Self-Supervised Representation Learning

Weiwen Chen

https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html

Outline

- Self-Supervised Learning
- Images-Based
- Video-Based
- Control-Based

https://lilianweng.github.io/lil-log/2019/11/10/self-supervised-learning.html

Self-Supervised Learning

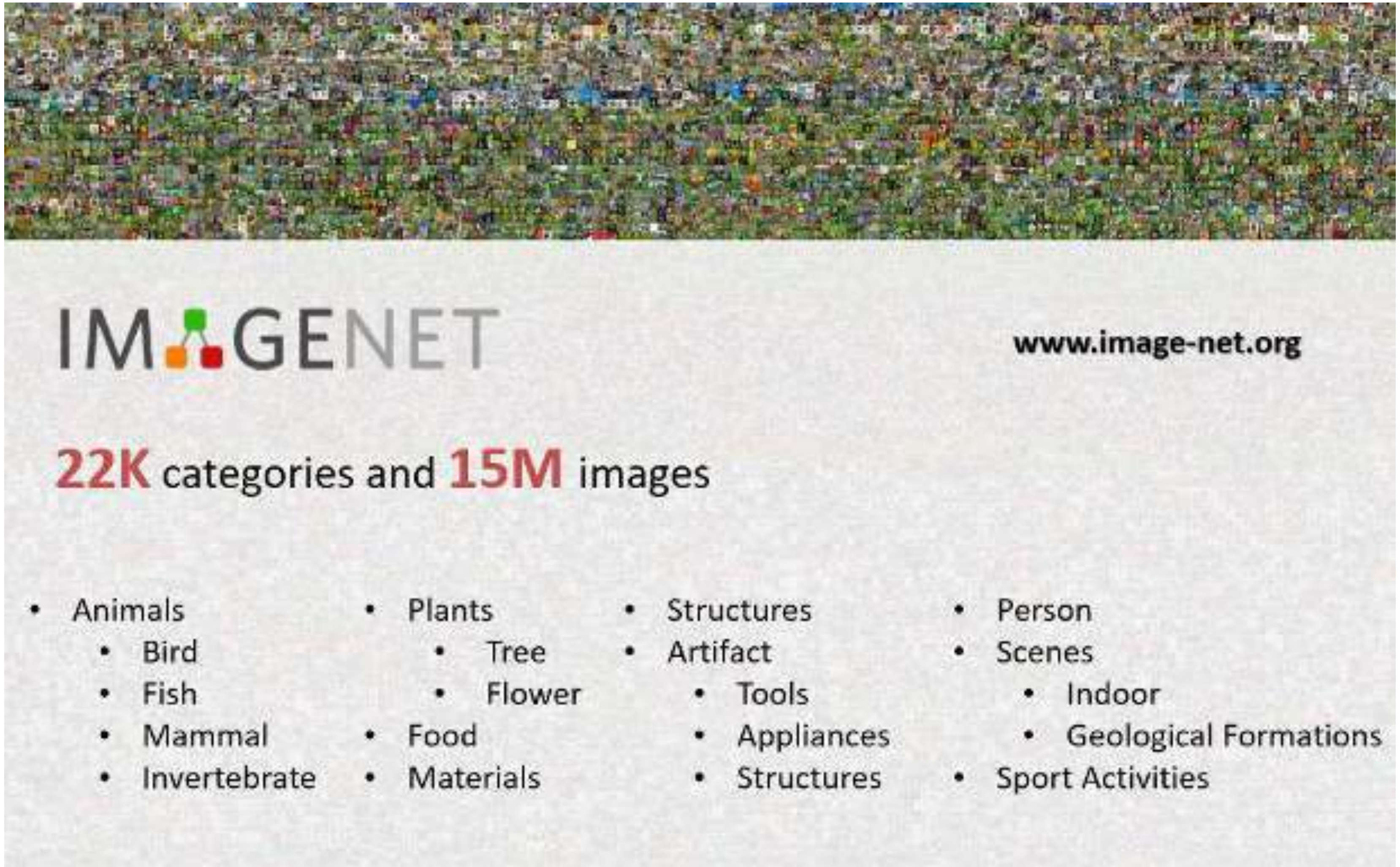
Why

Supervised Learning

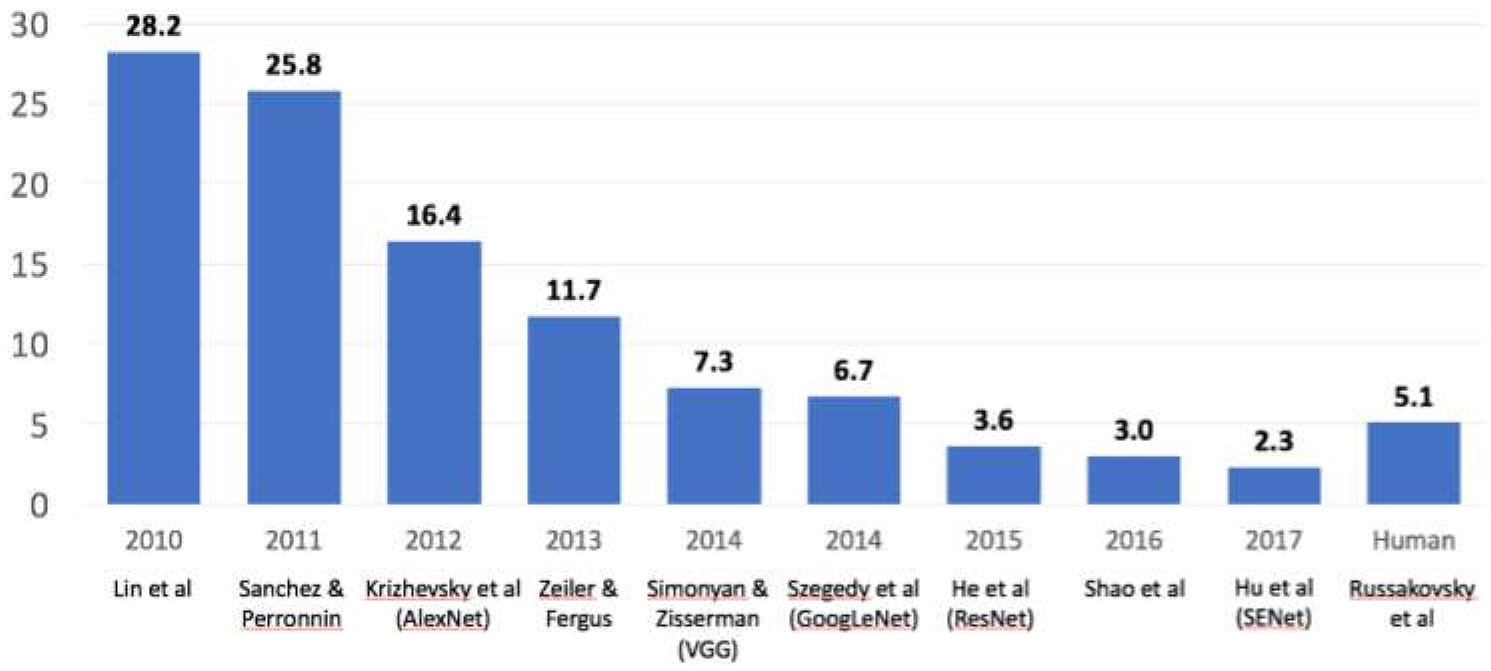
```
def train(images, labels):
  # Machine learning!
  return model
```

```
def predict(model, test_images):
  # Use model to predict labels
  return test_labels
```

Supervised Learning



Supervised Learning

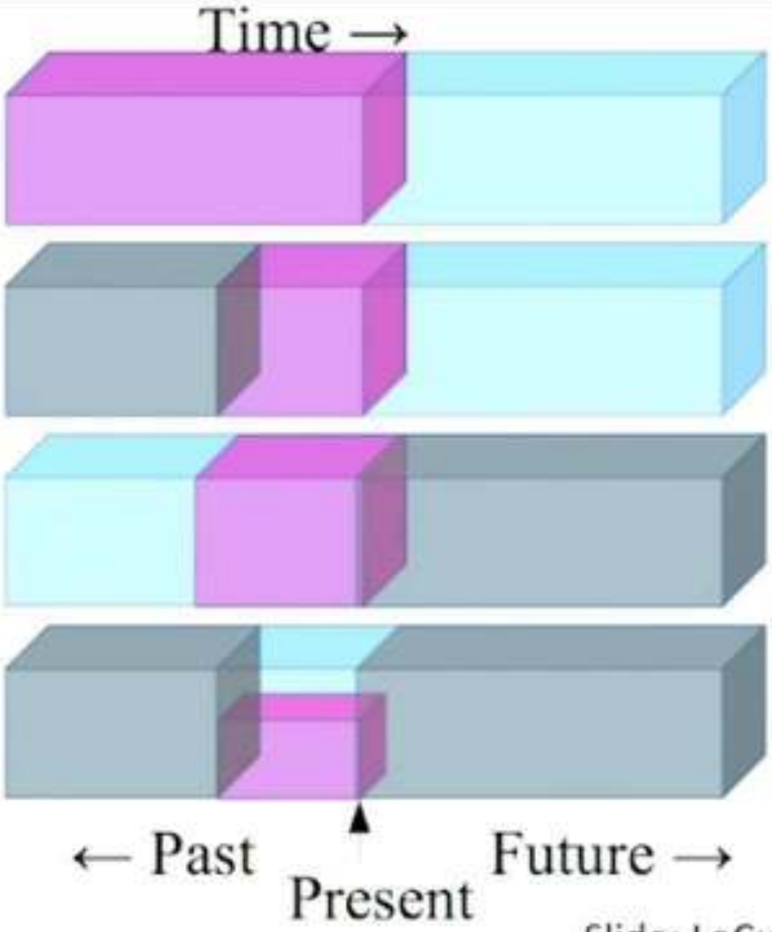


Label: Costly

12306

Language Model

- 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.



Slide: LeCun

Yann LeCun @EPFL - "Self-supervised learning: could machines learn like humans?"
https://www.youtube.com/watch?v=7I0Qt7GALVk

A List

Table of Contents

- Computer Vision (CV)
  - Survey
  - Image Representation Learning
  - Video Representation Learning
  - Geometry
  - Audio
  - Others
- Machine Learning
  - Reinforcement Learning
- Robotics
- Natural Language Processing (NLP)
- Talks
- Thesis

https://github.com/jason718/awesome-self-supervised-learning

Motivation

- Producing a dataset with clean labels is expensive
- but unlabeled data is being generated all the time

Self-supervised Task

- **don’t care** about the final performance
- interested in the learned intermediate **representation**
- representation can carry good semantic or structural **meanings**
- and can be beneficial to a variety of practical **downstream tasks**

Self-Supervised Learning

Images-Based

Video-Based

Control-Based

Images-Based

Distortion

Patches

Colorization

Generative Modeling

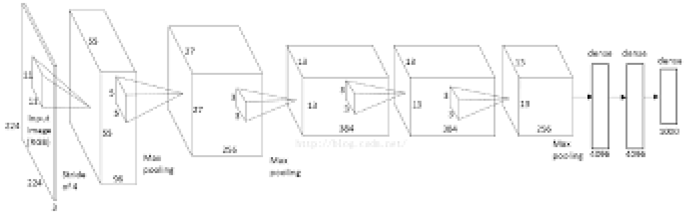


Figure 1. Example of Deep Convolutional Neural Network for Image Classification. Image source: [1].

Images-Based

- Distortion
- Patches
- Colorization
- Generative Modeling

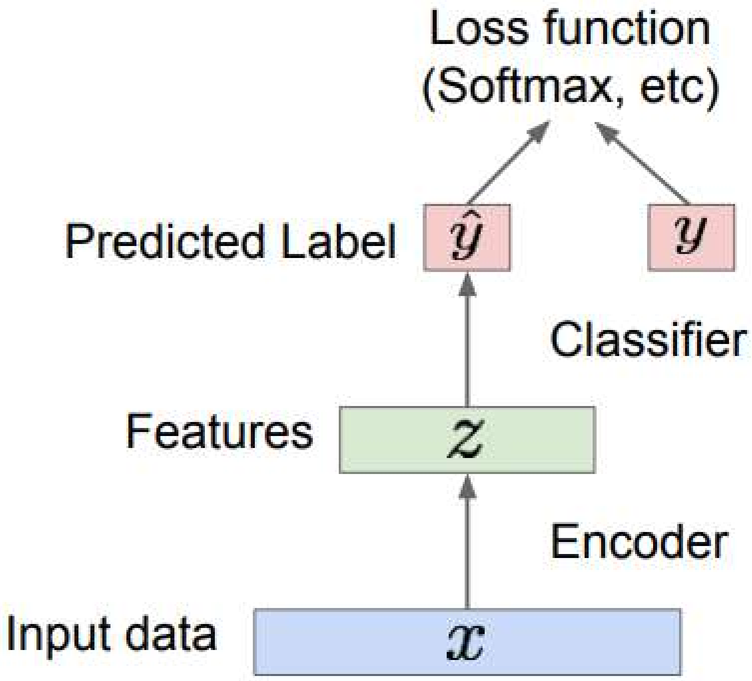


Image-based

Distortion

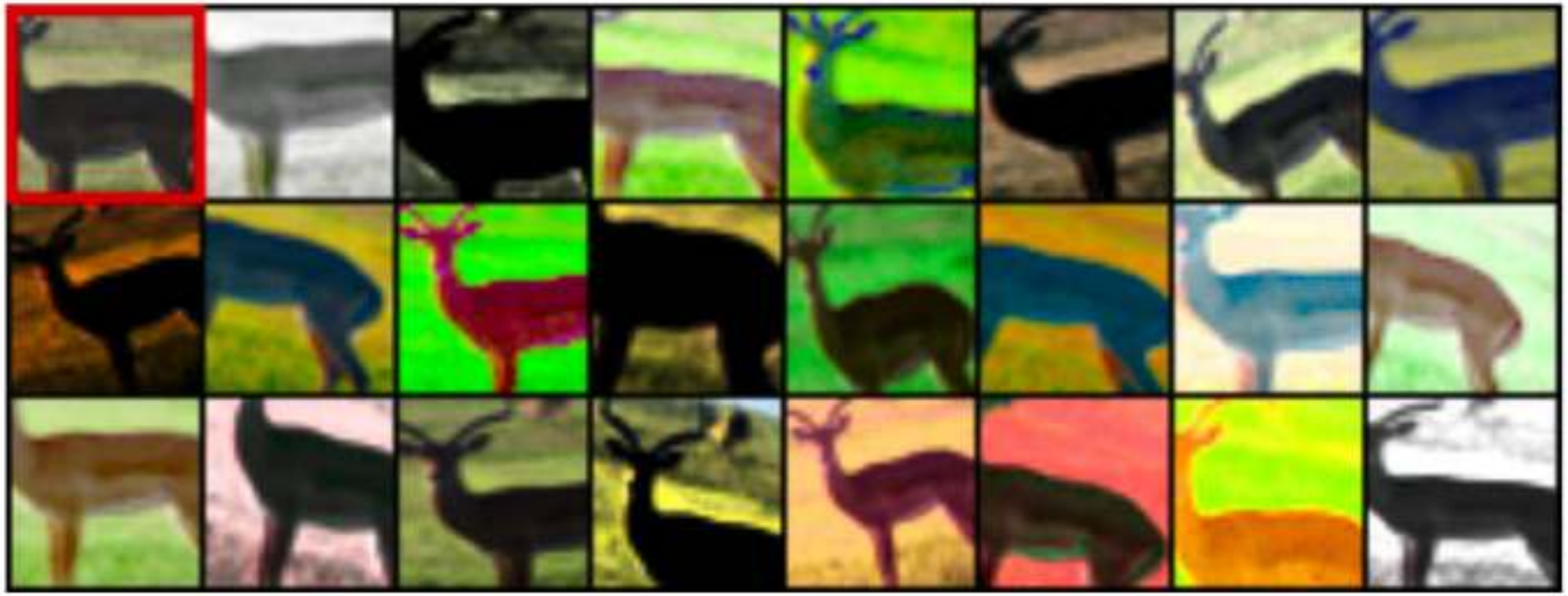
Exemplar-CNN
Rotation

Distortion

- does not modify its original semantic meaning
- or geometric forms

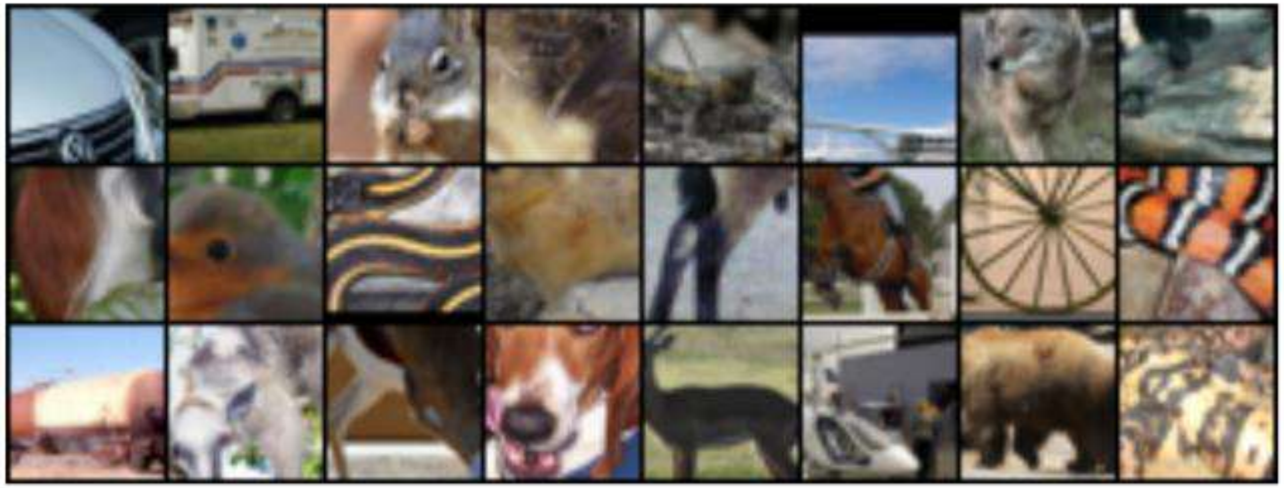
- considered the same as original
- the learned features are expected to be **invariant** to distortion

Exemplar-CNN



Alexey Dosovitskiy, et al. “Discriminative unsupervised feature learning with exemplar convolutional neural networks.” IEEE transactions on pattern analysis and machine intelligence 38.9 (2015): 1734-1747.

"exemplary" patches

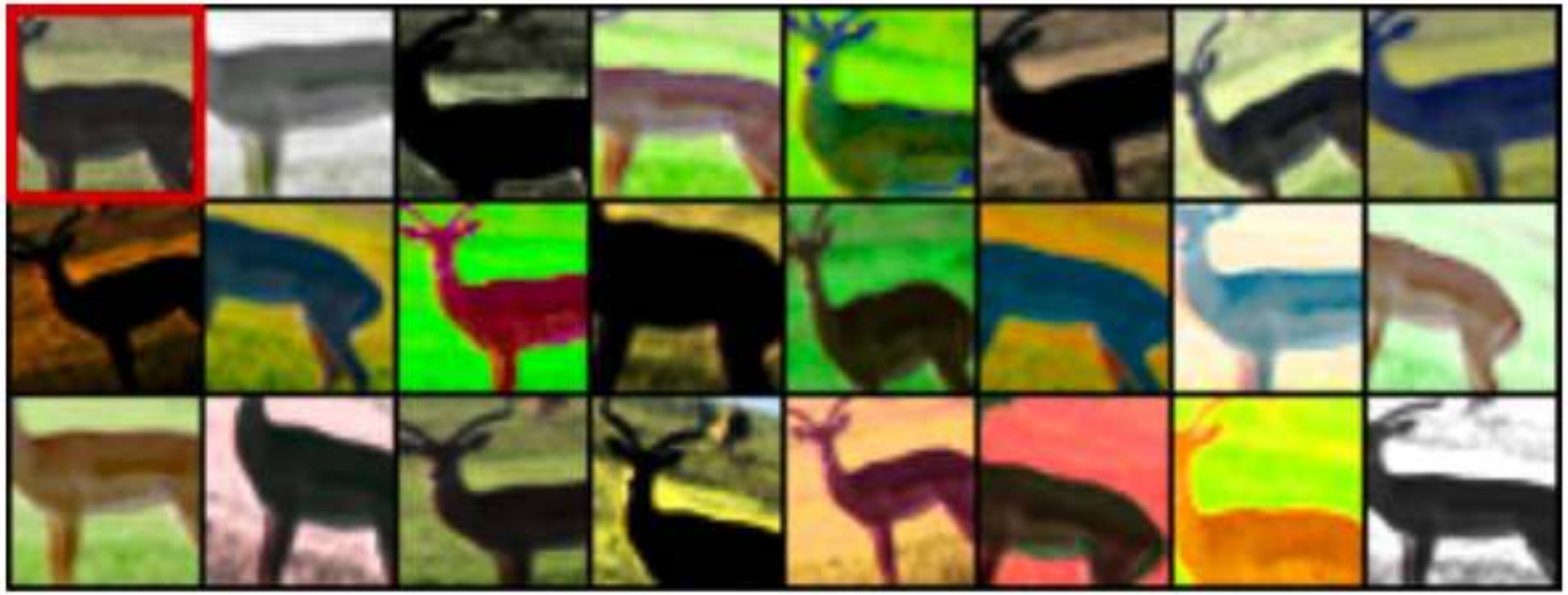


- Sample N patches of size 32 × 32 pixels from **different images** at varying positions and scales,
- only from regions containing considerable gradients as those areas cover **edges**
- and tend to contain **objects** or parts of objects.

