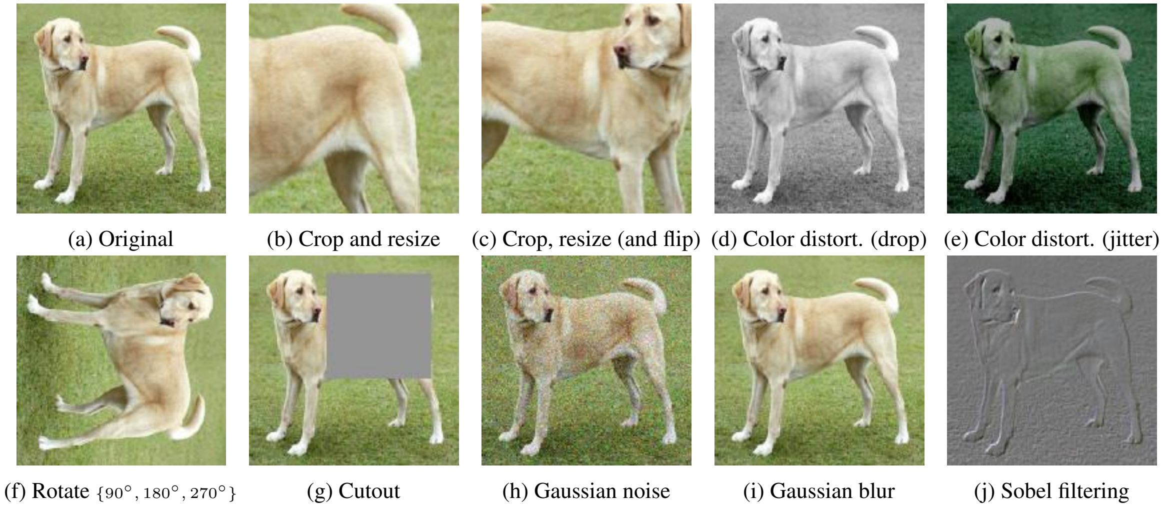


Figure 2. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder 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.

SimCLR

Algorithm 1 SimCLR's main learning algorithm.

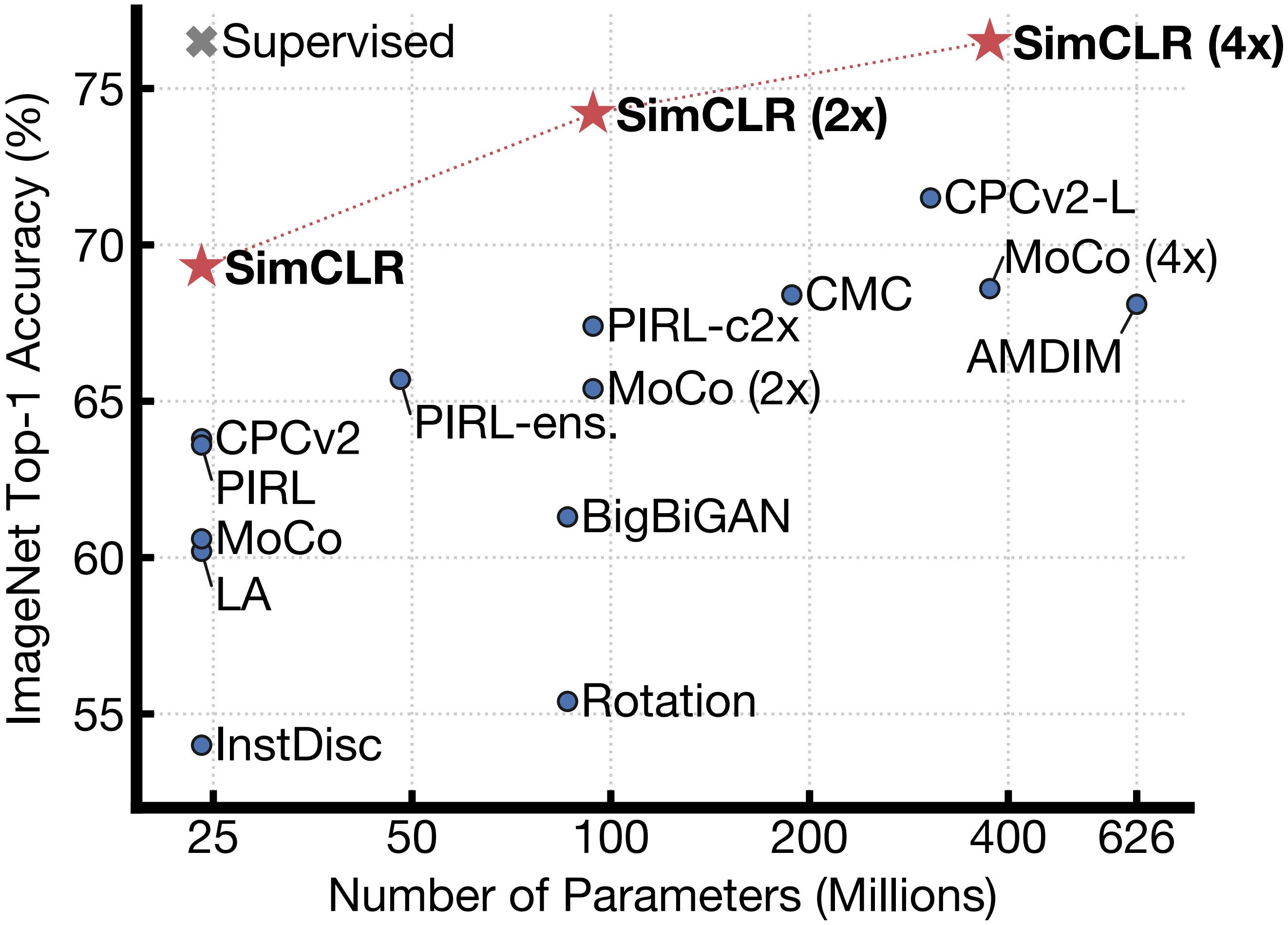
```
input: batch size N, constant τ, structure of f, g, 𝒯.
for sampled minibatch {x_k}_{k=1}^N do
    for all k ∈ {1, ..., N} do
        draw two augmentation functions t∼𝒯, t'∼𝒯
        # the first augmentation
        x̃_{2k−1} = t(x_k)
        h_{2k−1} = f(x̃_{2k−1})                # representation
        z_{2k−1} = g(h_{2k−1})                # projection
        # the second augmentation
        x̃_{2k} = t'(x_k)
        h_{2k} = f(x̃_{2k})                    # representation
        z_{2k} = g(h_{2k})                    # projection
    end for
    for all i ∈ {1, ..., 2N} and j ∈ {1, ..., 2N} do
        s_{i,j} = z_i^⊤ z_j / (‖z_i‖‖z_j‖)    # pairwise similarity
    end for
    define ℓ(i, j) as ℓ(i, j) = −log [exp(s_{i,j}/τ) / Σ_{k=1}^{2N} 𝟙_[k≠i] exp(s_{i,k}/τ)]
    ℒ = (1/2N) Σ_{k=1}^N [ℓ(2k−1, 2k) + ℓ(2k, 2k−1)]
    update networks f and g to minimize ℒ
end for
return encoder network f(·), and throw away g(·)
```