Alexey Dosovitskiy, et al. "Discriminative unsupervised feature learning with exemplar convolutional neural networks." IEEE transactions on pattern analysis and machine intelligence 38.9 (2015): 1734-1747.

Exemplar-CNN

- Each patch is distorted by applying a variety of random transformations (i.e., translation, rotation, scaling, etc.).
- All the resulting distorted patches are considered to belong to the **same** surrogate class.



Alexey Dosovitskiy, et al. “Discriminative unsupervised feature learning with exemplar convolutional neural networks.” IEEE transactions on pattern analysis and machine intelligence 38.9 (2015): 1734-1747.

Exemplar-CNN

- The pretext task is to **discriminate** between a set of surrogate classes.
- We can arbitrarily **create** as many surrogate classes as we want.

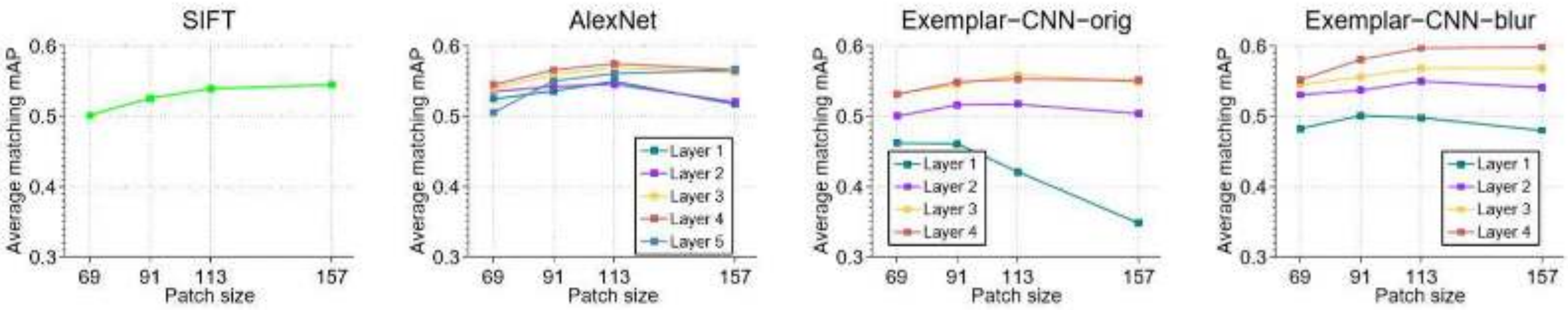


Fig. 8. Analysis of the matching performance depending on the patch size and the network layer at which features are computed.

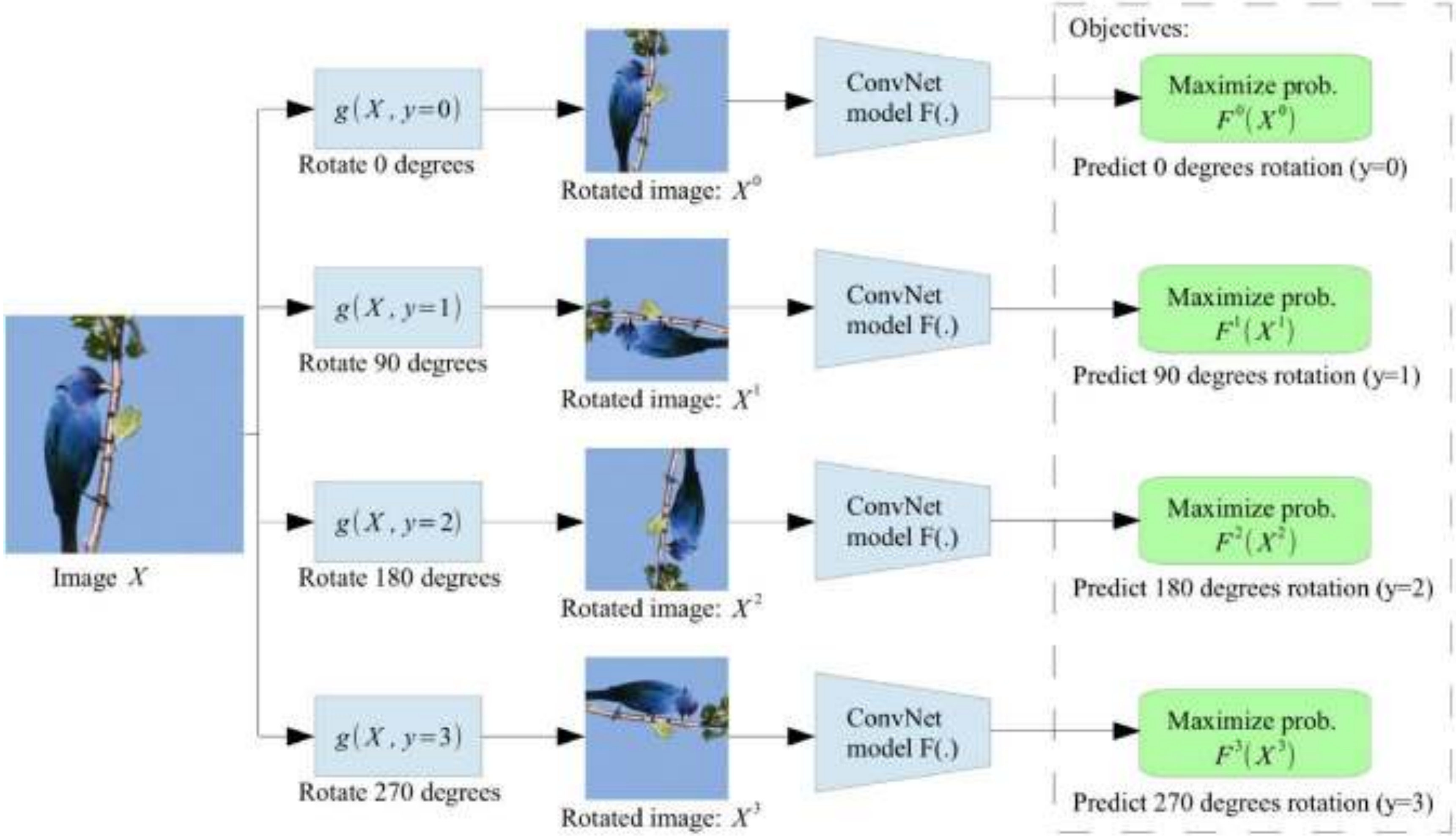
Alexey Dosovitskiy, et al. "Discriminative unsupervised feature learning with exemplar convolutional neural networks." IEEE transactions on pattern analysis and machine intelligence 38.9 (2015): 1734-1747.

Rotation

- Each input image is first rotated by a multiple of 90° at random, corresponding to **[0°,90°,180°,270°].**
- The model is trained to **predict which rotation** has been applied, thus a 4-class classification problem.
- the model has to learn to **recognize high level object parts**, such as heads, noses, and eyes, and the **relative positions** of these parts, rather than local patterns.

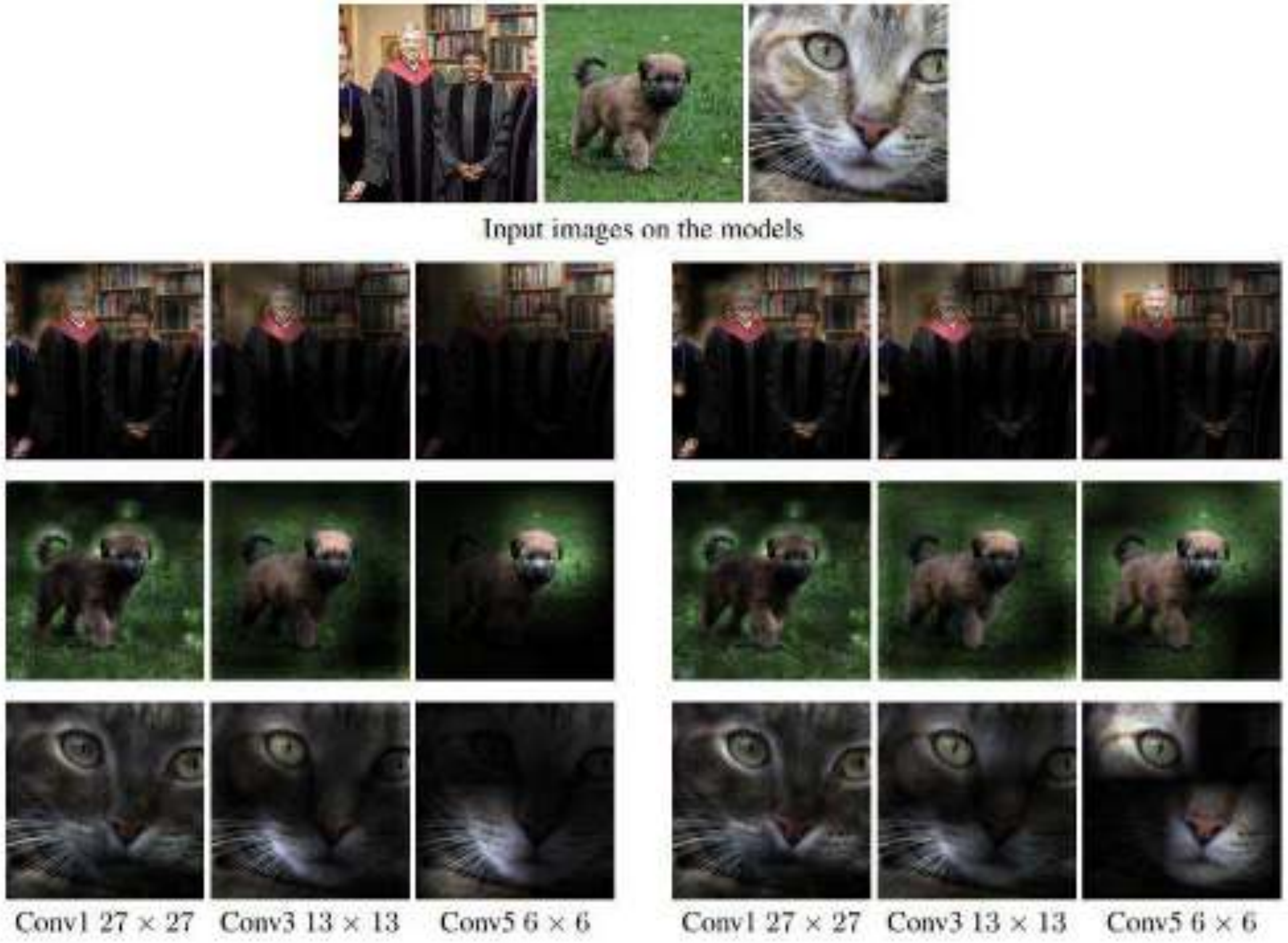
Spyros Gidaris, Praveer Singh & Nikos Komodakis. “Unsupervised Representation Learning by Predicting Image Rotations” ICLR 2018.

Rotation - Framwork



Spyros Gidaris, Praveer Singh & Nikos Komodakis. “Unsupervised Representation Learning by Predicting Image Rotations” ICLR 2018.

Rotation - Analysis



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

Spyros Gidaris, Praveer Singh & Nikos Komodakis. "Unsupervised Representation Learning by Predicting Image Rotations" ICLR 2018.

Image-based

Patches

relative position
jigsaw puzzle
counting features

Patches

多智体人工智能实验室
Multi-Agent Artificial Intelligence Laboratory

- extract multiple patches from one image
- and ask the model to **predict the relationship** between these patches.

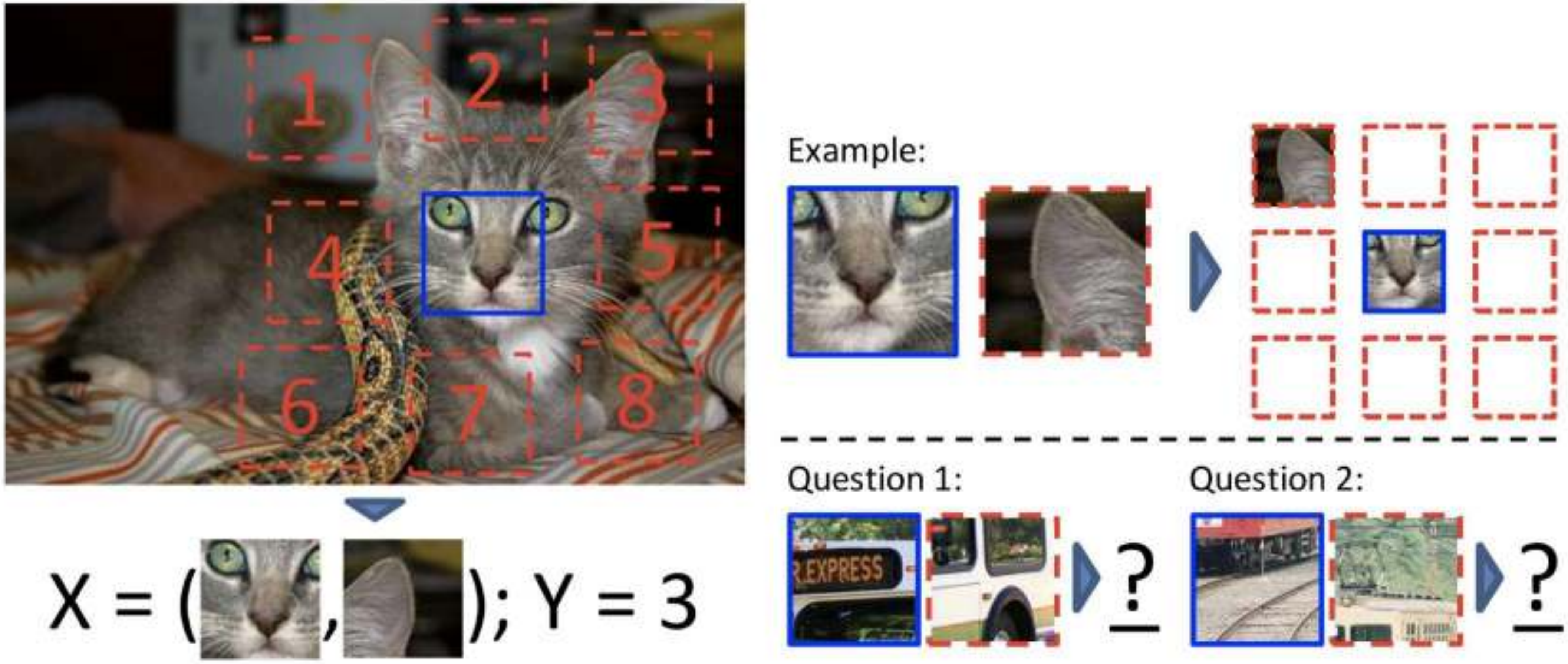
relative position

- predicting the **relative position** between two random patches from one image.

- A model needs to understand the spatial context of objects in order to tell the relative position between parts.

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

relative position - training



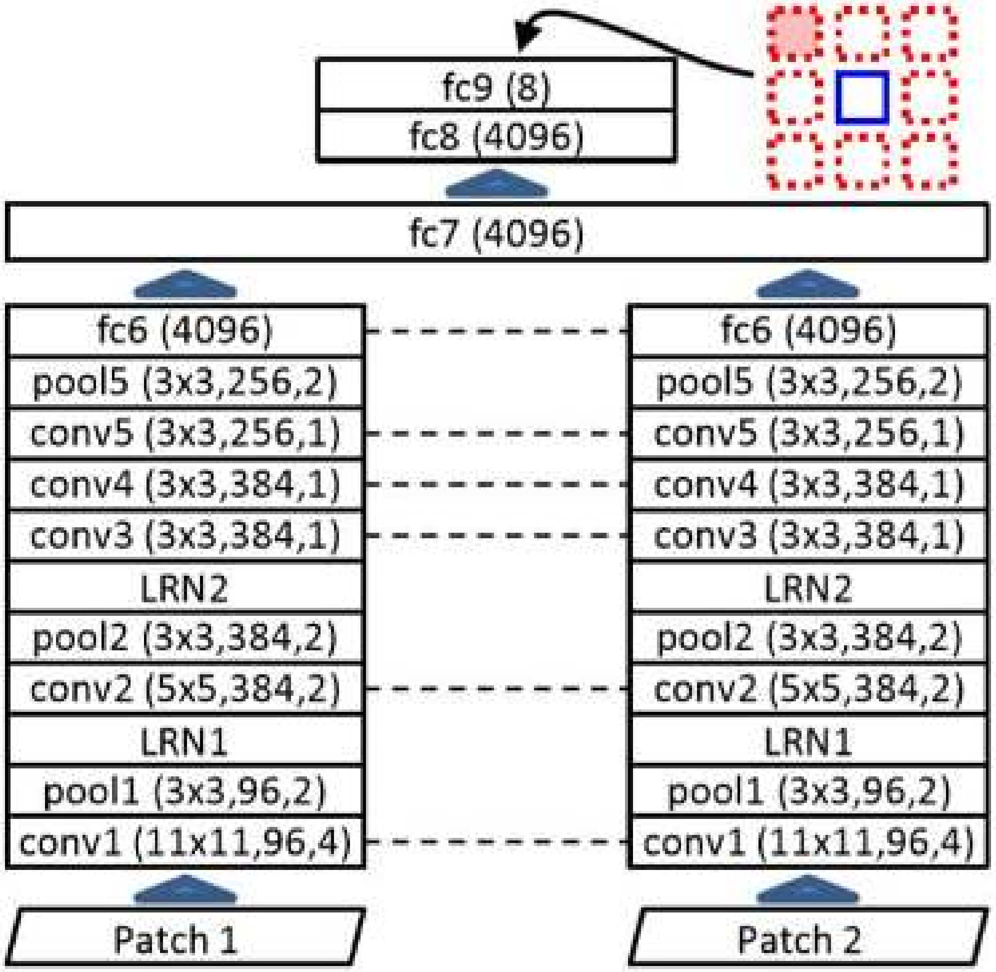
Carl Doersch, Abhinav Gupta, and Alexei A. Efros. “Unsupervised visual representation learning by context prediction.” ICCV. 2015.

relative position - Tricks

- To avoid the model only catching low-level trivial signals, such as connecting a straight line across boundary or matching local patterns, additional noise is introduced by:
  - Add gaps between patches
  - Small jitters
  - Randomly downsample some patches to as little as 100 total pixels, and then upsampling it, to build robustness to pixelation.
  - Shift green and magenta toward gray or randomly drop 2 of 3 color channels

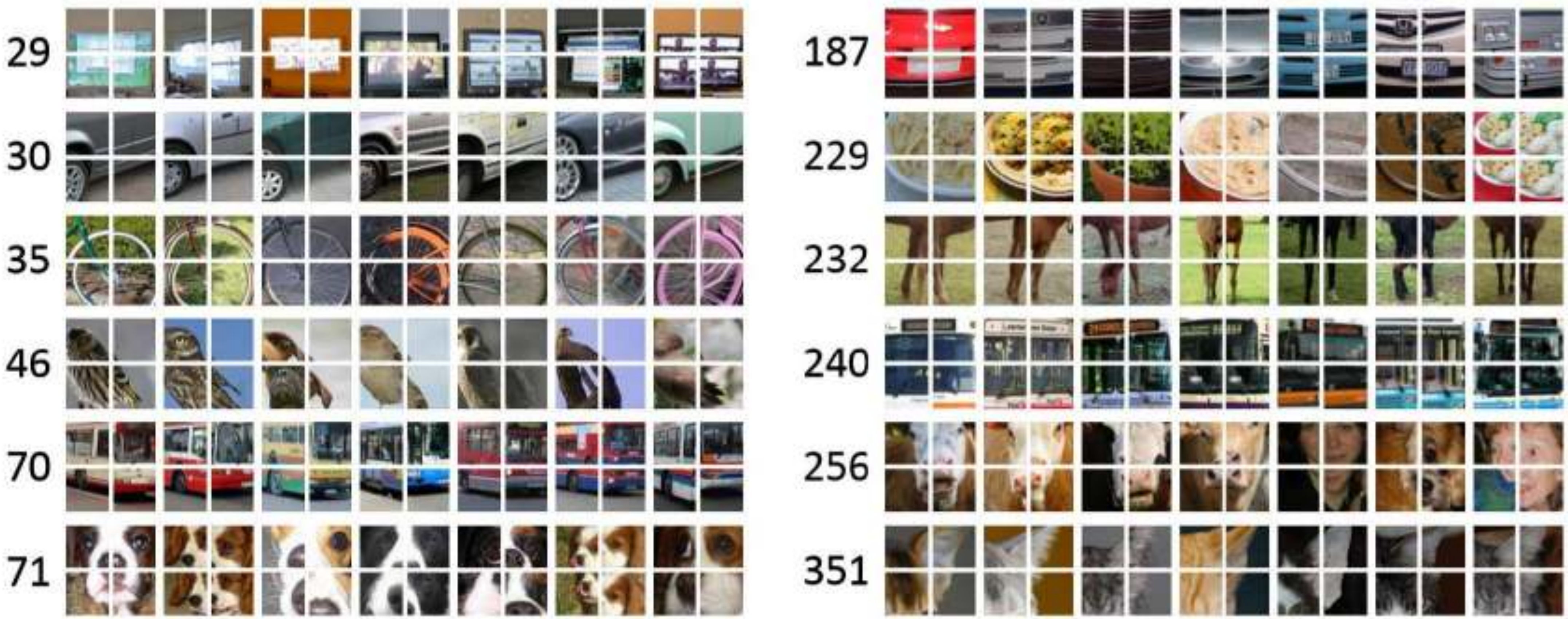
Carl Doersch, Abhinav Gupta, and Alexei A. Efros. “Unsupervised visual representation learning by context prediction.” ICCV. 2015.

relative position



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

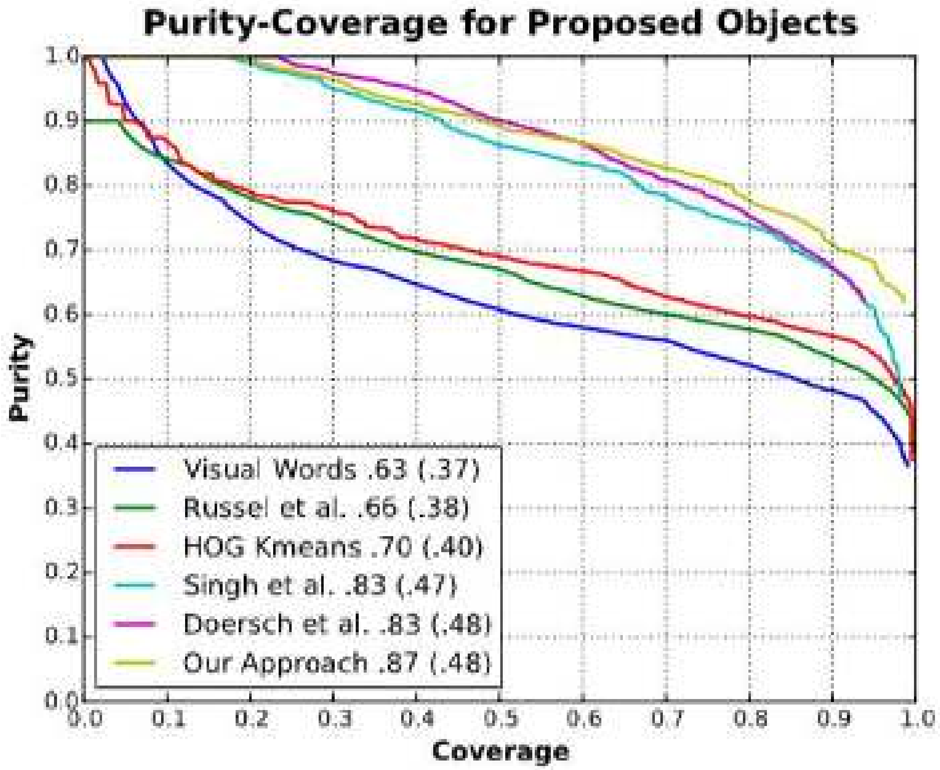
relative position



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

relative position

Figure 8. Clusters discovered and automatically ranked via our algorithm (§ 4.5) from the Paris Street View dataset.



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

jigsaw puzzle

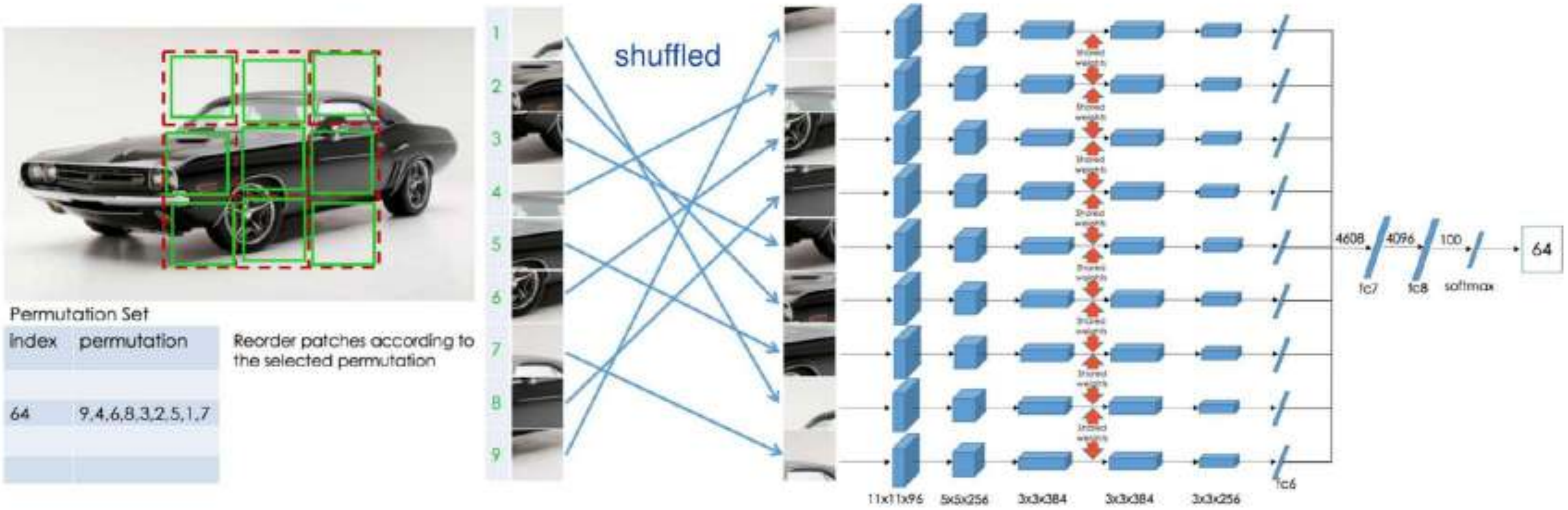
- The model is trained to place 9 shuffled patches **back to the original locations.**

(a) (b) (c)

Mehdi Noroozi & Paolo Favaro. “Unsupervised learning of visual representations by solving jigsaw puzzles.” ECCV, 2016.

jigsaw puzzle

多智体人工智能实验室
Multi-Agent Artificial Intelligence Laboratory



Mehdi Noroozi & Paolo Favaro. “Unsupervised learning of visual representations by solving jigsaw puzzles.” ECCV, 2016.

jigsaw puzzle

多智体人工智能实验室
Multi-Agent Artificial Intelligence Laboratory

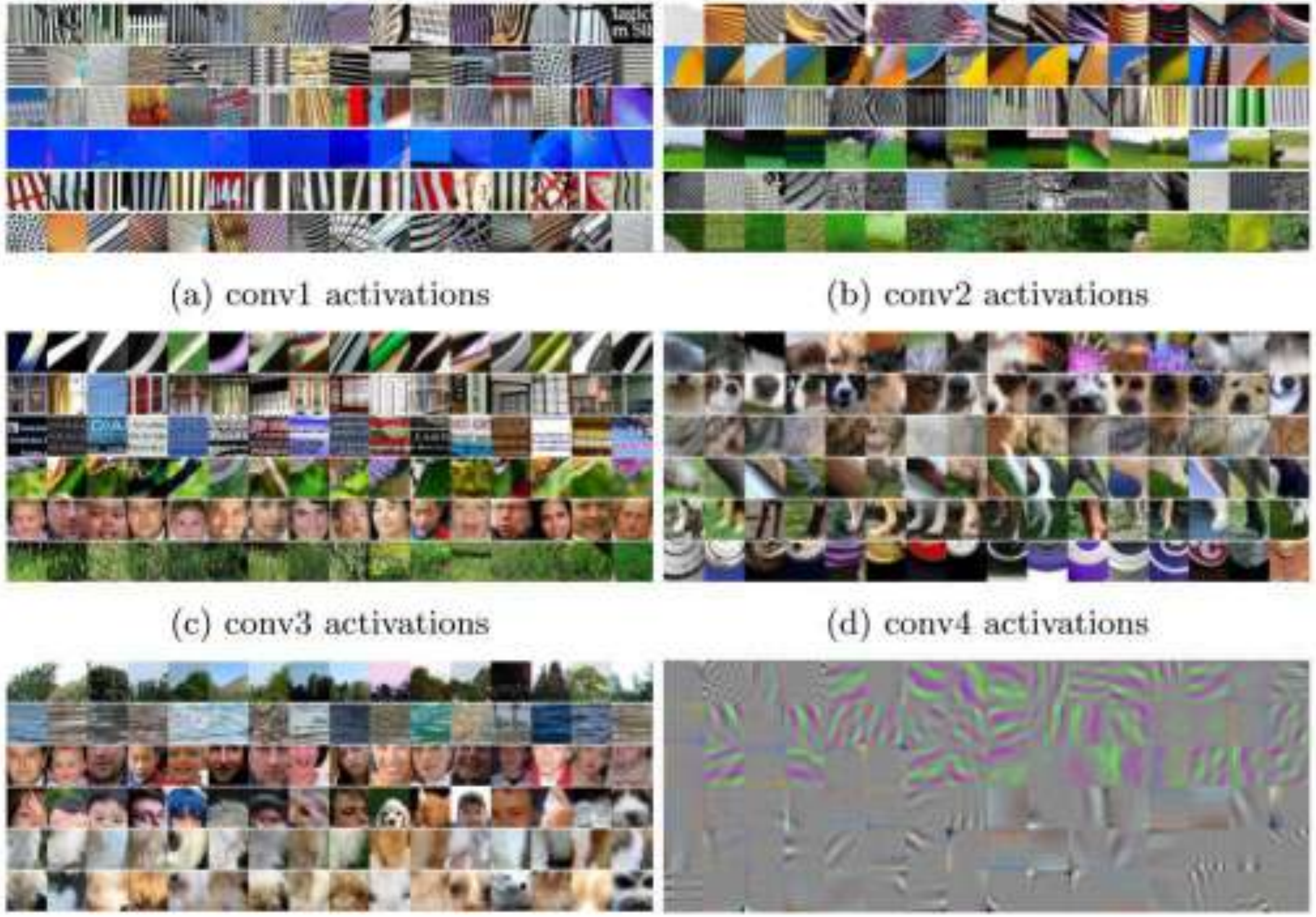
- A convolutional network processes each patch **independently** with shared weights and outputs a probability vector per patch index out of a predefined set of permutations.

- To **control the difficulty** of jigsaw puzzles, the paper proposed to shuffle patches according to a **predefined permutation** set and configured the model to predict a probability vector over all the indices in the set.

- GCN

Mehdi Noroozi & Paolo Favaro. “Unsupervised learning of visual representations by solving jigsaw puzzles.” ECCV, 2016.

jigsaw puzzle



(a) conv1 activations (b) conv2 activations

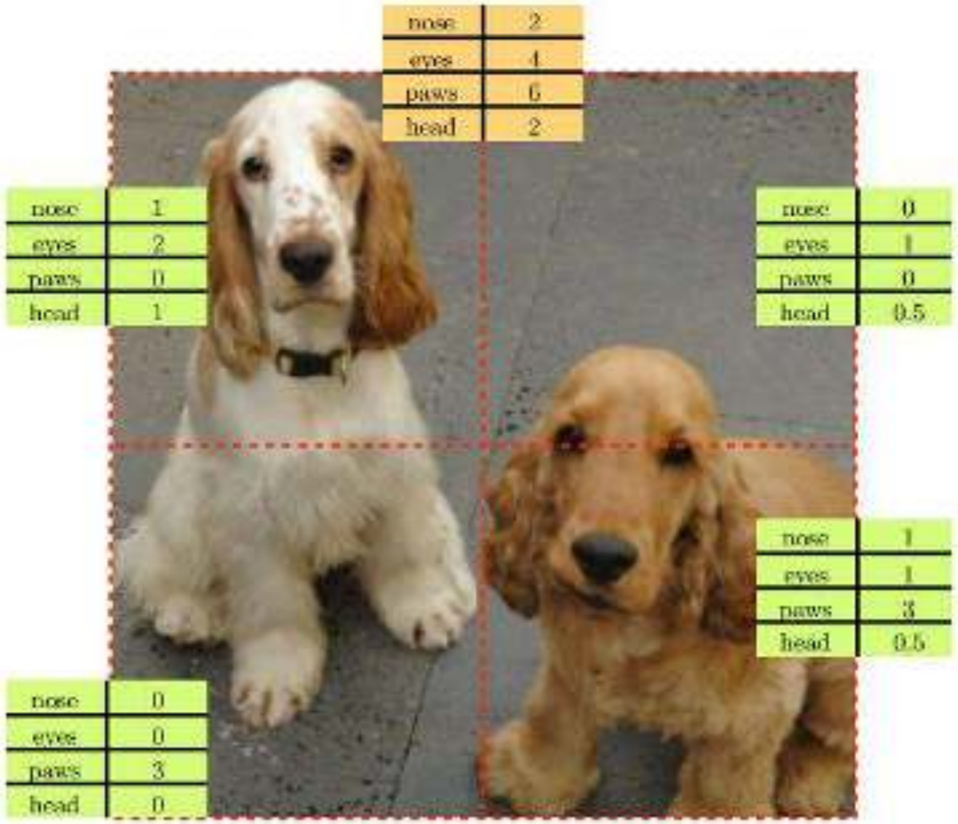
(c) conv3 activations (d) conv4 activations

(e) conv5 activations (f) conv1 filters without color jittering

Mehdi Noroozi & Paolo Favaro. “Unsupervised learning of visual representations by solving jigsaw puzzles.” ECCV, 2016.

counting features

- consider "**feature**" or "visual primitives" as a scalar-value attribute
- that **can be summed up** over multiple patches and compared across different patches



Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. "Representation learning by learning to count." ICCV. 2017.

counting features

- *Scaling*: If an image is **scaled up by 2x**, the number of visual primitives should stay the same.

- *Tiling*: If an image is **tiled into a 2x2 grid**, the number of visual primitives is expected to **be the sum**, 4 times the original feature counts.

Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. "Representation learning by learning to count." ICCV. 2017.

counting features

1. Downsampling operator, $D : \mathbb{R}^{m\times n\times 3} \mapsto \mathbb{R}^{\frac{m}{2}\times\frac{n}{2}\times 3}$: downsample by a factor of 2
2. Tiling operator $T_i : \mathbb{R}^{m\times n\times 3} \mapsto \mathbb{R}^{\frac{m}{2}\times\frac{n}{2}\times 3}$: extract the i-th tile from a 2x2 grid of the image.

We expect to learn:

$$\phi(\mathbf{x}) = \phi(D \circ \mathbf{x}) = \sum_{i=1}^{4} \phi(T_i \circ \mathbf{x})$$

Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. “Representation learning by learning to count.” ICCV. 2017.

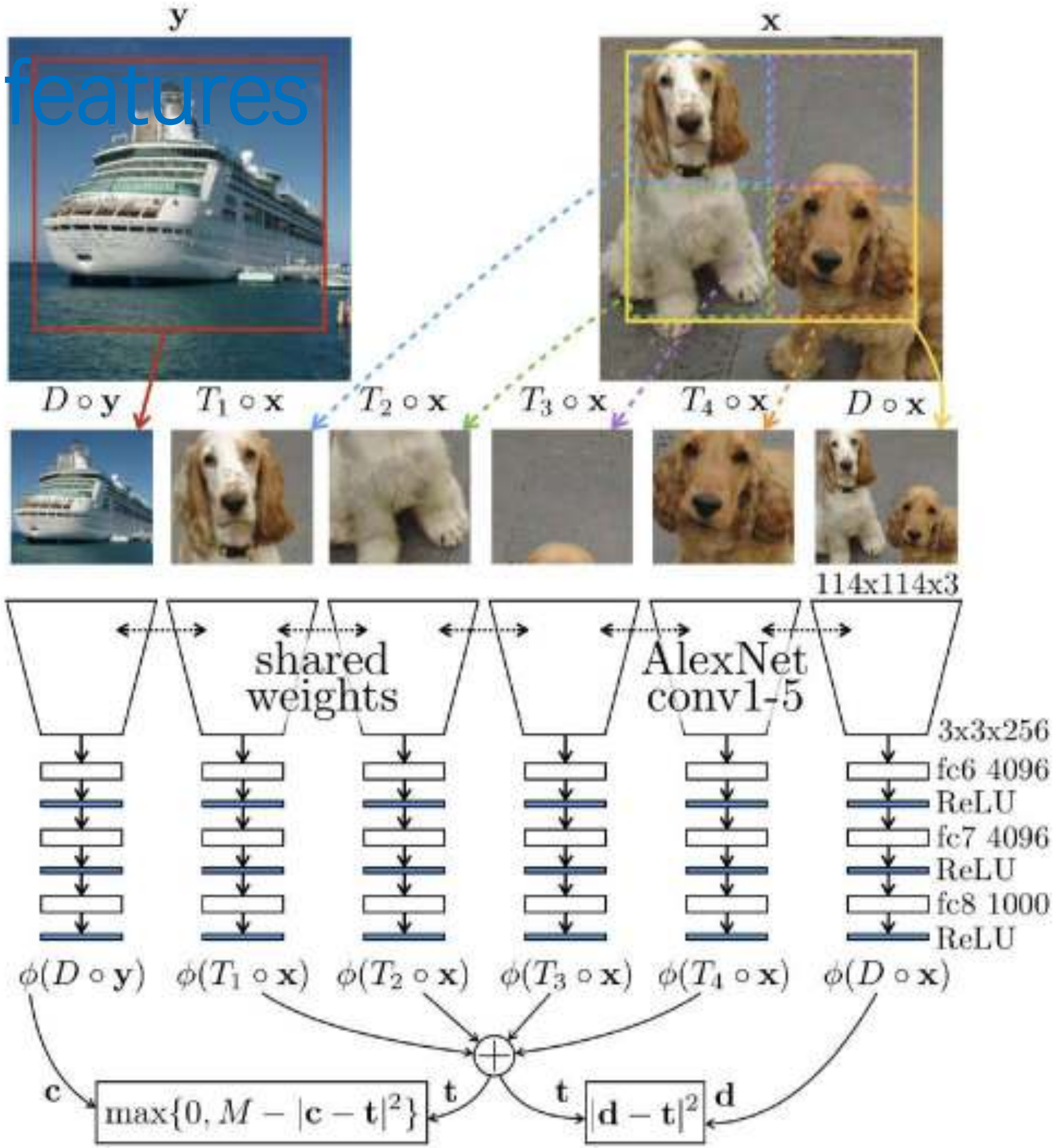
counting features

Thus the MSE loss is: $\mathcal{L}_{\text{feat}} = \|\phi(D \circ \mathbf{x}) - \sum_{i=1}^{4} \phi(T_i \circ \mathbf{x})\|_2^2$. To avoid trivial solution $\phi(\mathbf{x}) = \mathbf{0}, \forall \mathbf{x}$, another loss term is added to encourage the difference between features of two different images: $\mathcal{L}_{\text{diff}} = \max(0, c - \|\phi(D \circ \mathbf{y}) - \sum_{i=1}^{4} \phi(T_i \circ \mathbf{x})\|_2^2)$, where $\mathbf{y}$ is another input image different from $\mathbf{x}$ and c is a scalar constant. The final loss is:

$$\mathcal{L} = \mathcal{L}_{\text{feat}} + \mathcal{L}_{\text{diff}} = \|\phi(D \circ \mathbf{x}) - \sum_{i=1}^{4} \phi(T_i \circ \mathbf{x})\|_2^2 + \max(0, M - \|\phi(D \circ \mathbf{y}) - \sum_{i=1}^{4} \phi(T_i \circ \mathbf{x})\|_2^2)$$

Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. “Representation learning by learning to count.” ICCV. 2017.

counting features



Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. "Representation learning by learning to count." ICCV. 2017.

counting features

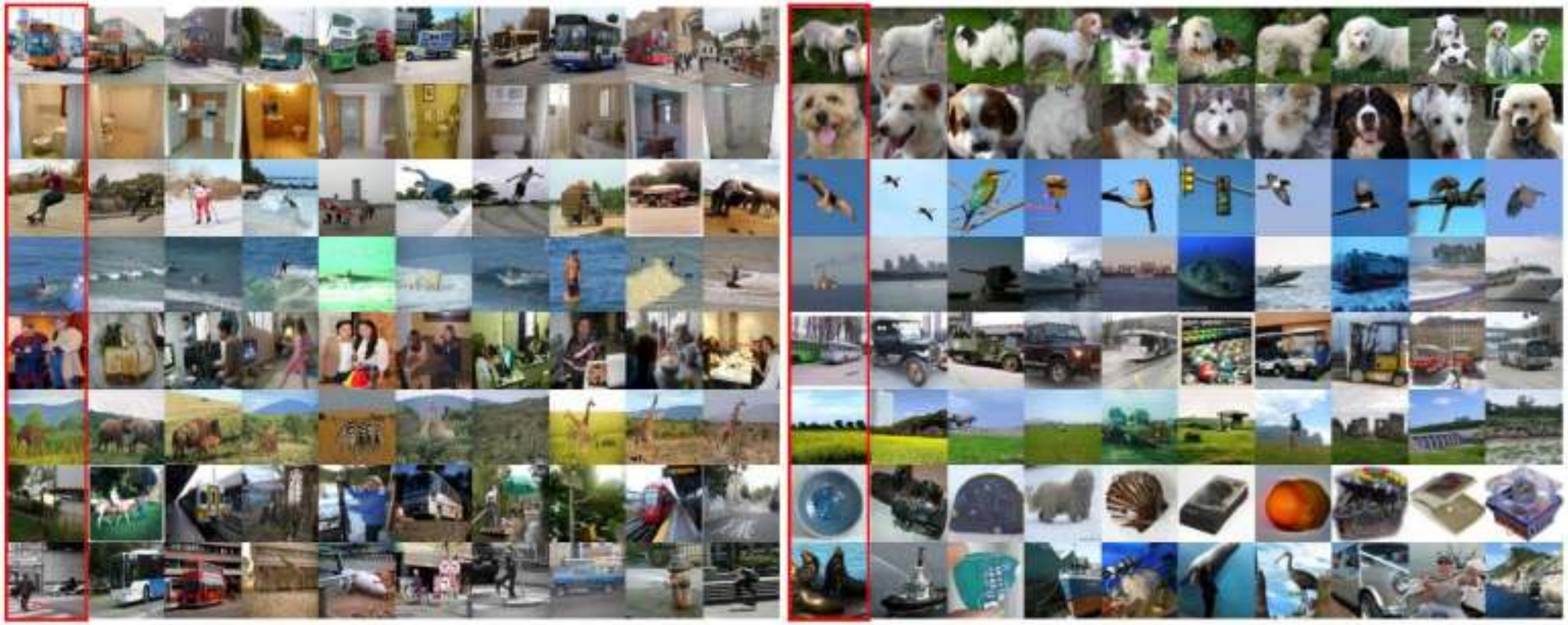


Figure 7: **Nearest neighbor retrievals.** Left: COCO retrievals. Right: ImageNet retrievals. In both datasets, the leftmost column (with a red border) shows the queries and the other columns show the top matching images sorted with increasing Euclidean distance in our counting feature space from left to right. On the bottom 3 rows, we show the failure retrieval cases. Note that the matches share a similar content and scene outline.

Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. "Representation learning by learning to count." ICCV. 2017.

Image-based

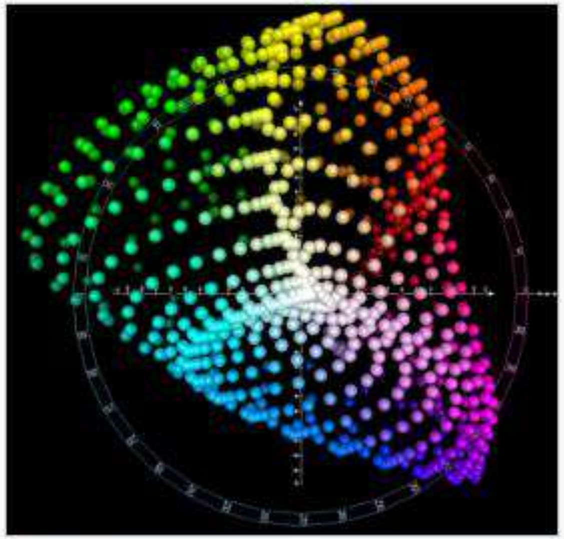
Colorzation

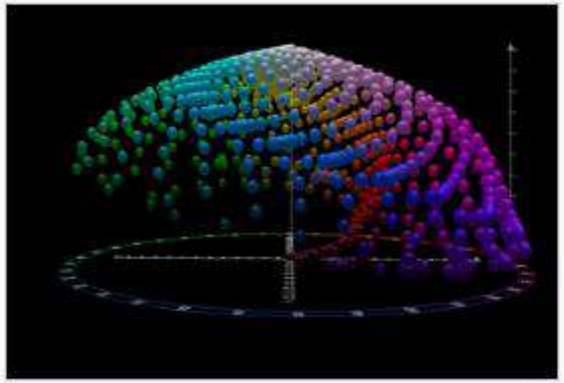
Colorization - CIE Lab* color space

- The Lab* color is designed to approximate human vision, while, in contrast, RGB or CMYK models the color output of physical devices.

- **L*** component matches human perception of lightness; L* = 0 is black and L* = 100 indicates white.
- **a*** component represents green (negative) / magenta (positive) value.
- **b*** component models blue (negative) /yellow (positive) value.

https://en.wikipedia.org/wiki/CIELAB_color_space


CIELAB color space top view


CIELAB color space front view

Colorization

2.1 Objective Function

Given an input lightness channel $\mathbf{X} \in \mathbb{R}^{H \times W \times 1}$, our objective is to learn a mapping $\widehat{\mathbf{Y}} = \mathcal{F}(\mathbf{X})$ to the two associated color channels $\mathbf{Y} \in \mathbb{R}^{H \times W \times 2}$, where H, W are image dimensions.

(We denote predictions with a $\widehat{\cdot}$ symbol and ground truth without.) We perform this task in CIE *Lab* color space. Because distances in this space model perceptual distance, a natural objective function, as used in [1,2], is the Euclidean loss $\mathrm{L}_2(\cdot,\cdot)$ between predicted and ground truth colors:

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

However, this loss is not robust to the inherent ambiguity and multimodal nature of the colorization problem. If an object can take on a set of distinct *ab* values, the optimal solution to the Euclidean loss will be the mean of the

Richard Zhang, Phillip Isola & Alexei A. Efros. “Colorful image colorization.” ECCV, 2016.

Colorization

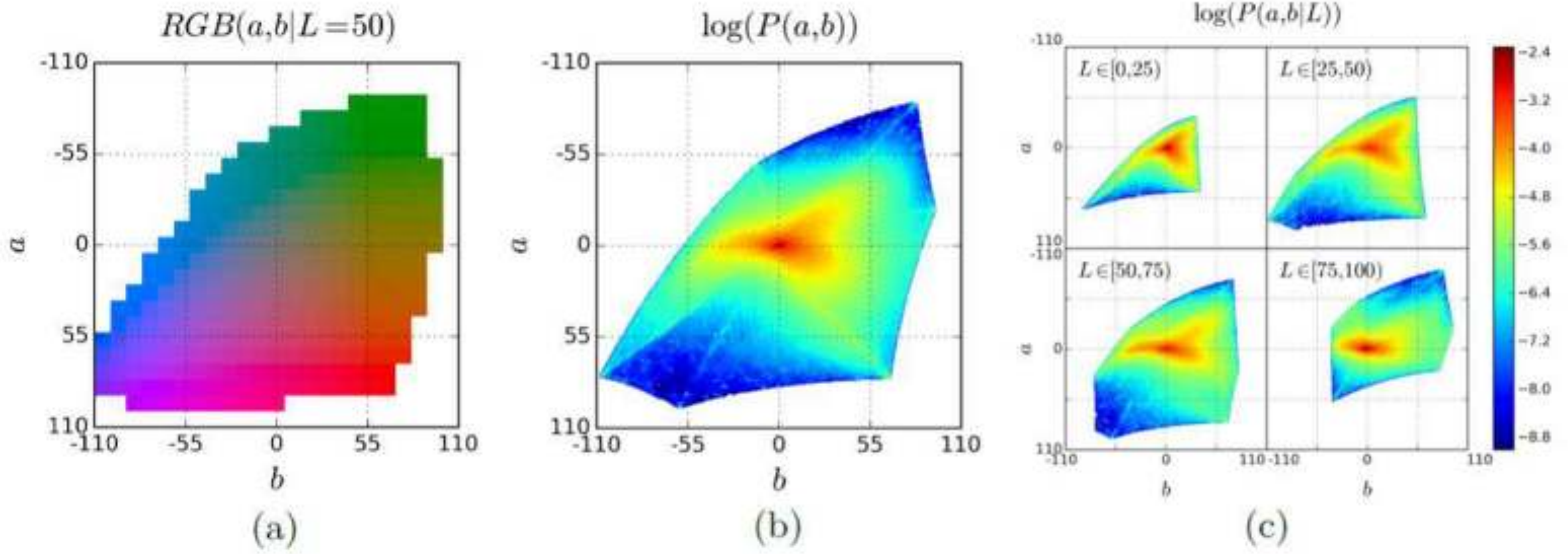


Fig. 3. (a) Quantized *ab* color space with a grid size of 10. A total of 313 *ab* pairs are in gamut. (b) Empirical probability distribution of *ab* values, shown in log scale. (c) Empirical probability distribution of *ab* values, conditioned on *L*, shown in log scale.

Richard Zhang, Phillip Isola & Alexei A. Efros. “Colorful image colorization.” ECCV, 2016.

Colorization

多智体人工智能实验室 Multi-Agent Artificial Intelligence Laboratory

Instead, we treat the problem as multinomial classification. We quantize the ab output space into bins with grid size 10 and keep the $Q = 313$ values which are in-gamut, as shown in Figure 3(a). For a given input $\mathbf{X}$, we learn a mapping $\widehat{\mathbf{Z}} = \mathcal{G}(\mathbf{X})$ to a probability distribution over possible colors $\widehat{\mathbf{Z}} \in [0,1]^{H \times W \times Q}$, where Q is the number of quantized ab values.

To compare predicted $\widehat{\mathbf{Z}}$ against ground truth, we define function $\mathbf{Z} = \mathcal{H}_{gt}^{-1}(\mathbf{Y})$, which converts ground truth color $\mathbf{Y}$ to vector $\mathbf{Z}$, using a soft-encoding scheme[2]. We then use multinomial cross entropy loss $\mathrm{L}_{cl}(\cdot,\cdot)$, defined as:

$$\mathrm{L}_{cl}(\widehat{\mathbf{Z}}, \mathbf{Z}) = -\sum_{h,w} v(\mathbf{Z}_{h,w}) \sum_{q} \mathbf{Z}_{h,w,q} \log(\widehat{\mathbf{Z}}_{h,w,q}) \tag{2}$$

Richard Zhang, Phillip Isola & Alexei A. Efros. “Colorful image colorization.” ECCV, 2016.

Colorization

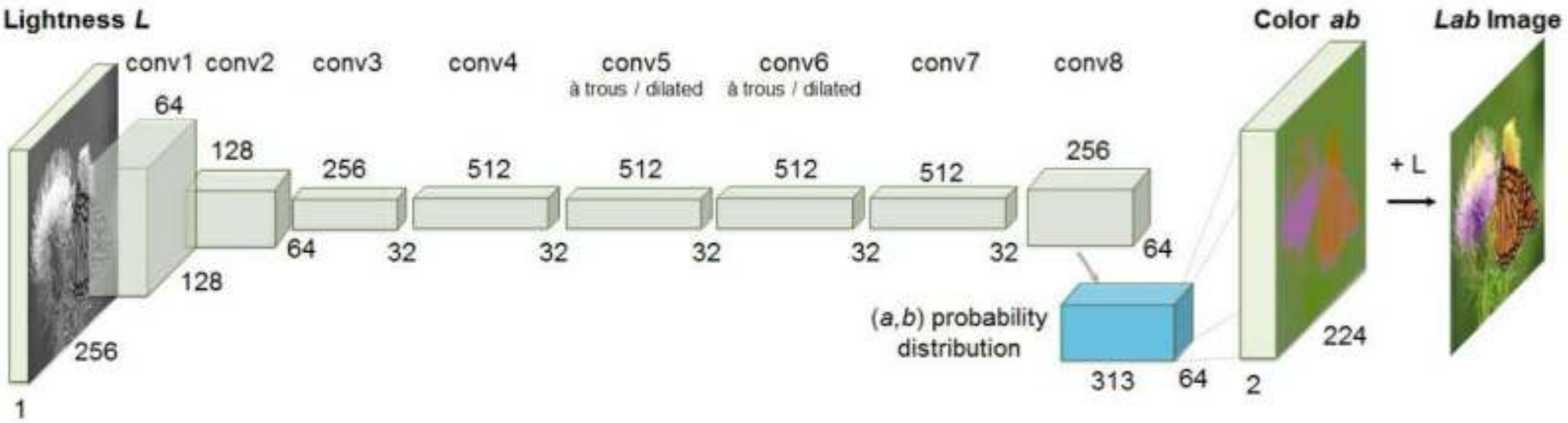
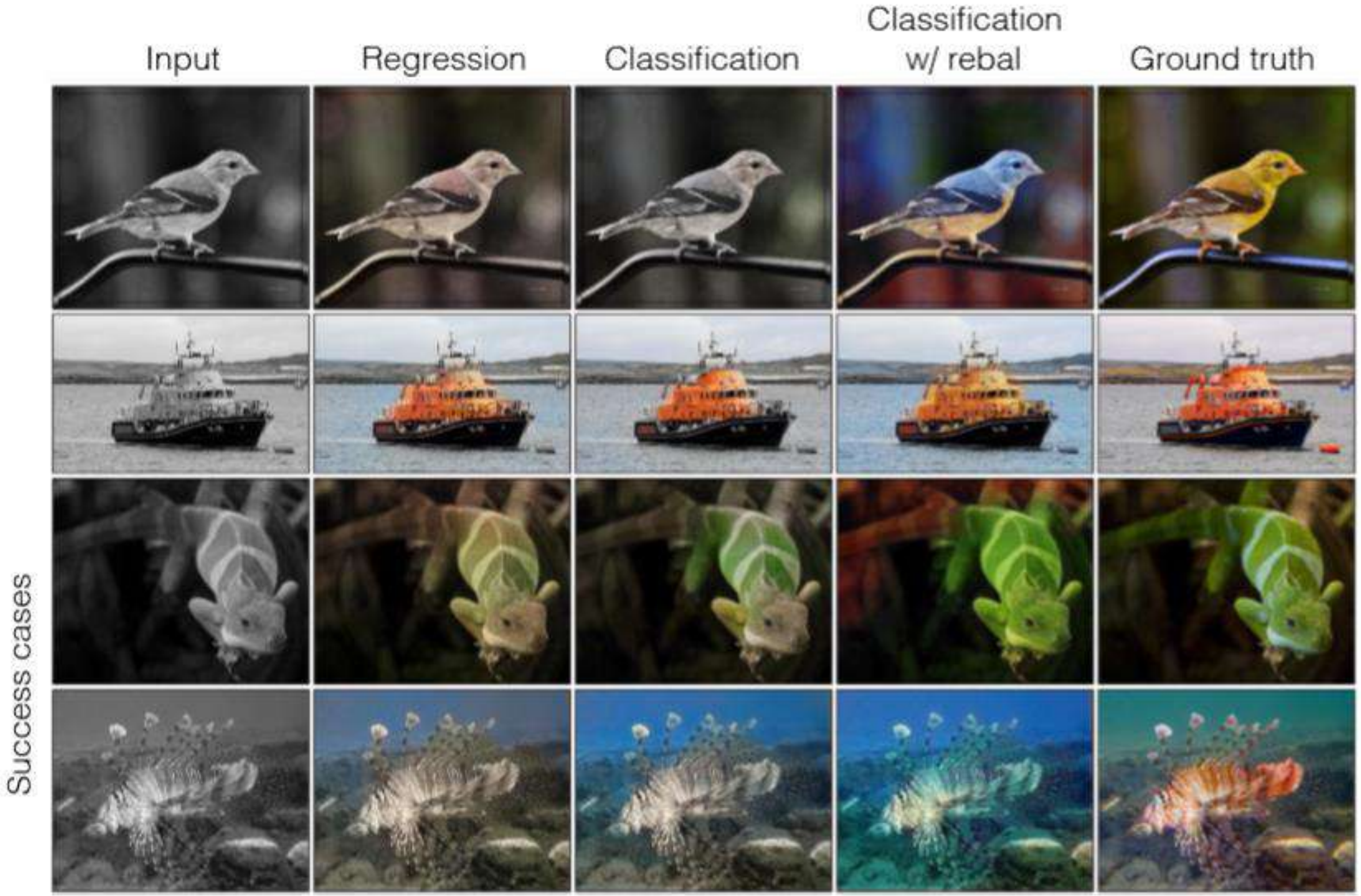


Fig. 2. Our network architecture. Each conv layer refers to a block of 2 or 3 repeated conv and ReLU layers, followed by a BatchNorm [30] layer. The net has no pool layers. All changes in resolution are achieved through spatial downsampling or upsampling between conv blocks.

Richard Zhang, Phillip Isola & Alexei A. Efros. "Colorful image colorization." ECCV, 2016.

Colorization



Richard Zhang, Phillip Isola & Alexei A. Efros. “Colorful image colorization.” ECCV, 2016.

Colorization

Fig. 8. Applying our method to legacy black and white photos. Left to right: photo by David Fleay of a Thylacine, now extinct, 1936; photo by Ansel Adams of Yosemite; amateur family photo from 1956; *Migrant Mother* by Dorothea Lange, 1936.

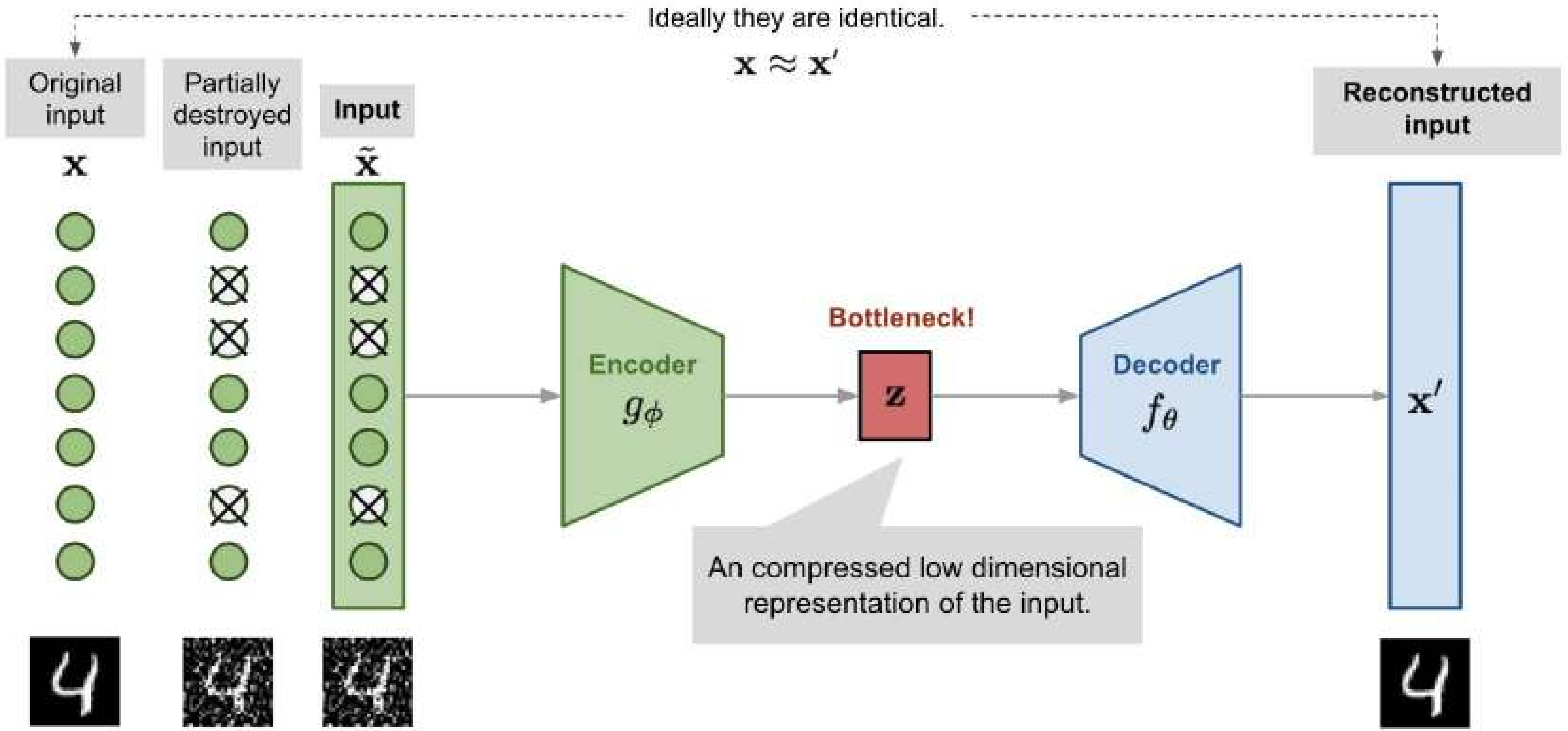
Richard Zhang, Phillip Isola & Alexei A. Efros. "Colorful image colorization." ECCV, 2016.