Where:

$$\ell(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp(s_{i,k}/\tau)}$$

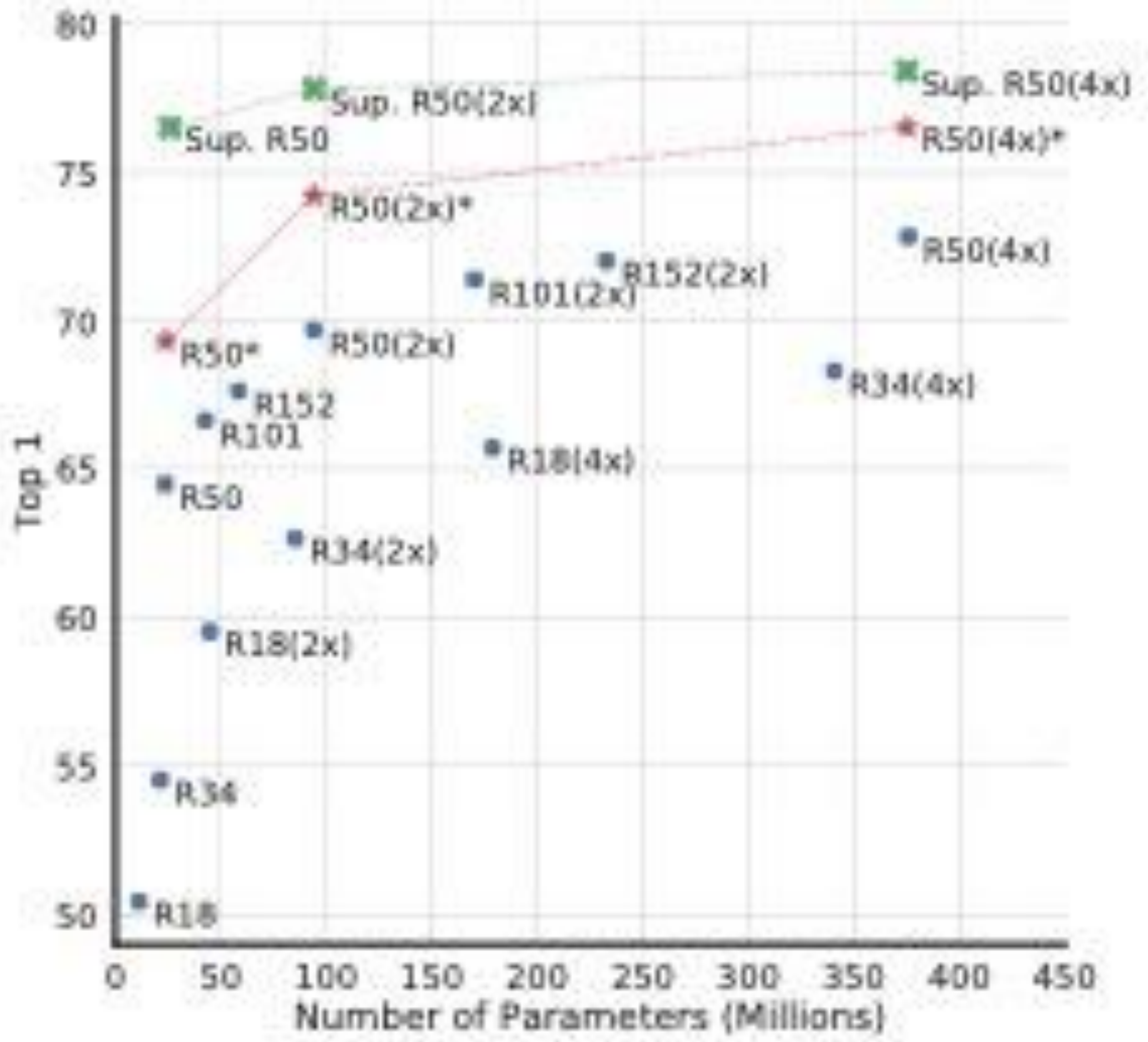
$$\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} \left[\ell(2k-1, 2k) + \ell(2k, 2k-1)\right]$$

SimCLR



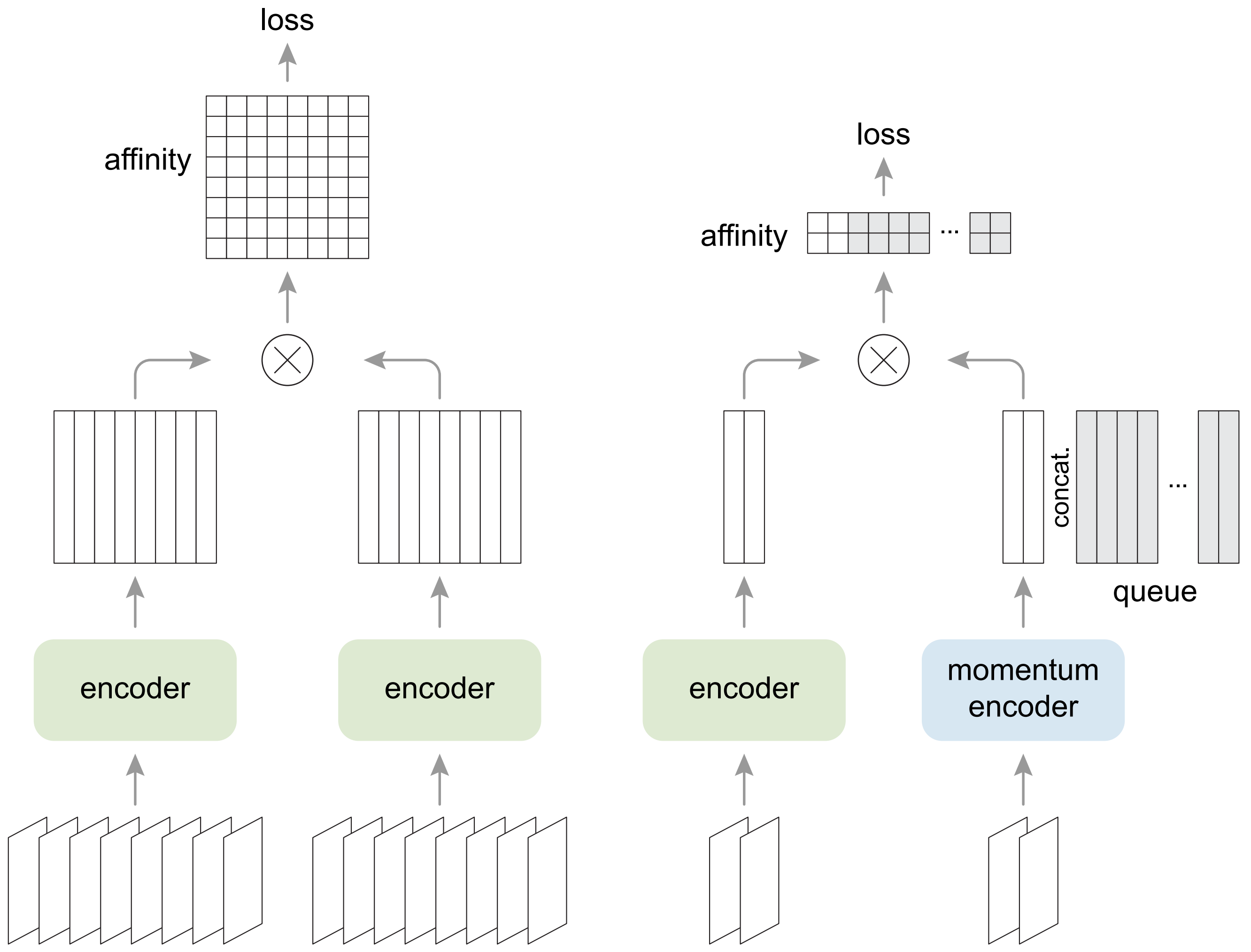
Supervised
SimCLR (4x)
SimCLR (2x)
SimCLR
CPCv2-L
MoCo (4x)
CMC
AMDIM
PIRL-c2x
MoCo (2x)
PIRL-ens.
CPCv2
PIRL
MoCo
LA
BigBiGAN
Rotation
InstDisc
ImageNet Top-1 Accuracy (%)
75
70
65
60
55
25
50
100
200
400
626
Number of Parameters (Millions)