Image-based

Generative Modeling

denoising autoencoder
context encoder
split-brain autoencoder
Bidirectional GANs

denoising autoencoder



Pascal Vincent, et al. “Extracting and composing robust features with denoising autoencoders.” ICML, 2008

denoising autoencoder

- an image from a version that is partially **corrupted** or has random **noise.**

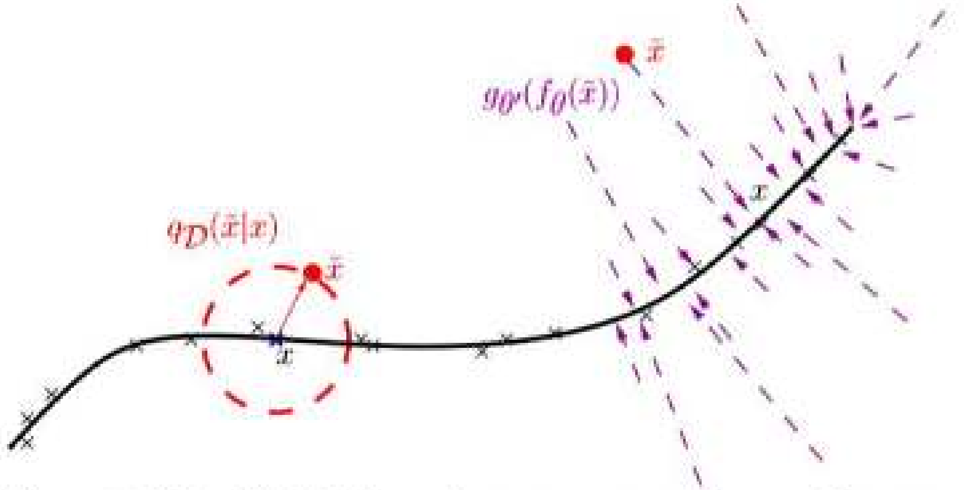
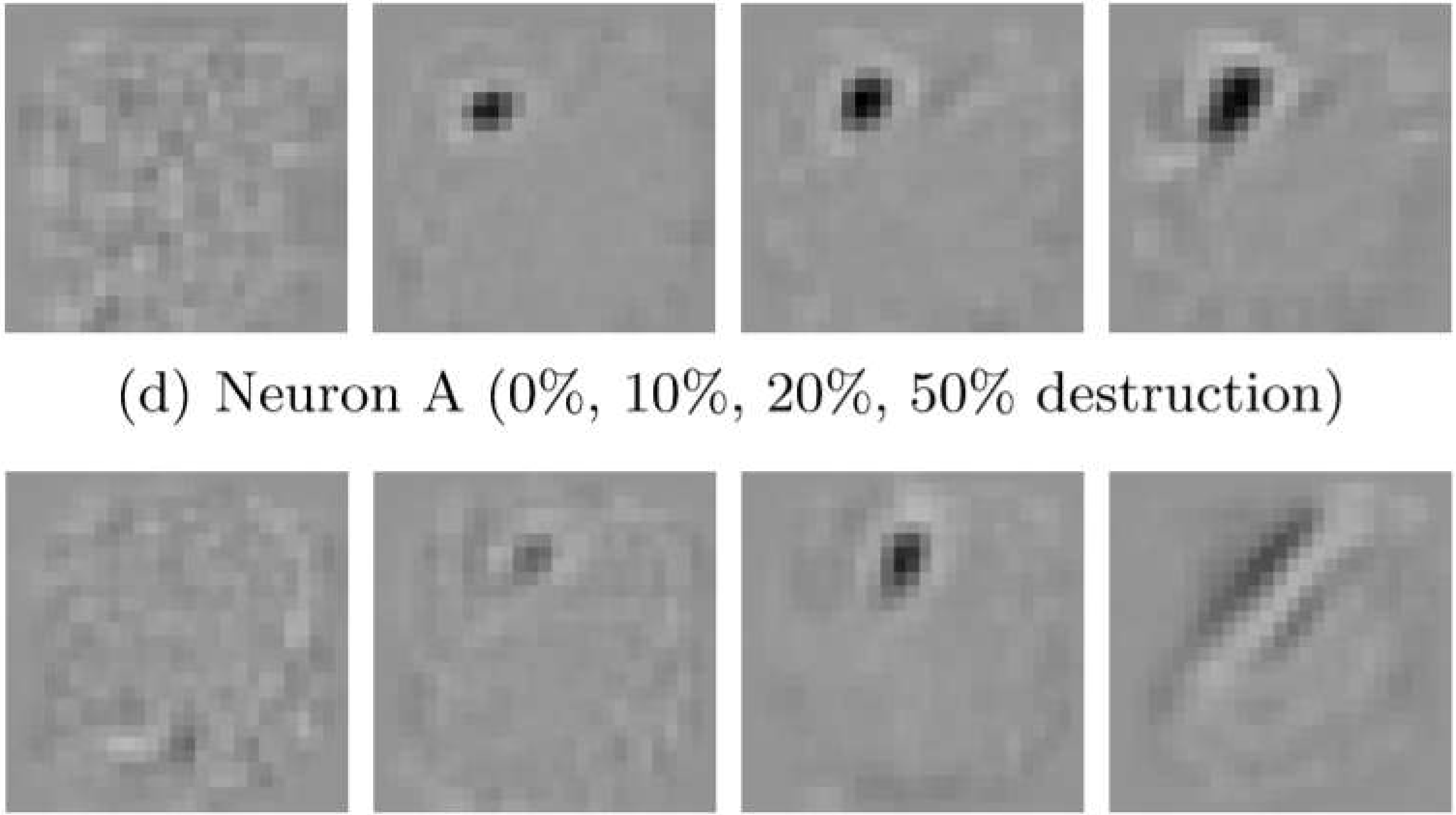


Figure 2. **Manifold learning perspective.** Suppose training data (×) concentrate near a low-dimensional manifold. Corrupted examples (•) obtained by applying corruption process $q_\mathcal{D}(\widetilde{X}|X)$ will lie farther from the manifold. The model learns with $p(X|\widetilde{X})$ to "project them back" onto the manifold. Intermediate representation Y can be interpreted as a coordinate system for points on the manifold.

Pascal Vincent, et al. "Extracting and composing robust features with denoising autoencoders." ICML, 2008

denoising autoencoder



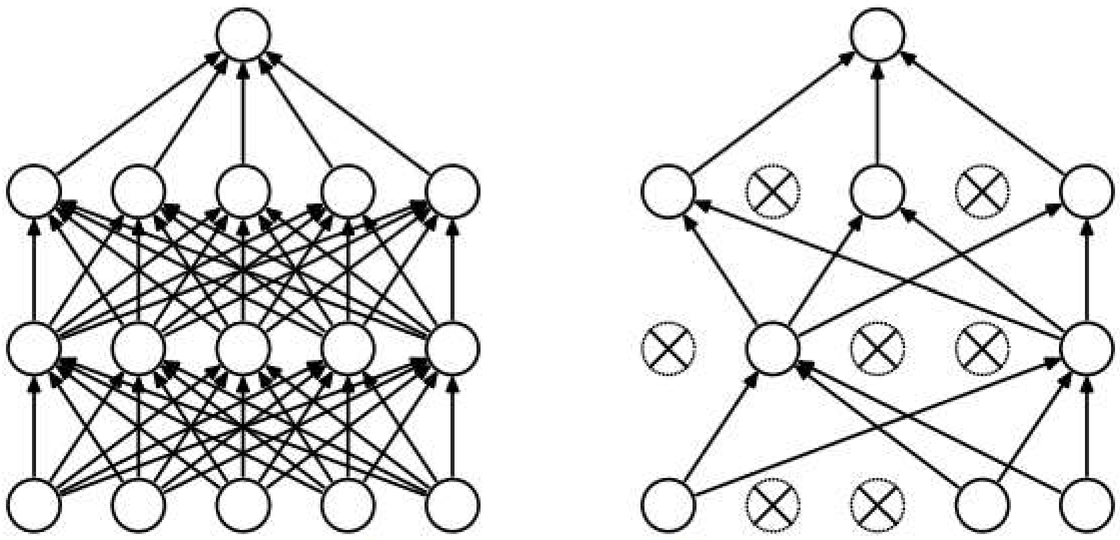
(d) Neuron A (0%, 10%, 20%, 50% destruction)

(e) Neuron B (0%, 10%, 20%, 50% destruction)

Pascal Vincent, et al. “Extracting and composing robust features with denoising autoencoders.” ICML, 2008

denoising autoencoder

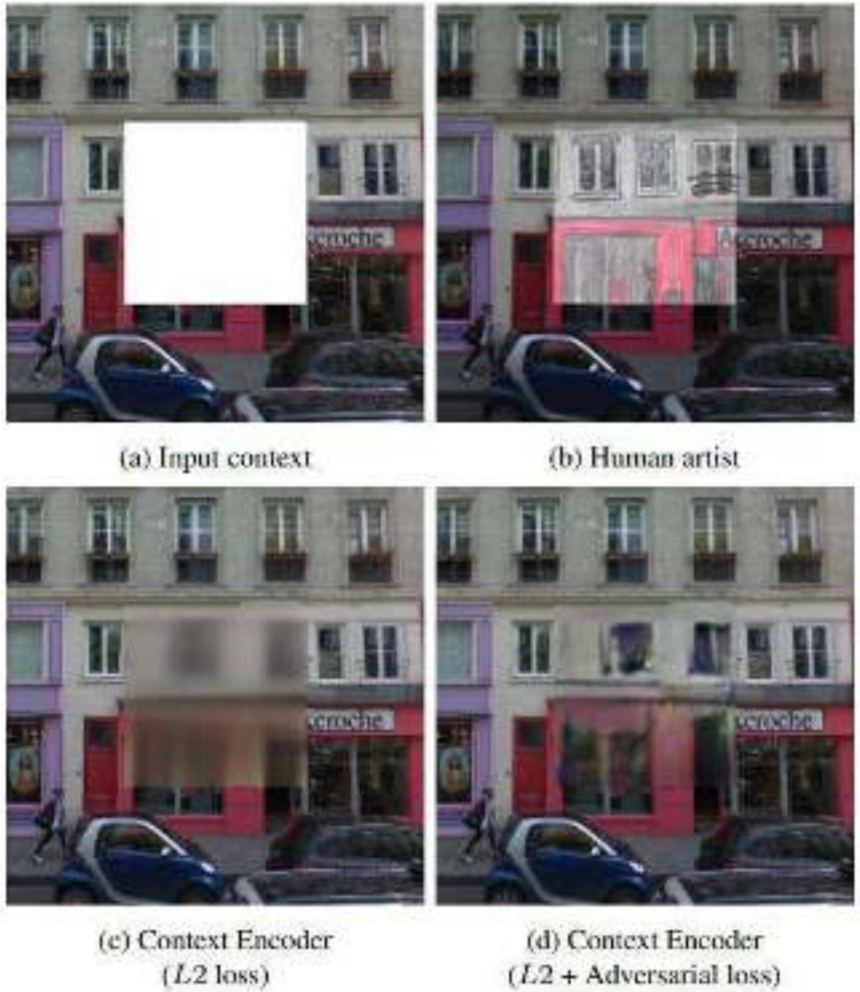
- Sounds a lot like dropout, right?
- Well, the denoising autoencoder was proposed in 2008, 4 years before the dropout paper



(a) Standard Neural Net (b) After applying dropout.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. "Improving neural networks by preventing co-adaptation of feature detectors." arXiv preprint arXiv:1207.0580 (2012).

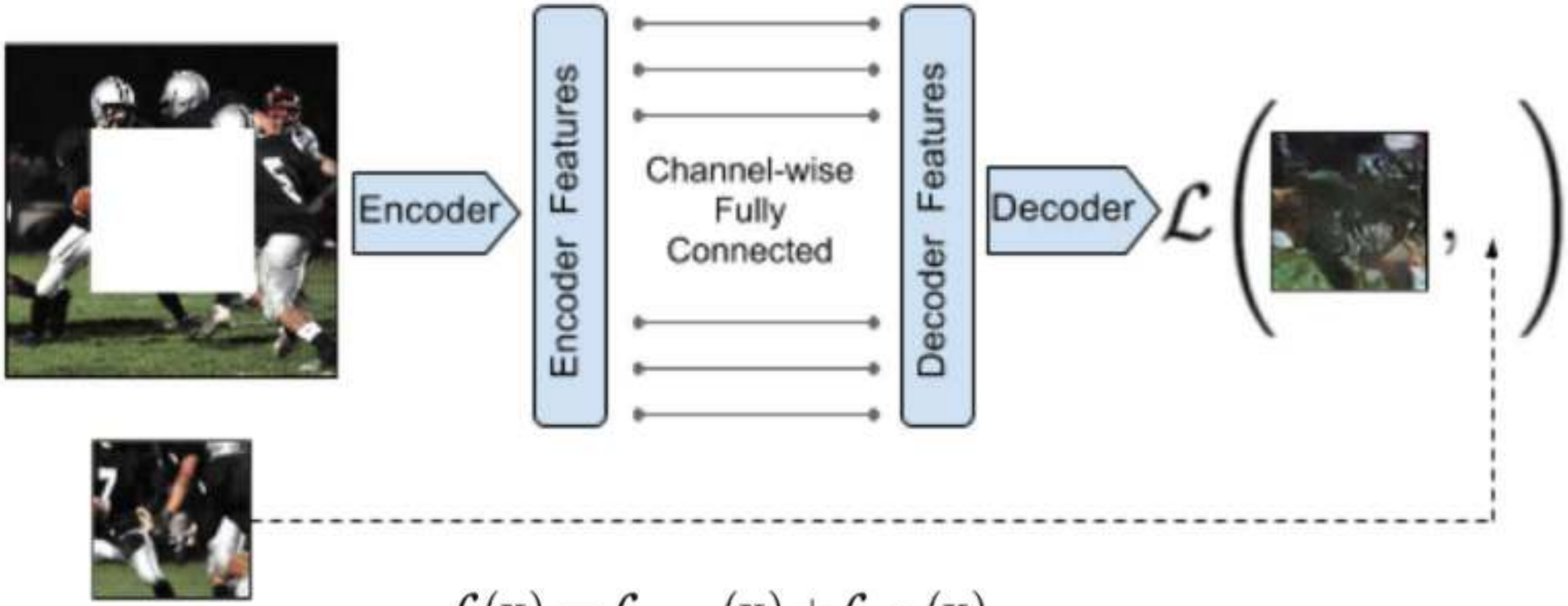
context encoder



(a) Input context
(b) Human artist
(c) Context Encoder ($L2$ loss)
(d) Context Encoder ($L2$ + Adversarial loss)

Deepak Pathak, et al. “Context encoders: Feature learning by inpainting.” CVPR. 2016.

context encoder



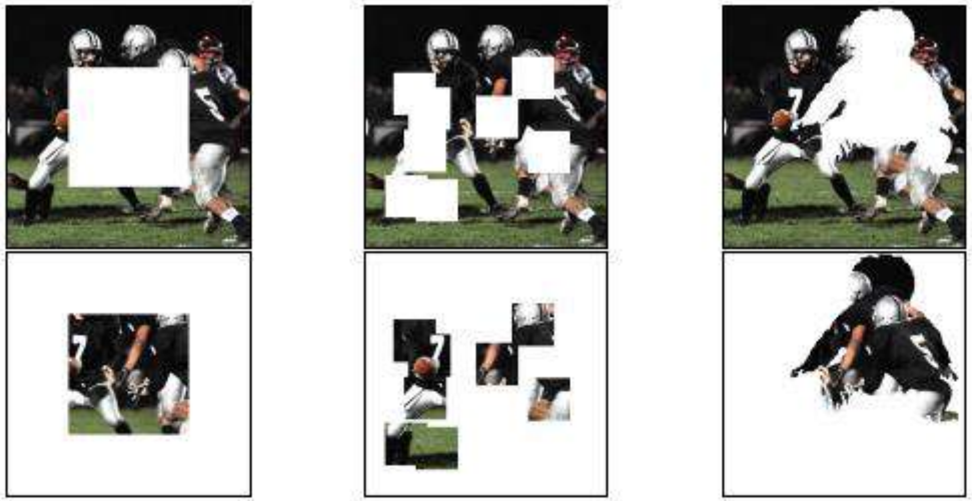
$$\mathcal{L}(\mathbf{x}) = \mathcal{L}_{recon}(\mathbf{x}) + \mathcal{L}_{adv}(\mathbf{x})$$

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

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

Deepak Pathak, et al. “Context encoders: Feature learning by inpainting.” CVPR. 2016.

context encoder

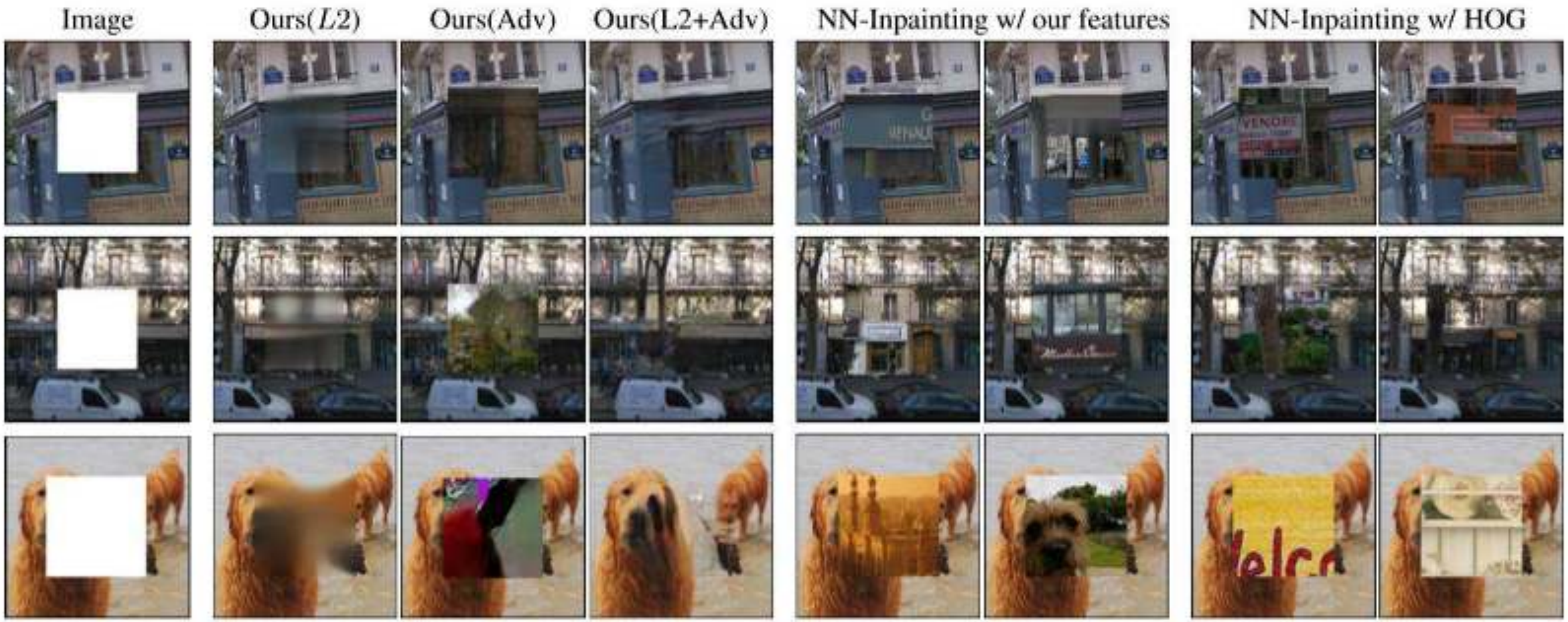


(a) Central region (b) Random block (c) Random region

Figure 3: An example of image x with our different region masks $\hat{M}$ applied, as described in Section 3.3.

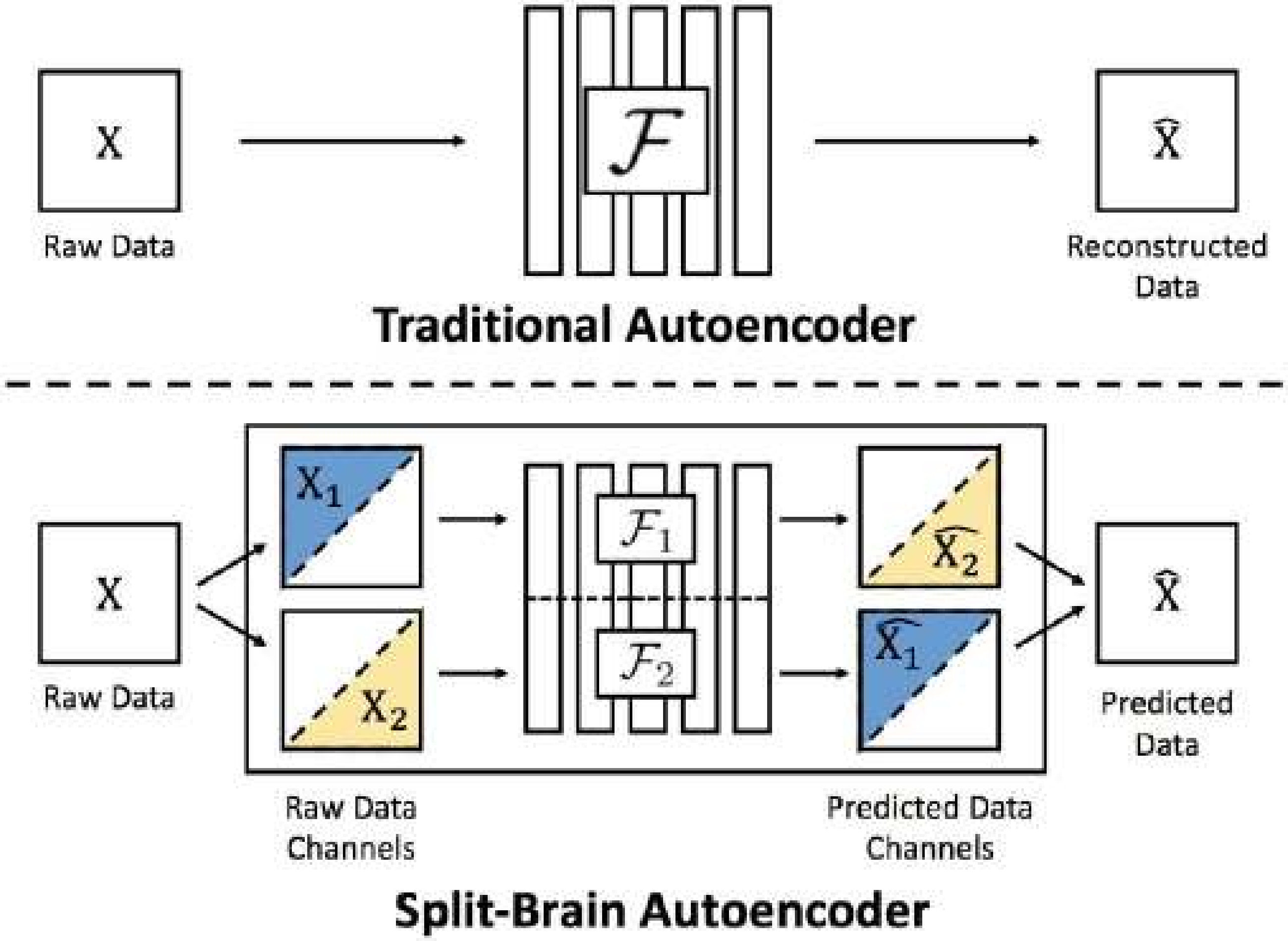
Deepak Pathak, et al. “Context encoders: Feature learning by inpainting.” CVPR. 2016.

context encoder



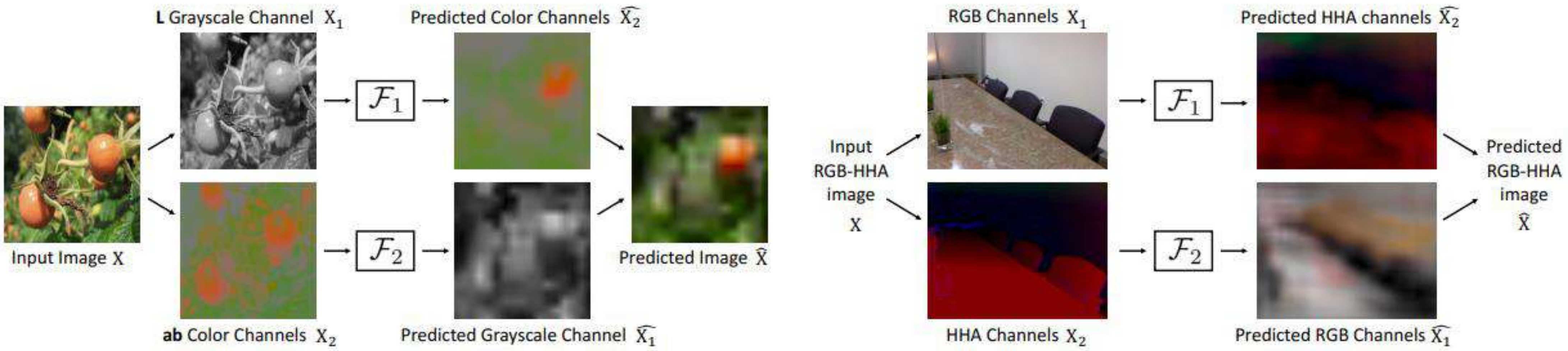
Deepak Pathak, et al. “Context encoders: Feature learning by inpainting.” CVPR. 2016.

Split-brain Autoencoder



Richard Zhang, Phillip Isola, and Alexei A. Efros. "Split-brain autoencoders: Unsupervised learning by cross-channel prediction." CVPR. 2017.

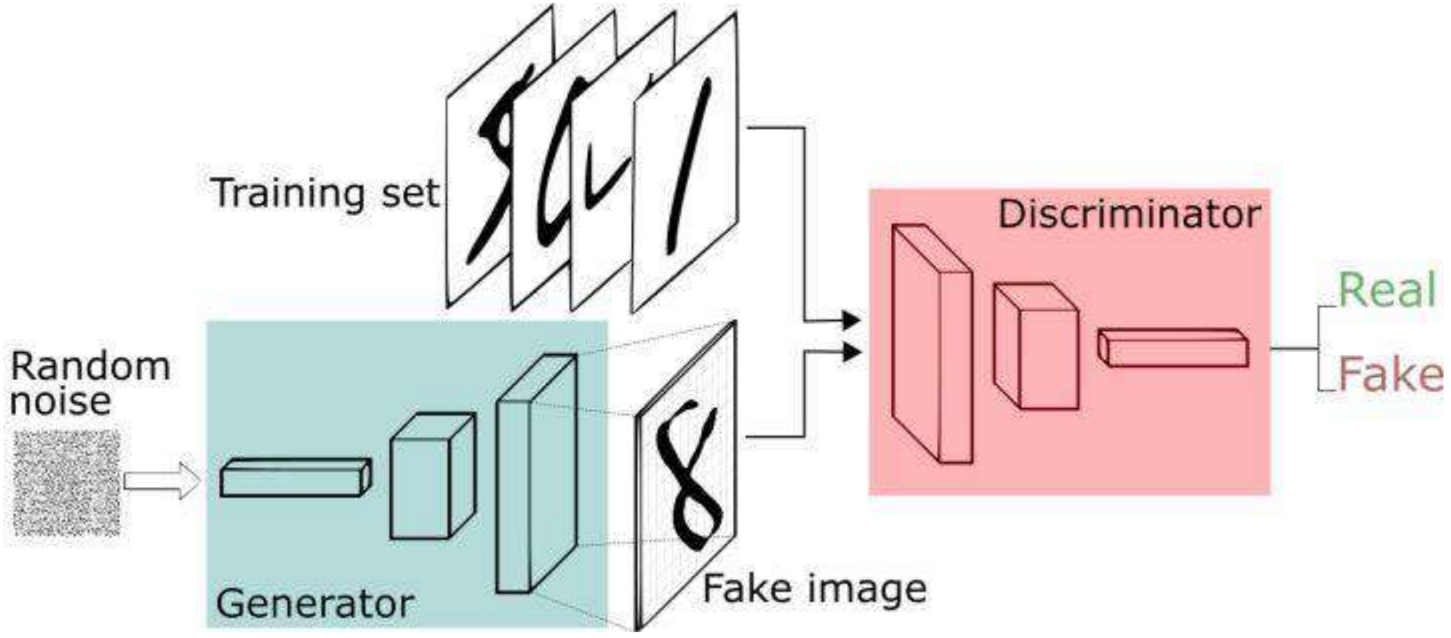
Split-brain Autoencoder



(a) ***Lab* Images** (b) **RGB-D Images**

Richard Zhang, Phillip Isola, and Alexei A. Efros. "Split-brain autoencoders: Unsupervised learning by cross-channel prediction." CVPR. 2017.

GAN



$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

Alec Radford, Luke Metz, Soumith Chintala, **Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks.**ICLR 2016

Bidirectional GAN

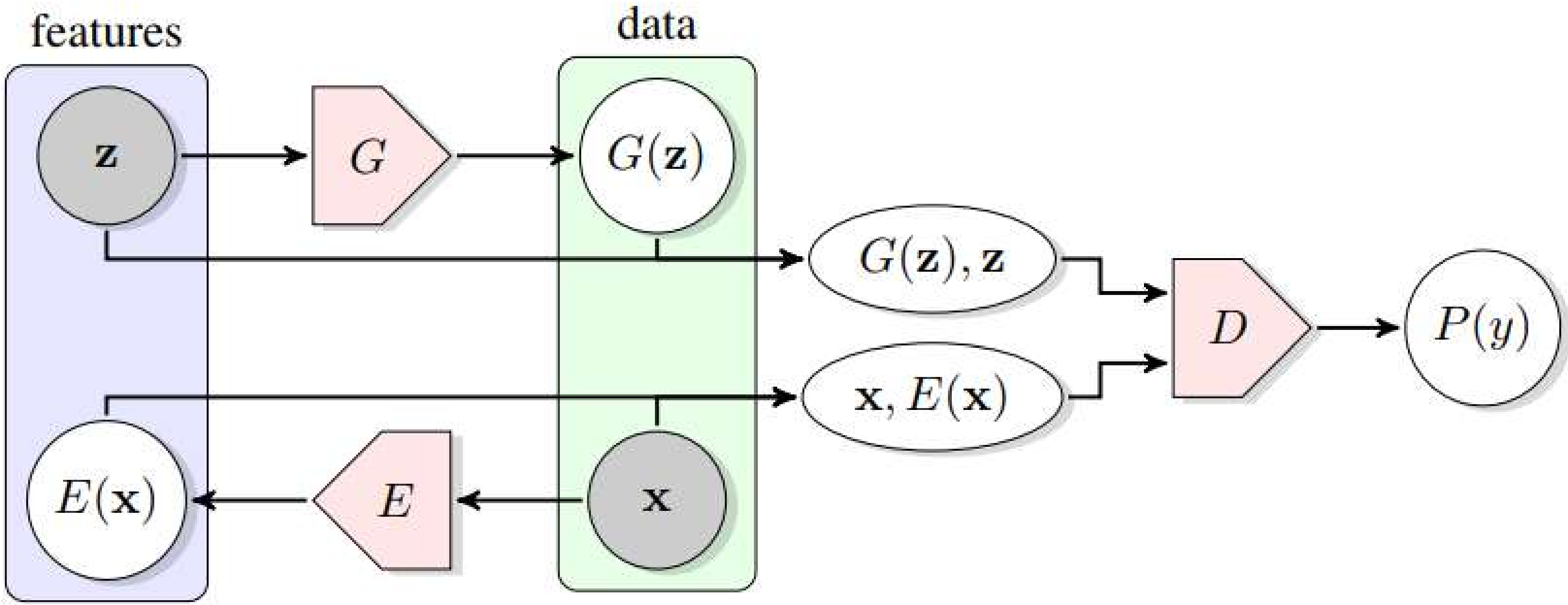


Figure 1: The structure of Bidirectional Generative Adversarial Networks (BiGAN).

$$V(D,E,G) := \mathbb{E}_{\mathbf{x}\sim p_{\mathbf{X}}}\Big[\underbrace{\mathbb{E}_{\mathbf{z}\sim p_E(\cdot|\mathbf{x})}\left[\log D(\mathbf{x},\mathbf{z})\right]}_{\log D(\mathbf{x},E(\mathbf{x}))}\Big] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{Z}}}\Big[\underbrace{\mathbb{E}_{\mathbf{x}\sim p_G(\cdot|\mathbf{z})}\left[\log\left(1-D(\mathbf{x},\mathbf{z})\right)\right]}_{\log(1-D(G(\mathbf{z}),\mathbf{z}))}\Big].$$

Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. “Adversarial feature learning.” ICLR 2017.

Self-Supervised Learning

Images-Based

Video-Based

Control-Based

Video-Based

Tracking

Frame Sequence

Video Colorization

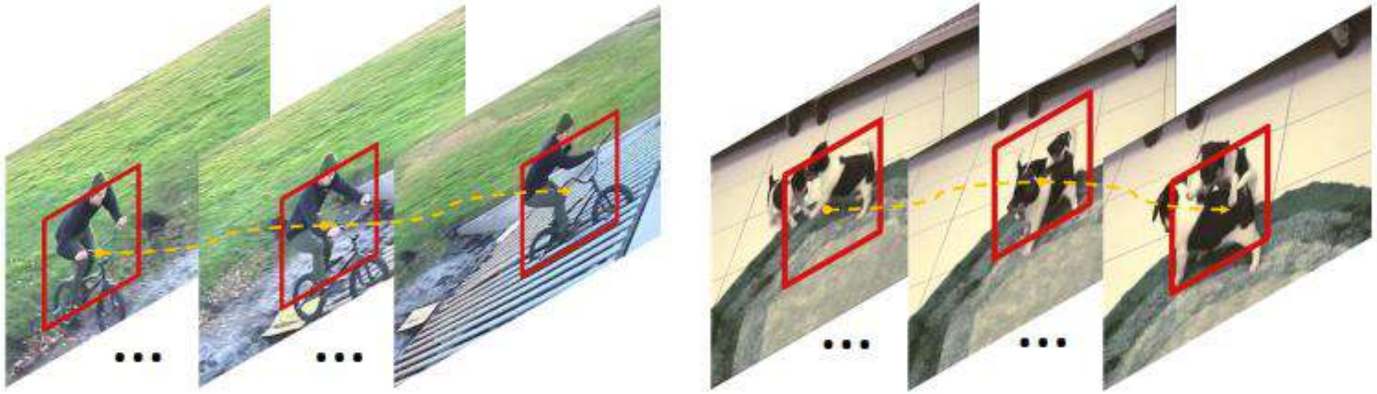
Video-Based

- Tracking
- Frame Sequence
- Video Colorization

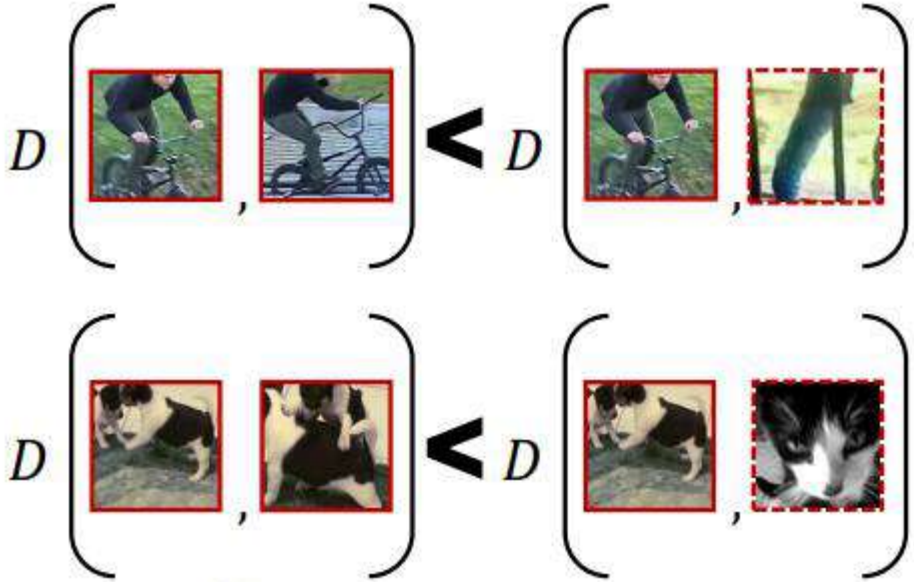
Video-Based

Tracking

Tracking Moving Objects



Tracking Moving Objects



D: Distance in deep feature space

5.2. Ranking Loss Function

Given the set of patch pairs $\mathbb{S}$ sampl
we propose to learn an image similarity
of CNN. Specifically, given an image X
network, we can obtain its feature in the final layer as $f(X)$.
Then, we define the distance of two image patches X_1, X_2
based on the cosine distance in the feature space as,

$$D(X_1, X_2) = 1 - \frac{f(X_1) \cdot f(X_2)}{\|f(X_1)\| \|f(X_2)\|}. \quad (1)$$

Xiaolong Wang & Abhinav Gupta. “Unsupervised Learning of Visual Representations using Videos.” ICCV. 2015.

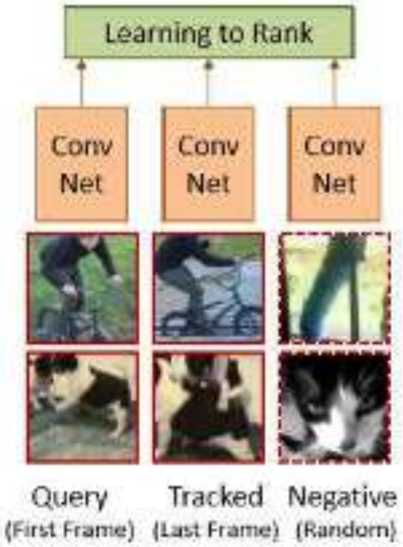
Tracking Moving Objects

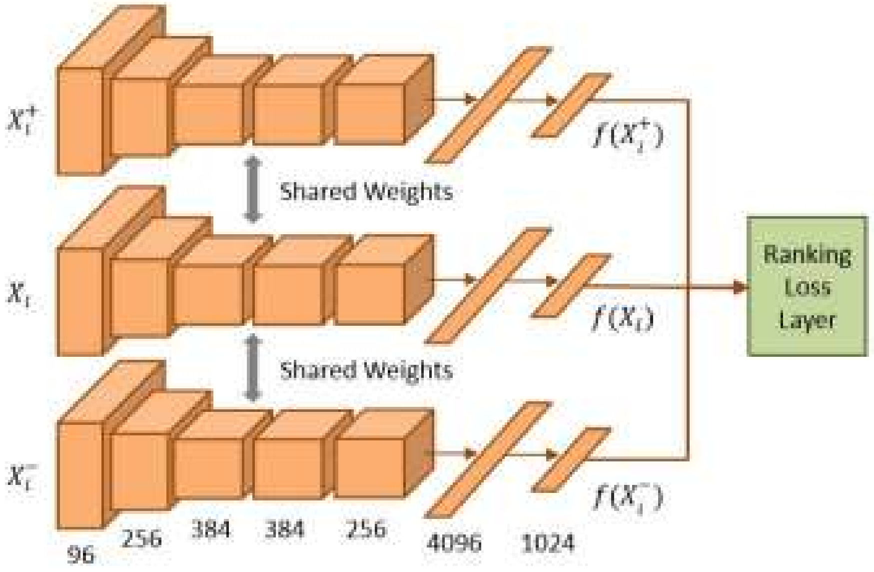
and X_i^- is a random patch from a different video, we want to enforce $D(X_i, X_i^-) > D(X_i, X_i^+)$. Therefore, the loss of our ranking model is defined by hinge loss as,

$$L(X_i, X_i^+, X_i^-) = \max\{0, D(X_i, X_i^+) - D(X_i, X_i^-) + M\}, \quad (2)$$

where M represents the gap parameters between two distances. We set $M = 0.5$ in the experiment. Then our objective function for training can be represented as,

$$\min_W \frac{\lambda}{2} \| W \|_2^2 + \sum_{i=1}^{N} \max\{0, D(X_i, X_i^+) - D(X_i, X_i^-) + M\}, \quad (3)$$





Xiaolong Wang & Abhinav Gupta. “Unsupervised Learning of Visual Representations using Videos.” ICCV. 2015.