SimCLR



Sup. R50
Sup. R50(2x)
Sup. R50(4x)
R50*
R50(2x)*
R50(4x)*
R18
R34
R50
R101
R152
R18(2x)
R34(2x)
R50(2x)
R101(2x)
R152(2x)
R18(4x)
R34(4x)
R50(4x)
Top 1
80
75
70
65
60
55
50
0
50
100
150
200
250
300
350
400
450
Number of Parameters (Millions)

MoCov2 vs SimCLR



(a) end-to-end (b) Momentum Contrast

MoCov2 vs SimCLR

case	unsup. pre-train MLP	aug+	cos	epochs	ImageNet acc.	VOC detection AP_{50}	AP	AP_{75}
supervised					76.5	81.3	53.5	58.8
MoCo v1				200	60.6	81.5	55.9	62.6
(a)	✓			200	66.2	82.0	56.4	62.6
(b)		✓		200	63.4	82.2	56.8	63.2
(c)	✓	✓		200	67.3	**82.5**	57.2	63.9
(d)	✓	✓	✓	200	67.5	82.4	57.0	63.6
(e)	✓	✓	✓	**800**	**71.1**	**82.5**	**57.4**	**64.0**

MLP: with an MLP head

aug+: with extra blur augmentation

cos: cosine learning rate schedule.

MoCov2 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**

BYOL

Bootstrap Your Own Latent
A New Approach to Self-Supervised Learning

Jean-Bastien Grill[*,1] **Florian Strub**[*,1] **Florent Altché**[*,1] **Corentin Tallec**[*,1] **Pierre H. Richemond**[*,1,2]
Elena Buchatskaya[1] **Carl Doersch**[1] **Bernardo Avila Pires**[1] **Zhaohan Daniel Guo**[1]
Mohammad Gheshlaghi Azar[1] **Bilal Piot**[1] **Koray Kavukcuoglu**[1] **Rémi Munos**[1] **Michal Valko**[1]

[1]DeepMind [2]Imperial College

[jbgrill,fstrub,altche,corentint,richemond]@google.com

Abstract

We introduce **B**ootstrap **Y**our **O**wn **L**atent (BYOL), a new approach to self-supervised image representation learning. BYOL relies on two neural networks, referred to as *online* and *target* networks, that interact and learn from each other. From an augmented view of an image, we train the online network to predict the target network representation of the same image under a different augmented view. At the same time, we update the target network with a slow-moving average of the online network. While state-of-the art methods intrinsically rely on negative pairs, BYOL achieves a new state of the art *without them*. BYOL reaches 74.3% top-1 classification accuracy on ImageNet using the standard linear evaluation protocol with a ResNet-50 architecture and 79.6% with a larger ResNet. We show that BYOL performs on par or better than the current state of the art on both transfer and semi-supervised benchmarks.

BYOL

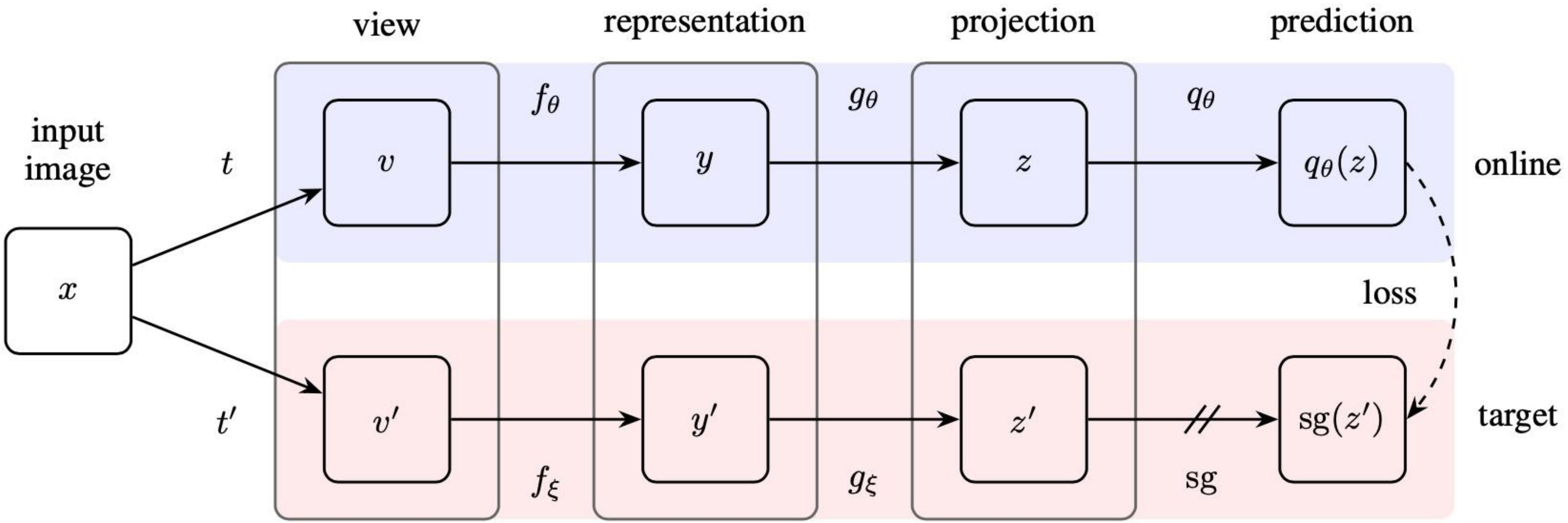
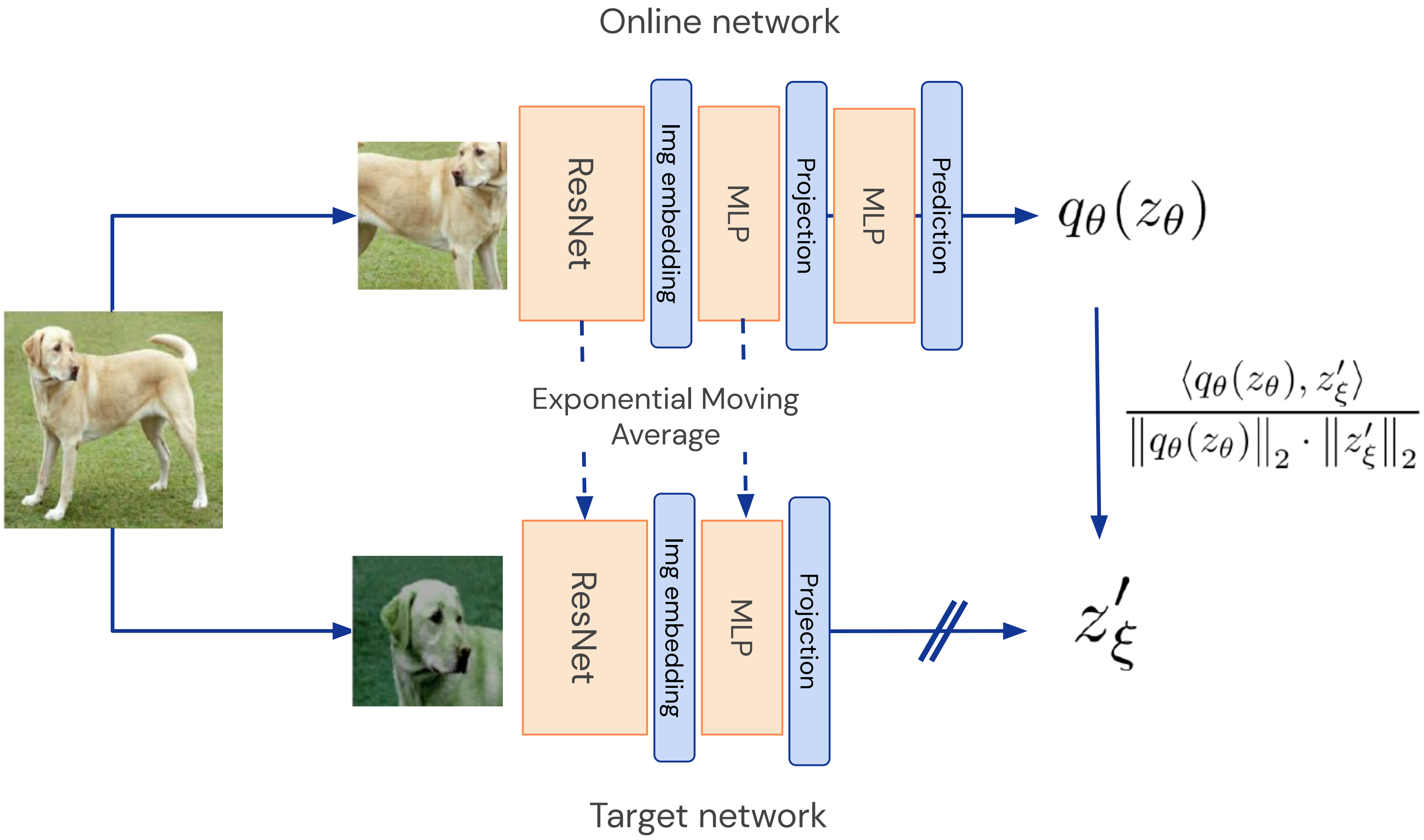


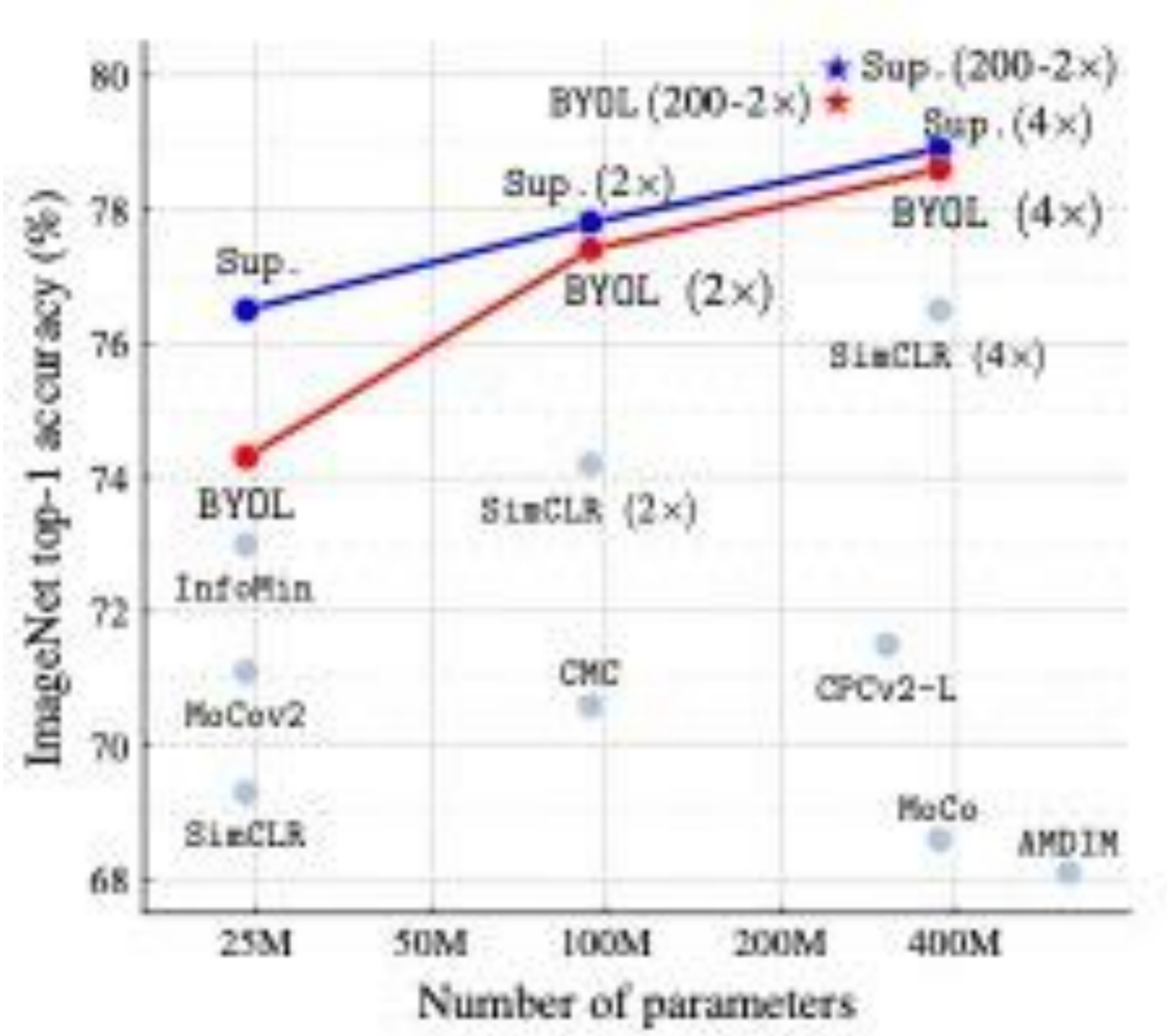
Figure 2: BYOL's architecture. BYOL minimizes a similarity loss between $q_\theta(z)$ and $\mathrm{sg}(z')$, where θ are the trained weights, ξ are an exponential moving average of θ and sg means stop-gradient. At the end of training, everything but f_θ is discarded and y is used as the image representation.

BYOL



- Does not use negative examples!

BYOL



Sup. (200-2×)
BYOL (200-2×)
Sup. (4×)
Sup. (2×)
BYOL (4×)
Sup.
BYOL (2×)
SimCLR (4×)
BYOL
SimCLR (2×)
InfoMin
CMC
CPCv2-L
MoCov2
MoCo
AMDIM
SimCLR
ImageNet top-1 accuracy (%)
80
78
76
74
72
70
68
25M
50M
100M
200M
400M
Number of parameters

BYOL

Method	Top-1	Top-5
Local Agg.	60.2	-
PIRL [35]	63.6	-
CPC v2 [32]	63.8	85.3
CMC [11]	66.2	87.0
SimCLR [8]	69.3	89.0
MoCo v2 [37]	71.1	-
InfoMin Aug. [12]	73.0	91.1
BYOL (ours)	**74.3**	**91.6**

(a) ResNet-50 encoder.

Method	Architecture	Param.	Top-1	Top-5
SimCLR [8]	ResNet-50 (2×)	94M	74.2	92.0
CMC [11]	ResNet-50 (2×)	94M	70.6	89.7
BYOL (ours)	ResNet-50 (2×)	94M	77.4	93.6
CPC v2 [32]	ResNet-161	305M	71.5	90.1
MoCo [9]	ResNet-50 (4×)	375M	68.6	-
SimCLR [8]	ResNet-50 (4×)	375M	76.5	93.2
BYOL (ours)	ResNet-50 (4×)	375M	78.6	94.2
BYOL (ours)	ResNet-200 (2×)	250M	**79.6**	**94.8**

(b) Other ResNet encoder architectures.

Table 1: Top-1 and top-5 accuracies (in %) under linear evaluation on ImageNet.