FaceNet

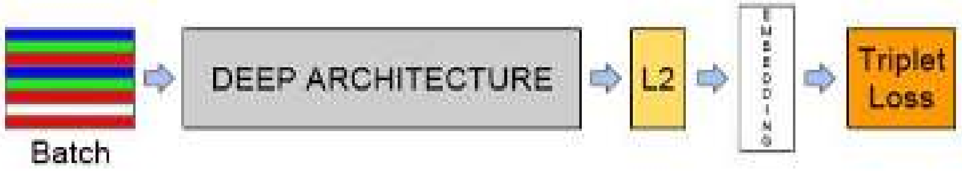


Figure 2. **Model structure.** Our network consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training.

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

3. Method

motivated in [19] in the context of nearest-neighbor classification. Here we want to ensure that an image x_i^a (*anchor*) of a specific person is closer to all other images x_i^p (*positive*) of the same person than it is to any image x_i^n (*negative*) of any other person. This is visualized in Figure 3.

Thus we want,

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2 , \tag{1}$$

$$\forall \left(f(x_i^a), f(x_i^p), f(x_i^n)\right) \in \mathcal{T} . \tag{2}$$

where α is a margin that is enforced between positive and negative pairs. $\mathcal{T}$ is the set of all possible triplets in the training set and has cardinality N.

The loss that is being minimized is then $L =$

$$\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+ . \tag{3}$$

Florian Schroff, Dmitry Kalenichenko, James Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. CVPR 2015

Tracking Moving Objects

多智体人工智能实验室
Multi-Agent Artificial Intelligence Laboratory

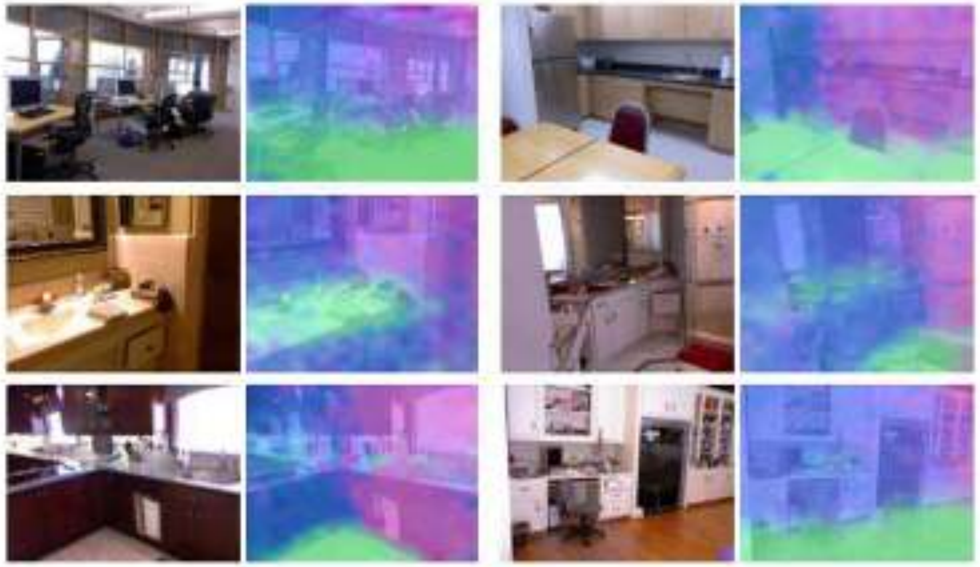


Figure 8. Surface normal estimation results on NYU dataset. For visualization, we use green for horizontal surface, blue for facing right and red for facing left, i.e., blue → X; green → Y; red → Z.

Figure 5. Top response regions for the pool5 neurons of our unsupervised-CNN. Each row shows top response of one neuron.

Xiaolong Wang & Abhinav Gupta. "Unsupervised Learning of Visual Representations using Videos." ICCV. 2015.

Video-Based

Frame Sequence

Validate Frame Order
O3N (Odd-One-Out Network)

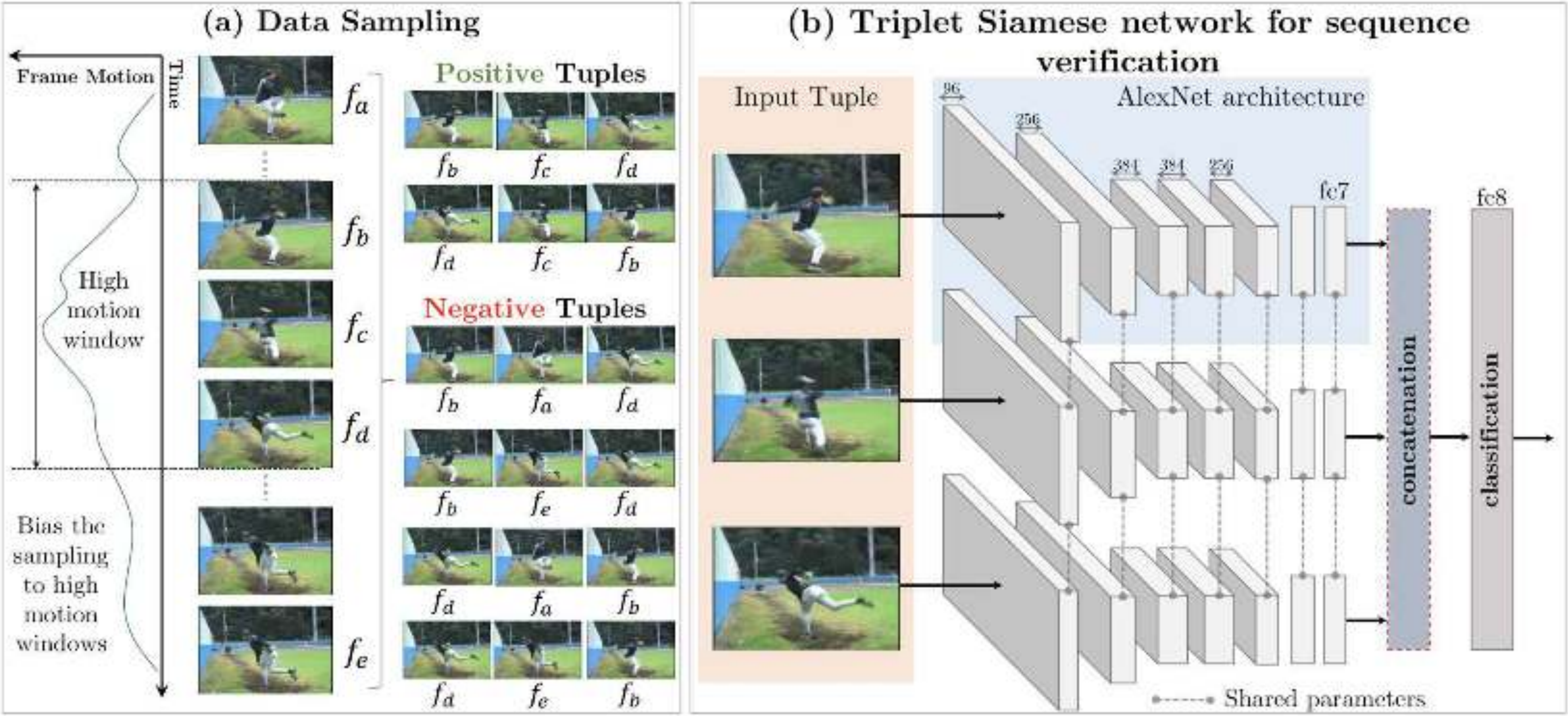
Validate Frame Order

- determine whether a sequence of frames from a video is placed in the **correct temporal order** ("temporal valid").

The training frames are sampled from high-motion windows. Every time 5 frames are sampled $(f_a, f_b, f_c, f_d, f_e)$ and the timestamps are in order $a < b < c < d < e$. Out of 5 frames, one positive tuple (f_b, f_c, f_d) and two negative tuples, (f_b, f_a, f_d) and (f_b, f_e, f_d) are created. The parameter $\tau_{\max} = |b - d|$ controls the difficulty of positive training instances (i.e. higher $\rightarrow$ harder) and the parameter $\tau_{\min} = \min(|a - b|, |d - e|)$ controls the difficulty of negatives (i.e. lower $\rightarrow$ harder).

Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. "Shuffle and learn: unsupervised learning using temporal order verification." ECCV. 2016.

Validate Frame Order



Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. "Shuffle and learn: unsupervised learning using temporal order verification." ECCV. 2016.

Validate Frame Order

Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. “Shuffle and learn: unsupervised learning using temporal order verification.” ECCV. 2016.

Validate Frame Order

Ishan Misra, C. Lawrence Zitnick, and Martial Hebert. “Shuffle and learn: unsupervised learning using temporal order verification.” ECCV. 2016.

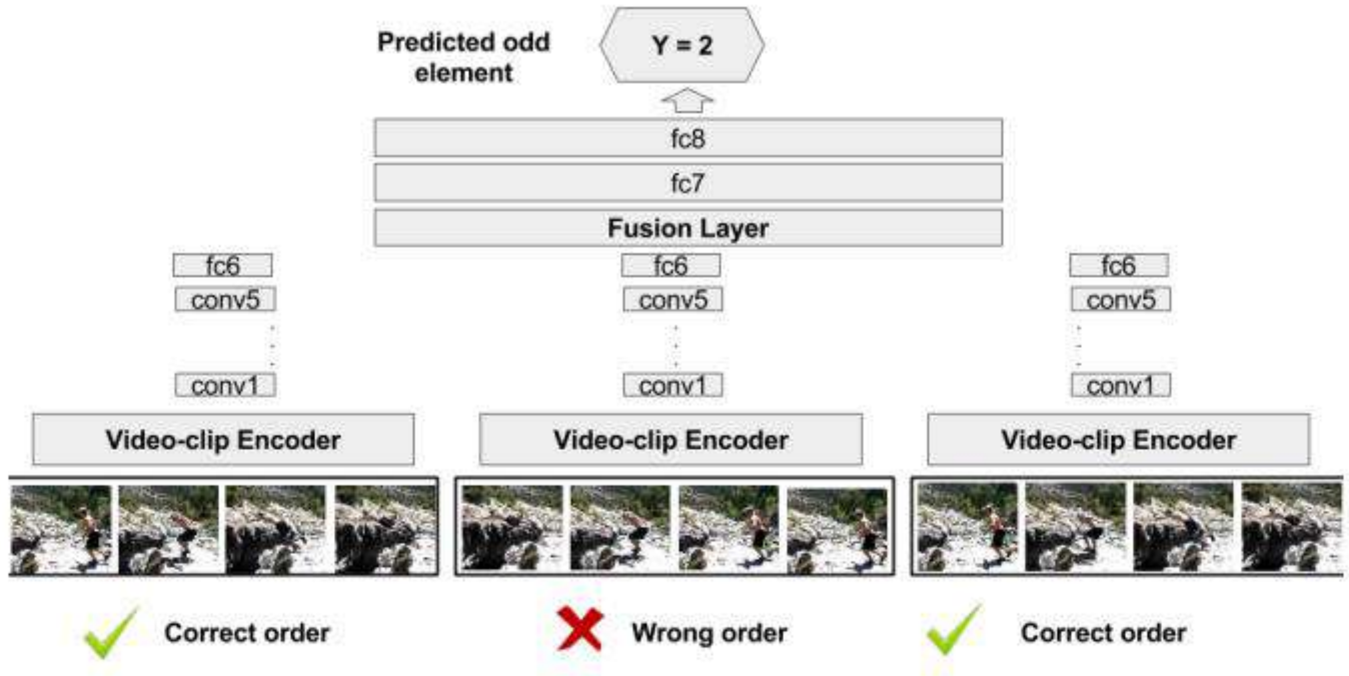
O3N (Odd-One-Out Network)

- is based on video **frame sequence validation** too.
- One step further, **pick the incorrect sequence**

- Given input video clips
- **one** of them has frames shuffled, thus in the **wrong order**,
- O3N learns to **predict** the location of the **odd video clip**.

Basura Fernando, et al. "Self-Supervised Video Representation Learning With Odd-One-Out Networks" CVPR. 2017.

O3N (Odd-One-Out Network)



Basura Fernando, et al. “Self-Supervised Video Representation Learning With Odd-One-Out Networks” CVPR. 2017.

Arrow of Time

Figure 1: Seeing these ordered frames from videos, can you tell whether each video is playing forward or backward? (answer below[1]). Depending on the video, solving the task may require (a) low-level understanding (e.g. physics), (b) high-level reasoning (e.g. semantics), or (c) familiarity with very subtle effects or with (d) camera conventions. In this work, we learn and exploit several types of knowledge to predict the arrow of time automatically with neural network models trained on large-scale video datasets.

Donglai Wei, et al. "Learning and Using the Arrow of Time" CVPR. 2018.

Arrow of Time

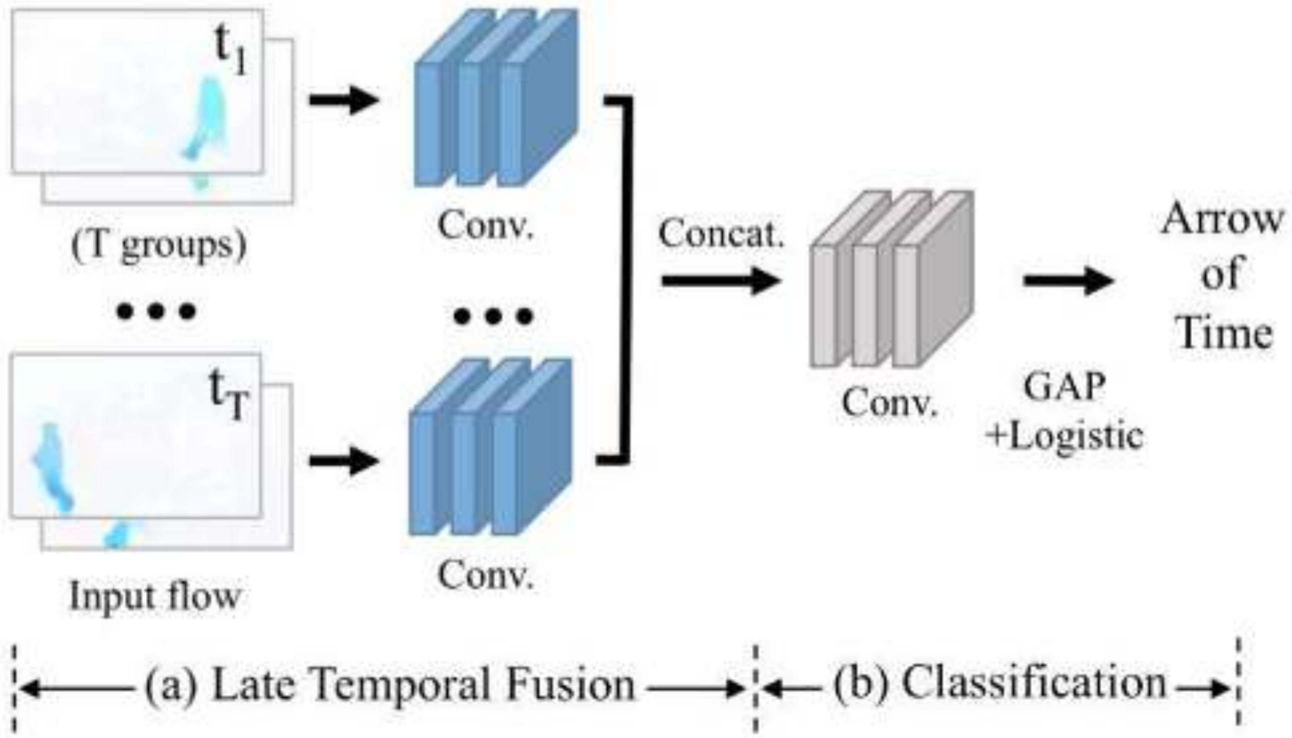


Fig. 13. Overview of learning representation by predicting the arrow of time. (a) Conv features of multiple groups of frame sequences are concatenated. (b) The top level contains 3 conv layers and average pooling. (Image source: Wei et al, 2018)

Donglai Wei, et al. "Learning and Using the Arrow of Time" CVPR. 2018.

Video-Based

Video Colorization

Video Colorization

多智体人工智能实验室
Multi-Agent Artificial Intelligence Laboratory

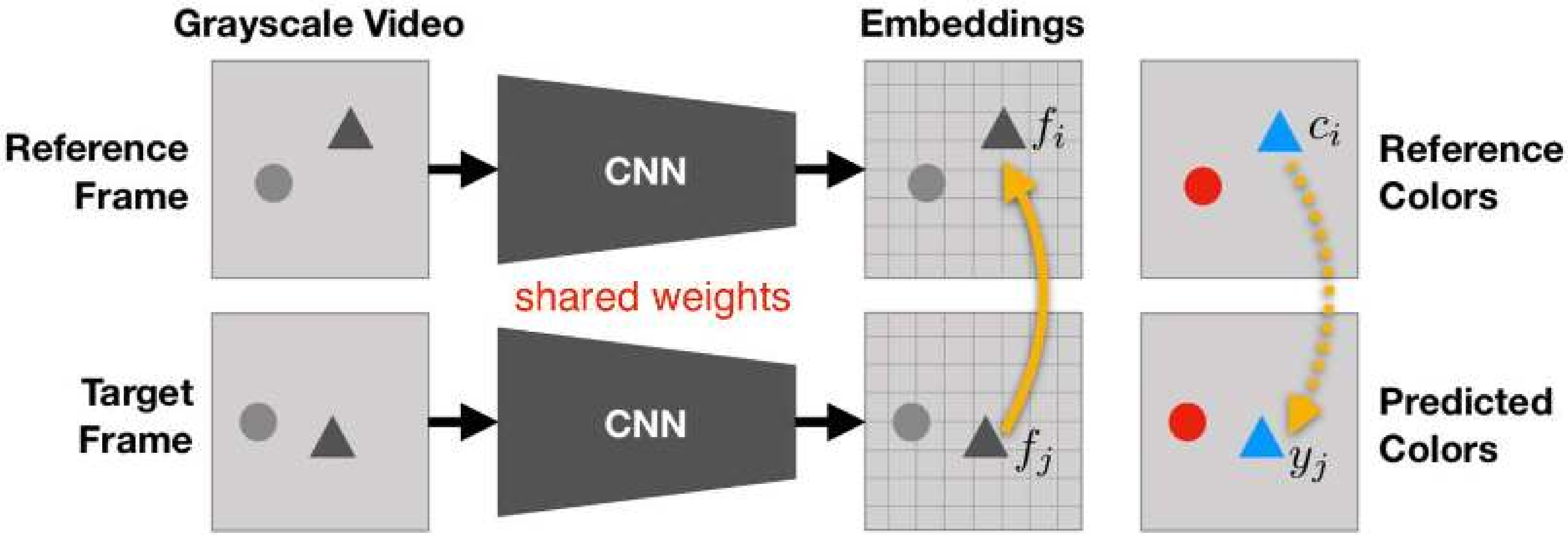
- **a rich representation** that can be used for **video segmentation** and unlabelled visual region **tracking**, *without extra fine-tuning*.

- here the task is to **copy colors** from a normal reference frame in color to another target frame in grayscale by leveraging the natural temporal coherency of colors across video frames

- the model is designed to learn to **keep track of correlated pixels** in different frames.

Carl Vondrick, et al. "Tracking Emerges by Colorizing Videos" ECCV. 2018.