BYOL

Method	Food101	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
Linear evaluation:												
BYOL (ours)	**75.3**	91.3	**78.4**	**57.2**	**62.2**	**67.8**	60.6	82.5	75.5	90.4	94.2	**96.1**
SimCLR (repro)	72.8	90.5	74.4	42.4	60.6	49.3	49.8	81.4	**75.7**	84.6	89.3	92.6
SimCLR [8]	68.4	90.6	71.6	37.4	58.8	50.3	50.3	80.5	74.5	83.6	90.3	91.2
Supervised-IN [8]	72.3	**93.6**	78.3	53.7	61.9	66.7	**61.0**	**82.8**	74.9	**91.5**	**94.5**	94.7
Fine-tuned:												
BYOL (ours)	**88.5**	**97.8**	86.1	**76.3**	63.7	91.6	**88.1**	**85.4**	**76.2**	91.7	**93.8**	97.0
SimCLR (repro)	87.5	97.4	85.3	75.0	63.9	91.4	87.6	84.5	75.4	89.4	91.7	96.6
SimCLR [8]	88.2	97.7	85.9	75.9	63.5	91.3	88.1	84.1	73.2	89.2	92.1	97.0
Supervised-IN [8]	88.3	97.5	**86.4**	75.8	**64.3**	**92.1**	86.0	85.0	74.6	**92.1**	93.3	**97.6**
Random init [8]	86.9	95.9	80.2	76.1	53.6	91.4	85.9	67.3	64.8	81.5	72.6	92.0

Table 3: Transfer learning results from ImageNet (IN) with the standard ResNet-50 architecture.

DINO

Emerging Properties in Self-Supervised Vision Transformers

Mathilde Caron[1,2] Hugo Touvron[1,3] Ishan Misra[1] Hervé Jegou[1]
Julien Mairal[2] Piotr Bojanowski[1] Armand Joulin[1]

[1] Facebook AI Research [2] Inria* [3] Sorbonne University

Figure 1: **Self-attention from a Vision Transformer with 8×8 patches trained with no supervision.** We look at the self-attention of the `[CLS]` token on the heads of the last layer. This token is not attached to any label nor supervision. These maps show that the model automatically learns class-specific features leading to unsupervised object segmentations.

Abstract

*In this paper, we question if self-supervised learning provides new properties to Vision Transformer (ViT) [16] that stand out compared to convolutional networks (convnets). Beyond the fact that adapting self-supervised methods to this architecture works particularly well, we make the following observations: first, self-supervised ViT features contain explicit information about the semantic segmentation of an image, which does not emerge as clearly with supervised ViTs, nor with convnets. Second, these features are also excellent k-NN classifiers, reaching 78.3% top-1 on ImageNet with a small ViT. Our study also underlines the importance of momentum encoder [26], multi-crop training [9], and the use of small patches with ViTs. We implement our findings into a simple self-supervised method, called DINO, which we interpret as a form of self-**di**stillation with **no** labels. We show the synergy between DINO and ViTs by achieving 80.1% top-1 on ImageNet in linear evaluation with ViT-Base.*

DINO

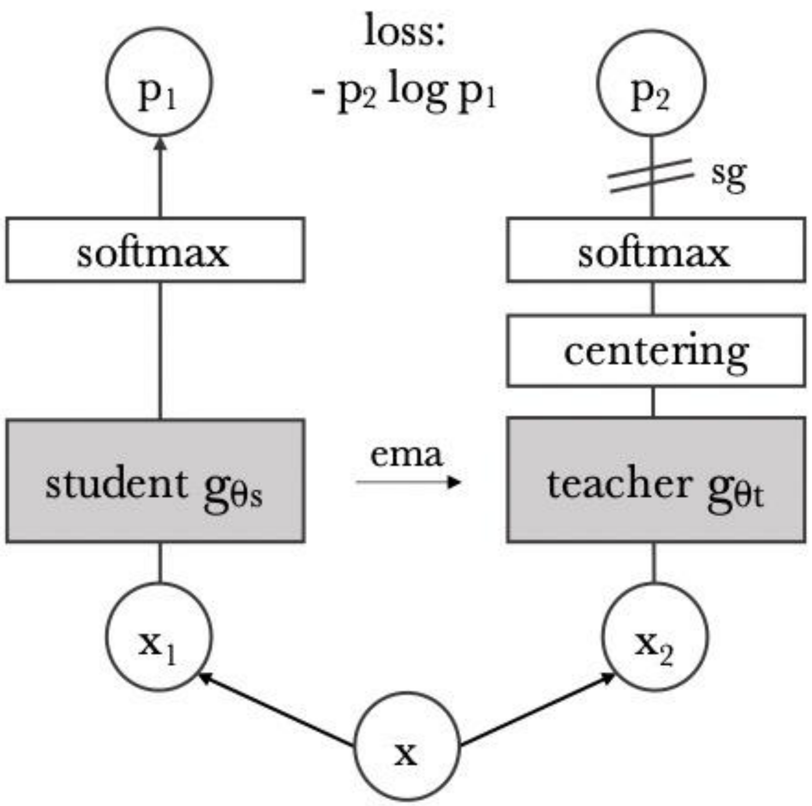


Figure 2: **Self-distillation with no labels.** We illustrate DINO in the case of one single pair of views (x_1, x_2) for simplicity. The model passes two different random transformations of an input image to the student and teacher networks. Both networks have the same architecture but different parameters. The output of the teacher network is centered with a mean computed over the batch. Each networks outputs a K dimensional feature that is normalized with a temperature softmax over the feature dimension. Their similarity is then measured with a cross-entropy loss. We apply a stop-gradient (sg) operator on the teacher to propagate gradients only through the student. The teacher parameters are updated with an exponential moving average (ema) of the student parameters.

DINO

Algorithm 1 DINO PyTorch pseudocode w/o multi-crop.

```
# gs, gt: student and teacher networks
# C: center (K)
# tps, tpt: student and teacher temperatures
# l, m: network and center momentum rates
gt.params = gs.params
for x in loader: # load a minibatch x with n samples
    x1, x2 = augment(x), augment(x) # random views

    s1, s2 = gs(x1), gs(x2) # student output n-by-K
    t1, t2 = gt(x1), gt(x2) # teacher output n-by-K

    loss = H(t1, s2)/2 + H(t2, s1)/2
    loss.backward() # back-propagate

    # student, teacher and center updates
    update(gs) # SGD
    gt.params = l*gt.params + (1-l)*gs.params
    C = m*C + (1-m)*cat([t1, t2]).mean(dim=0)

def H(t, s):
    t = t.detach() # stop gradient
    s = softmax(s / tps, dim=1)
    t = softmax((t - C) / tpt, dim=1) # center + sharpen
    return - (t * log(s)).sum(dim=1).mean()
```

DINO

Table 2: **Linear and k-NN classification on ImageNet.** We report top-1 accuracy for linear and k-NN evaluations on the validation set of ImageNet for different self-supervised methods. We focus on ResNet-50 and ViT-small architectures, but also report the best results obtained across architectures. * are run by us. We run the k-NN evaluation for models with official released weights. The throughput (im/s) is calculated on a NVIDIA V100 GPU with 128 samples per forward. Parameters (M) are of the feature extractor.

Method	Arch.	Param.	im/s	Linear	k-NN
Supervised	RN50	23	1237	79.3	79.3
SCLR [11]	RN50	23	1237	69.1	60.7
MoCov2 [13]	RN50	23	1237	71.1	61.9
InfoMin [54]	RN50	23	1237	73.0	65.3
BarlowT [66]	RN50	23	1237	73.2	66.0
OBoW [21]	RN50	23	1237	73.8	61.9
BYOL [23]	RN50	23	1237	74.4	64.8
DCv2 [9]	RN50	23	1237	75.2	67.1
SwAV [9]	RN50	23	1237	**75.3**	65.7
DINO	RN50	23	1237	**75.3**	**67.5**
Supervised	ViT-S	21	1007	79.8	79.8
BYOL* [23]	ViT-S	21	1007	71.4	66.6
MoCov2* [13]	ViT-S	21	1007	72.7	64.4
SwAV* [9]	ViT-S	21	1007	73.5	66.3
DINO	ViT-S	21	1007	**77.0**	**74.5**
Comparison across architectures					
SCLR [11]	RN50w4	375	117	76.8	69.3
SwAV [9]	RN50w2	93	384	77.3	67.3
BYOL [23]	RN50w2	93	384	77.4	–
DINO	ViT-B/16	85	312	78.2	76.1
SwAV [9]	RN50w5	586	76	78.5	67.1
BYOL [23]	RN50w4	375	117	78.6	–
BYOL [23]	RN200w2	250	123	79.6	73.9
DINO	ViT-S/8	21	180	79.7	**78.3**
SCLRv2 [12]	RN152w3+SK	794	46	79.8	73.1
DINO	ViT-B/8	85	63	**80.1**	77.4

DINO

	Random	Supervised	DINO
ViT-S/16	22.0	27.3	45.9
ViT-S/8	21.8	23.7	44.7

Figure 4: **Segmentations from supervised versus DINO.** We visualize masks obtained by thresholding the self-attention maps to keep 60% of the mass. On top, we show the resulting masks for a ViT-S/8 trained with supervision and DINO. We show the best head for both models. The table at the bottom compares the Jaccard similarity between the ground truth and these masks on the validation images of PASCAL VOC12 dataset.

Barlow Twins

Barlow Twins: Self-Supervised Learning via Redundancy Reduction

Jure Zbontar [*1] **Li Jing** [*1] **Ishan Misra** [1] **Yann LeCun** [1,2] **Stéphane Deny** [1]

Self-supervised learning (SSL) is rapidly closing the gap with supervised methods on large computer vision benchmarks. A successful approach to SSL is to learn embeddings which are invariant to distortions of the input sample. However, a recurring issue with this approach is the existence of trivial constant solutions. Most current methods avoid such solutions by careful implementation details. We propose an objective function that naturally avoids collapse by measuring the cross-correlation matrix between the outputs of two identical networks fed with distorted versions of a sample, and making it as close to the identity matrix as possible. This causes the embedding vectors of distorted versions of a sample to be similar, while minimizing the redundancy between the components of these vectors. The method is called BARLOW TWINS, owing to neuroscientist H. Barlow's *redundancy-reduction principle* applied to a pair of identical networks. BARLOW TWINS does not require large batches nor asymmetry between the network twins such as a predictor network, gradient stopping, or a moving average on the weight updates. Intriguingly it benefits from very high-dimensional output vectors. BARLOW TWINS outperforms previous methods on ImageNet for semi-supervised classification in the low-data regime, and is on par with current state of the art for ImageNet classification with a linear classifier head, and for transfer tasks of classification and object detection.

Barlow Twins

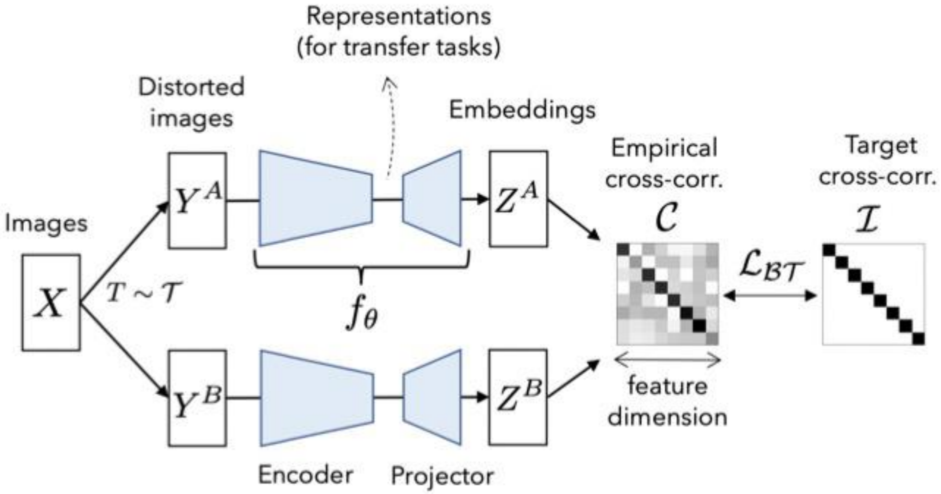


Figure 1. BARLOW TWINS's objective function measures the cross-correlation matrix between the embeddings of two identical networks fed with distorted versions of a batch of samples, and tries to make this matrix close to the identity. This causes the embedding vectors of distorted versions of a sample to be similar, while minimizing the redundancy between the components of these vectors. BARLOW TWINS is competitive with state-of-the-art methods for self-supervised learning while being conceptually simpler, naturally avoiding trivial constant (i.e. collapsed) embeddings, and being robust to the training batch size.

Barlow Twins

Algorithm 1 PyTorch-style pseudocode for Barlow Twins.

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

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

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

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

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

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

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

Barlow Twins

Table 1. **Top-1 and top-5 accuracies (in %) under linear evaluation on ImageNet**. All models use a ResNet-50 encoder. Top-3 best self-supervised methods are underlined.

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

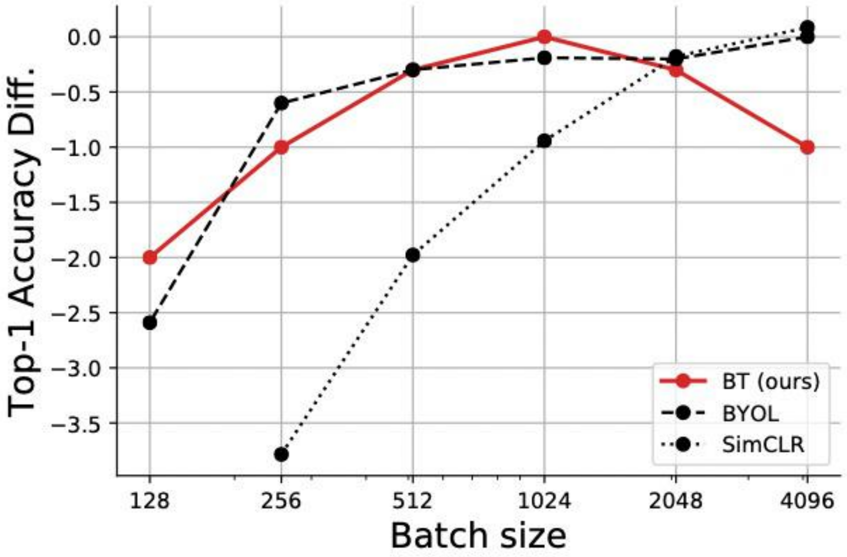


Figure 2. Effect of batch size. To compare the effect of the batch size across methods, for each method we report the difference between the top-1 accuracy at a given batch size and the best obtained accuracy among all batch size tested. BYOL: best accuracy is 72.5% for a batch size of 4096 (data from (Grill et al., 2020) fig. 3A). SimCLR: best accuracy is 67.1% for a batch size of 4096 (data from (Chen et al., 2020a) fig. 9, model trained for 300 epochs). Barlow Twins: best accuracy is 71.7% for a batch size of 1024.

CLIP Contrastive Language-Image Pre-training

Learning Transferable Visual Models From Natural Language Supervision

Alec Radford [*1] Jong Wook Kim [*1] Chris Hallacy [1] Aditya Ramesh [1] Gabriel Goh [1] Sandhini Agarwal [1]
Girish Sastry [1] Amanda Askell [1] Pamela Mishkin [1] Jack Clark [1] Gretchen Krueger [1] Ilya Sutskever [1]

Abstract

State-of-the-art computer vision systems are trained to predict a fixed set of predetermined object categories. This restricted form of supervision limits their generality and usability since additional labeled data is needed to specify any other visual concept. Learning directly from raw text about images is a promising alternative which leverages a much broader source of supervision. We demonstrate that the simple pre-training task of predicting which caption goes with which image is an efficient and scalable way to learn SOTA image representations from scratch on a dataset of 400 million (image, text) pairs collected from the internet. After pre-training, natural language is used to reference learned visual concepts (or describe new ones) enabling zero-shot transfer of the model to downstream tasks. We study the performance of this approach by benchmarking on over 30 different existing computer vision datasets, spanning tasks such as OCR, action recognition in videos, geo-localization, and many types of fine-grained object classification. The model transfers non-trivially to most tasks and is often competitive with a fully supervised baseline without the need for any dataset specific training. For instance, we match the ac- curacy of the original ResNet-50 on ImageNet zero-shot without needing to use any of the 1.28 million training examples it was trained on.

CLIP

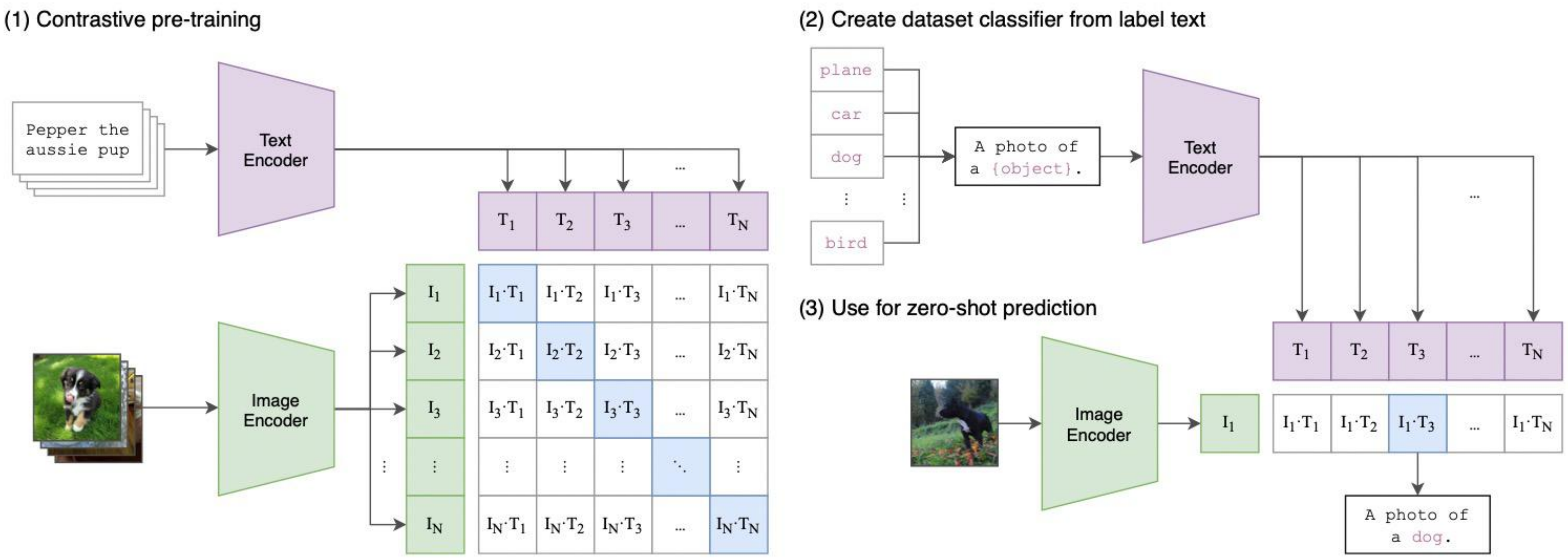


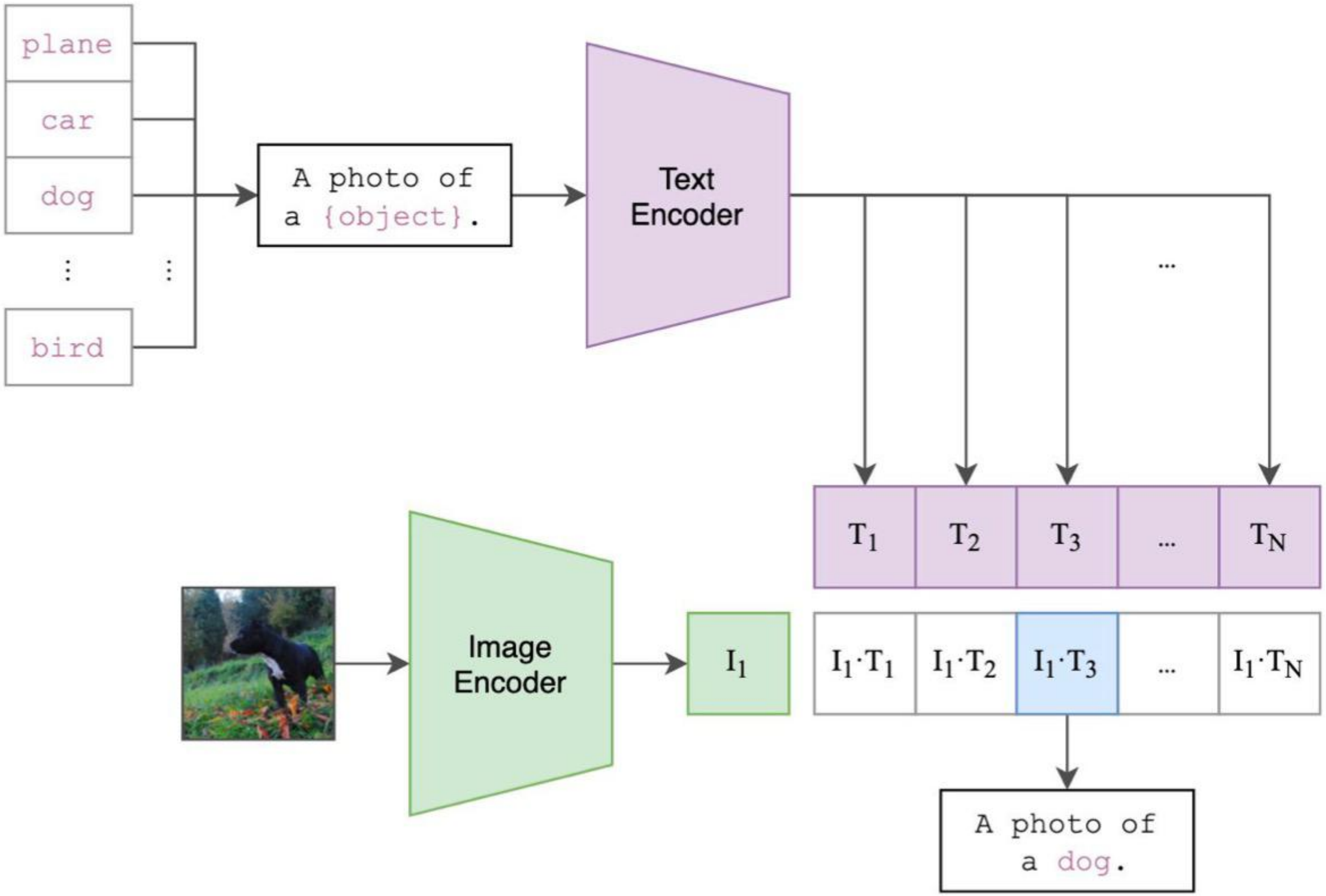
Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

CLIP

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

CLIP



plane
car
dog
bird
A photo of a {object}.
Text Encoder
T1
T2
T3
...
TN
Image Encoder
I1
I1·T1
I1·T2
I1·T3
...
I1·TN
A photo of a dog.

CLIP

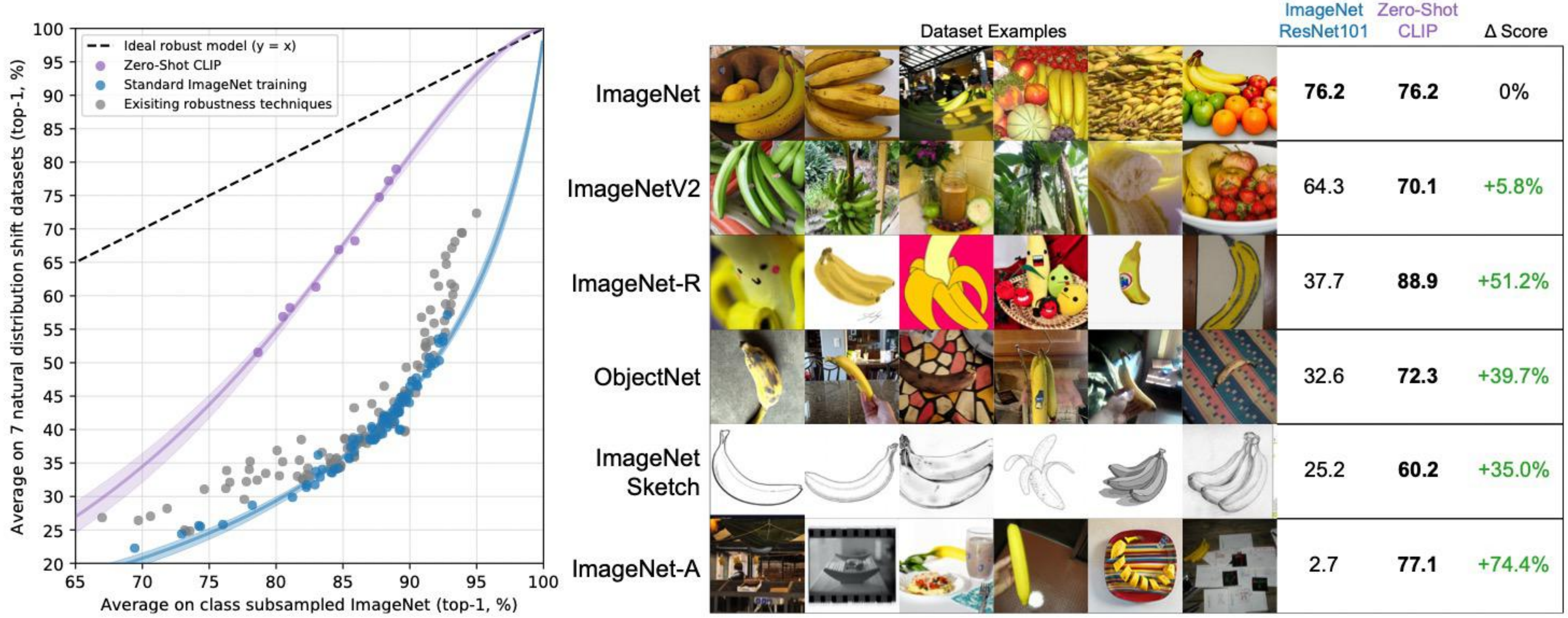
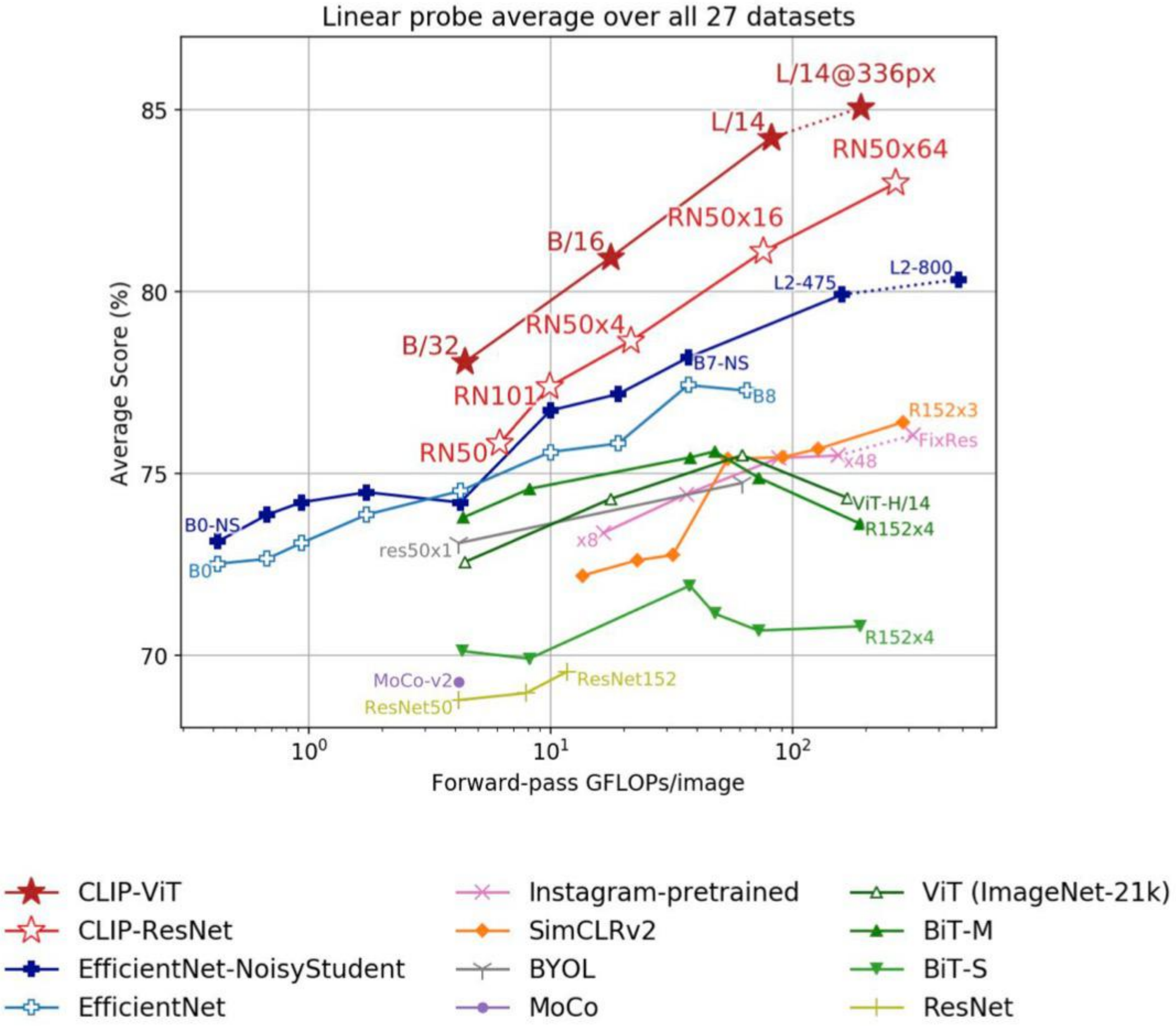


Figure 13. **Zero-shot CLIP is much more robust to distribution shift than standard ImageNet models.** (Left) An ideal robust model (dashed line) performs equally well on the ImageNet distribution and on other natural image distributions. Zero-shot CLIP models shrink this "robustness gap" by up to 75%. Linear fits on logit transformed values are shown with bootstrap estimated 95% confidence intervals. (Right) Visualizing distribution shift for bananas, a class shared across 5 of the 7 natural distribution shift datasets. The performance of the best zero-shot CLIP model, ViT-L/14@336px, is compared with a model that has the same performance on the ImageNet validation set, ResNet-101.

CLIP



Linear probe average over all 27 datasets
L/14@336px
L/14
RN50x64
RN50x16
B/16
L2-475
L2-800
B/32
RN50x4
B7-NS
B8
RN101
RN50
R152x3
FixRes
x48
ViT-H/14
R152x4
B0-NS
res50x1
x8
B0
R152x4
MoCo-v2
ResNet152
ResNet50
Average Score (%)
Forward-pass GFLOPs/image
CLIP-ViT
CLIP-ResNet
EfficientNet-NoisyStudent
EfficientNet
Instagram-pretrained
SimCLRv2
BYOL
MoCo
ViT (ImageNet-21k)
BiT-M
BiT-S
ResNet

Next lecture:
Pretraining Language Models