Video Colorization

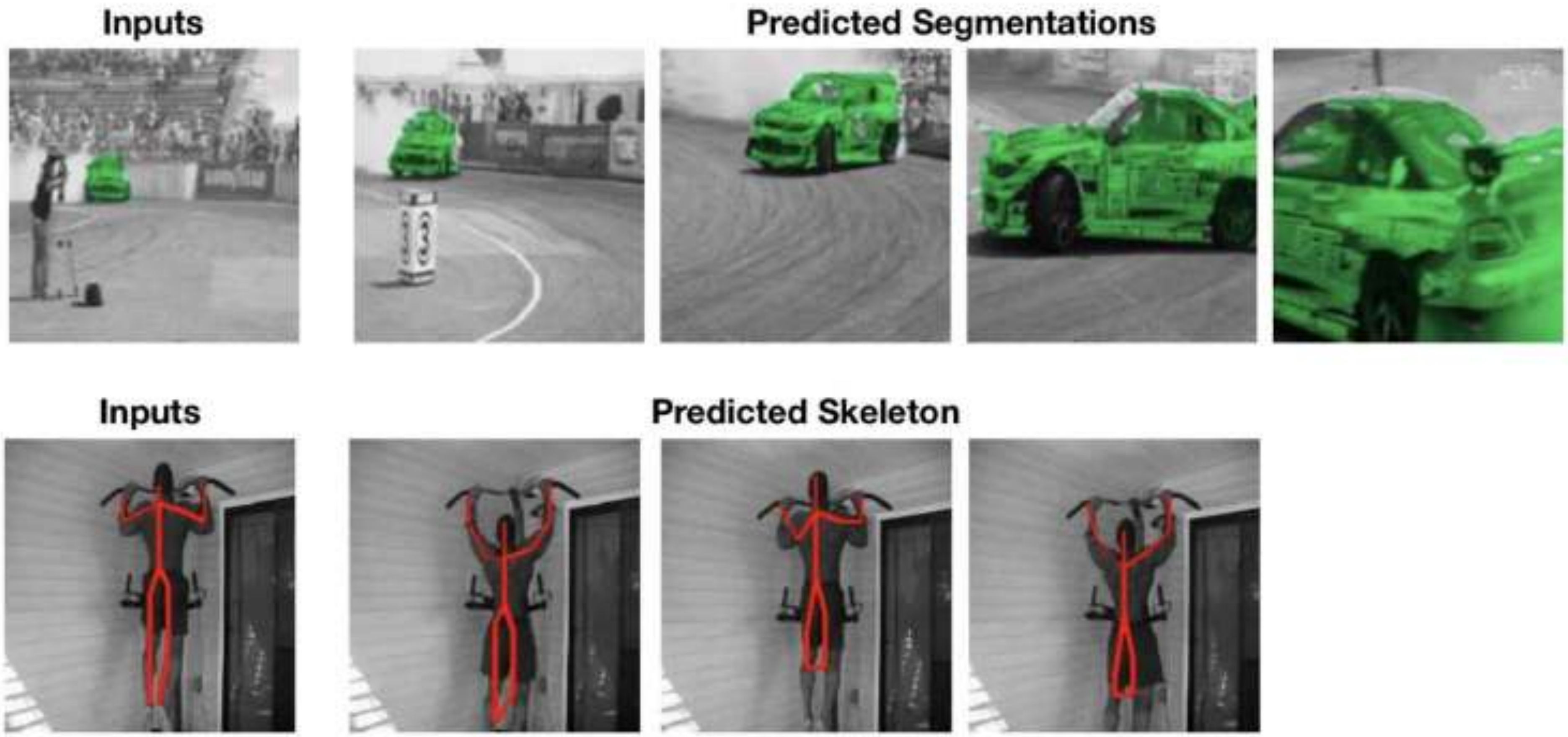


$$\hat{c}_j = \sum_i A_{ij} c_i \text{ where } A_{ij} = \frac{\exp(f_i f_j)}{\sum_{i'} \exp(f_{i'} f_j)}$$

Carl Vondrick, et al. “Tracking Emerges by Colorizing Videos” ECCV. 2018.

Video Colorization

- Use video colorization to track object segmentation and human pose in time.



Carl Vondrick, et al. “Tracking Emerges by Colorizing Videos” ECCV. 2018.

Self-Supervised Learning

Images-Based

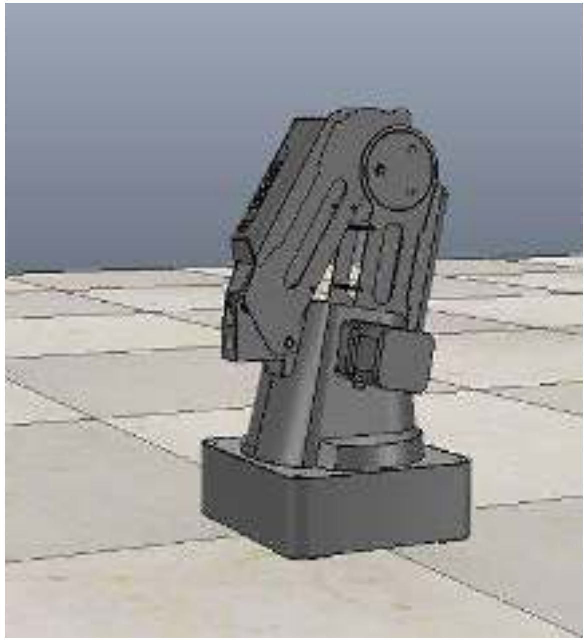
Video-Based

Control-Based

Control-Based

Multi-View Metric Learning

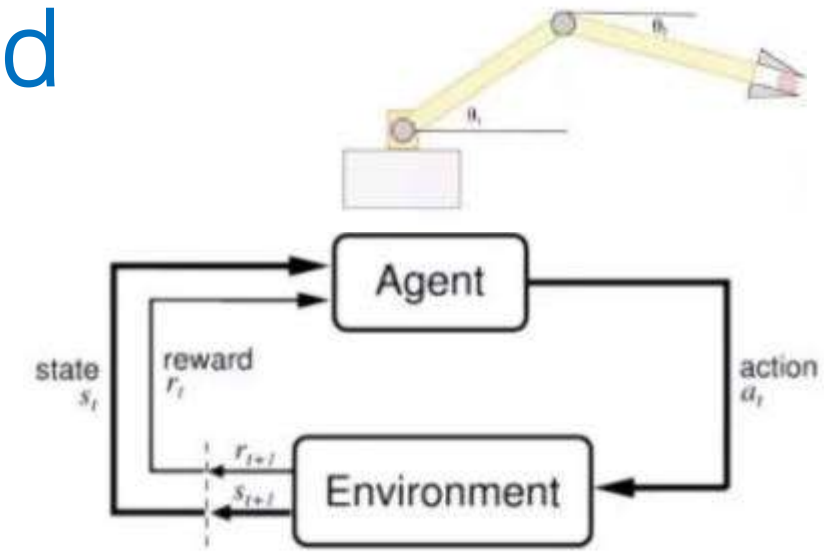
Autonomous Goal Generation



Control-Based

- Multi-View Metric Learning
- Autonomous Goal Generation

Control-Based



- The visual data has a lot of noise that is irrelevant to the true state and thus the equivalence of states **cannot be inferred from pixel-level comparison.**
- Self-supervised representation learning has shown great potential in learning useful **state embedding** that can be used directly as **input to a control policy**.

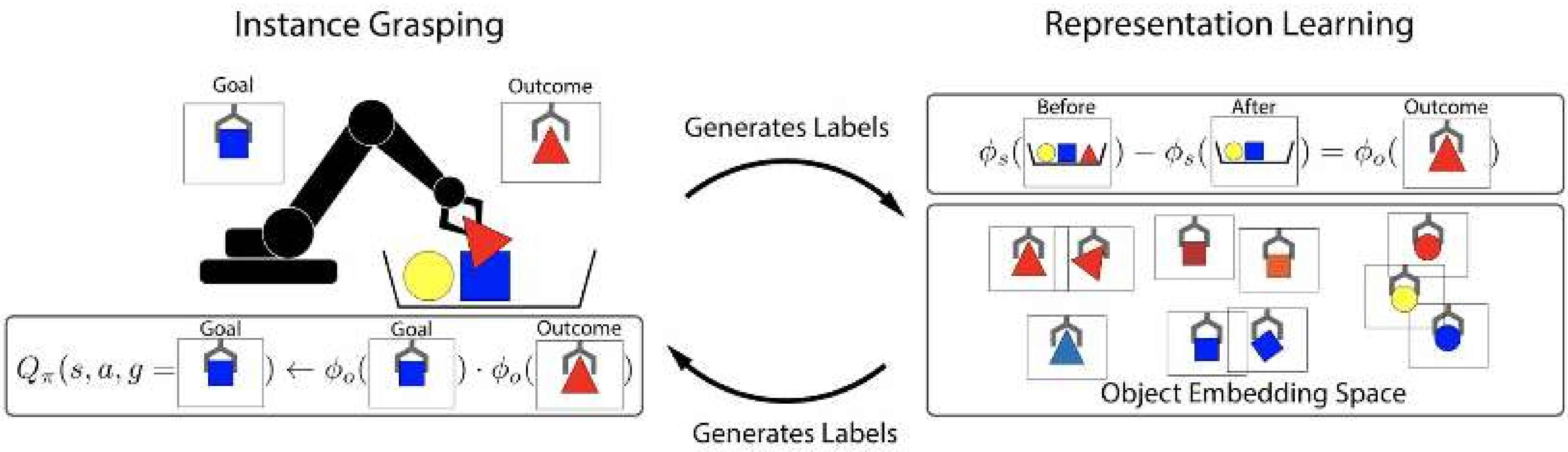
Multi-View Metric Learning

Grasp2Vec
TCN (Time-Contrastive Networks)

Multi-View Metric Learning

The concept of metric learning has been mentioned multiple times in the previous sections. A common setting is: Given a triple of samples, (*anchor* s_a, *positive* sample s_p, *negative* sample s_n), the learned representation embedding $\phi(s)$ fulfills that s_a stays close to s_p but far away from s_n in the latent space.

Grasp2Vec



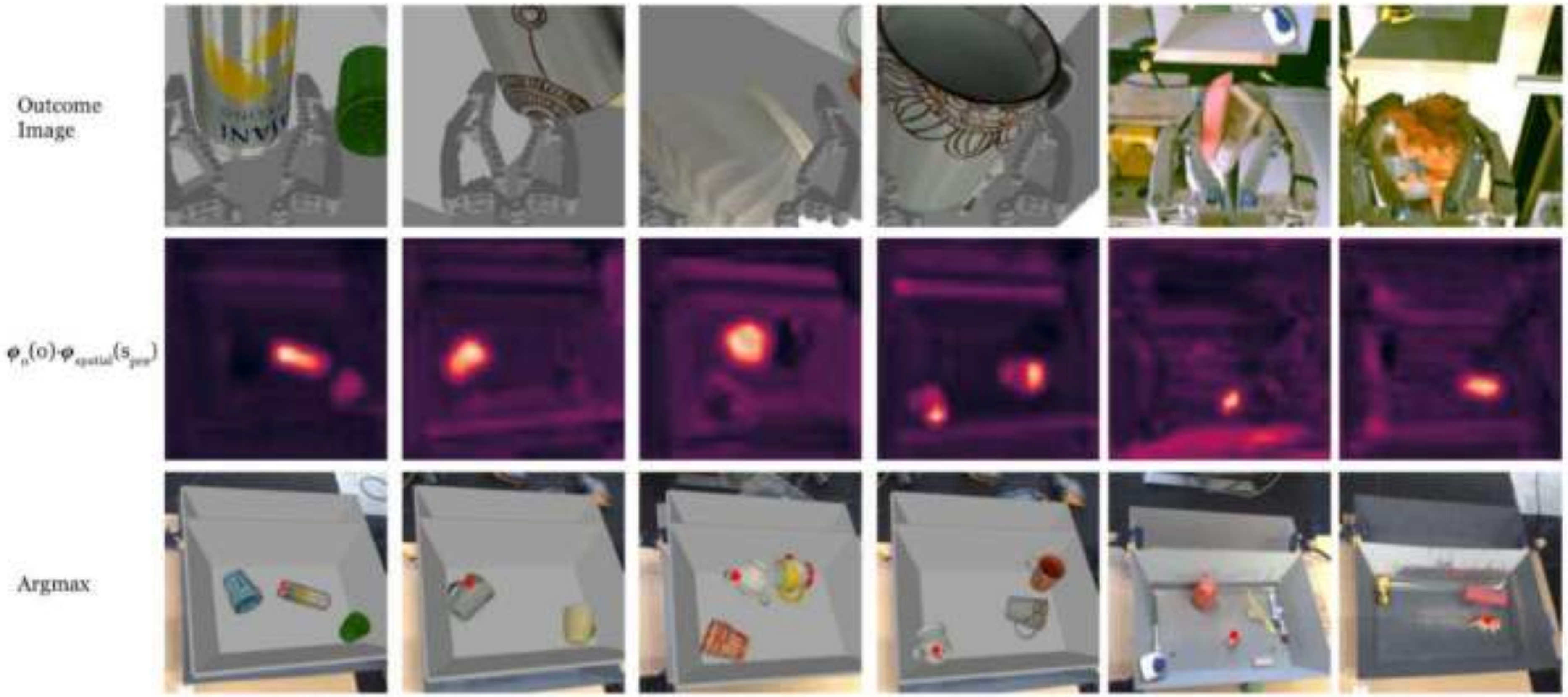
$$\mathscr{L}_{\text{Grasp2Vec}} = \text{NPairs}((\phi_s(s_{\text{pre}}) - \phi_s(s_{\text{post}})), \phi_o(\mathbf{o})) + \text{NPairs}(\phi_o(\mathbf{o}), (\phi_s(s_{\text{pre}}) - \phi_s(s_{\text{post}}))).$$

$$\text{NPairs}(a,p) = \sum_{i<B} -\log\left(\frac{e^{a_i{}^\top p_i}}{\sum_{j<B} e^{a_i,p_j}}\right) + \lambda(||a_i||_2^2 + ||p_i||_2^2).$$

Eric Jang & Coline Devin, et al. “Grasp2Vec: Learning Object Representations from Self-Supervised Grasping” CoRL. 2018.

Grasp2Vec

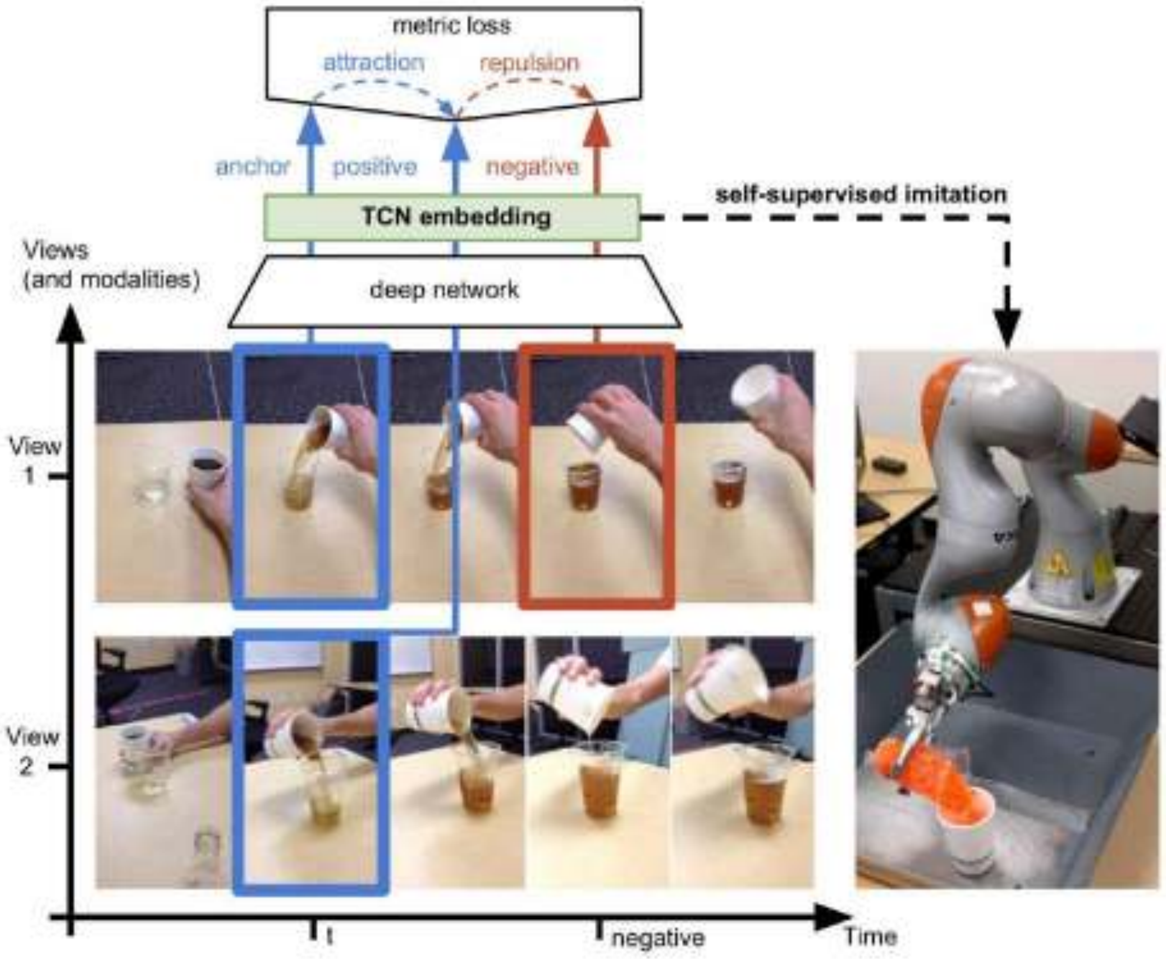
$r = \phi_o(g) \cdot \phi_o(o)$. Note that computing rewards only relies on the learned latent space and doesn't involve ground truth positions, so it can be used for training on real robots.



Eric Jang & Coline Devin, et al. "Grasp2Vec: Learning Object Representations from Self-Supervised Grasping" CoRL. 2018.

TCN (Time-Contrastive Networks)

多智体人工智能实验室
Multi-Agent Artificial Intelligence Laboratory

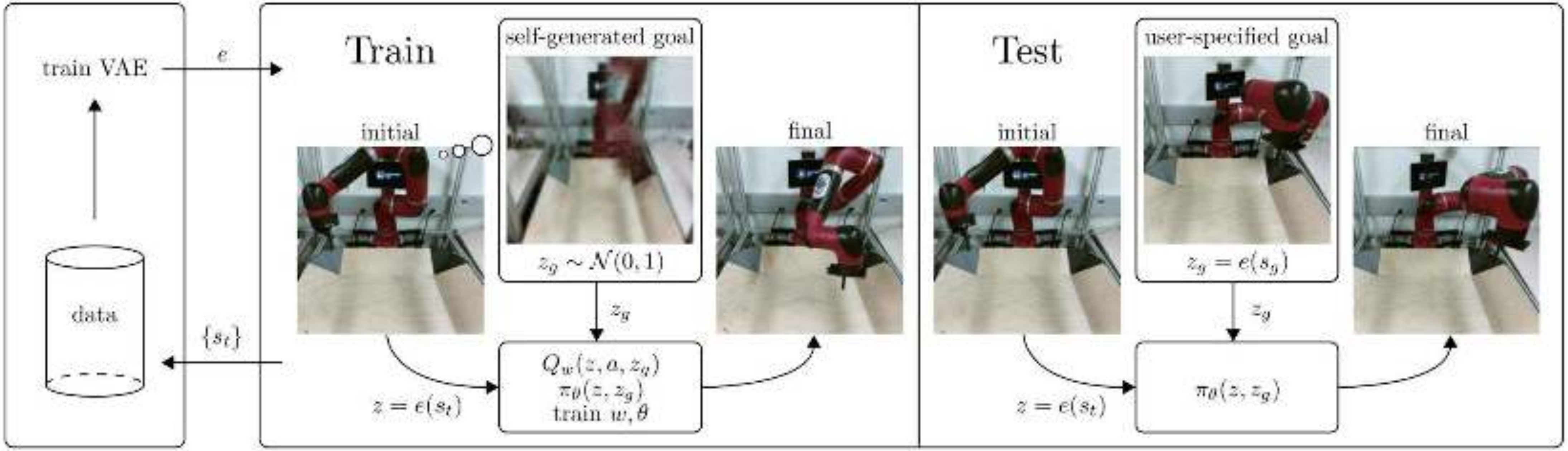


Pierre Sermanet, et al. “Time-Contrastive Networks: Self-Supervised Learning from Video” CVPR. 2018.

Autonomous Goal Generation

RIG (Reinforcement learning with Imagined Goals)

RIG (RL with Imagined Goals)



Ashvin Nair, et al. "Visual reinforcement learning with imagined goals" NeuriPS. 2018.

RIG (RL with Imagined Goals)

多智体人工智能实验室
Multi-Agent Artificial Intelligence Laboratory

Let's say a β-VAE has an encoder q_ϕ mapping input states to latent variable z which i
a Gaussian distribution and a decoder p_ψ mapping z back to the states. The state en
is set to be the mean of β-VAE encoder.

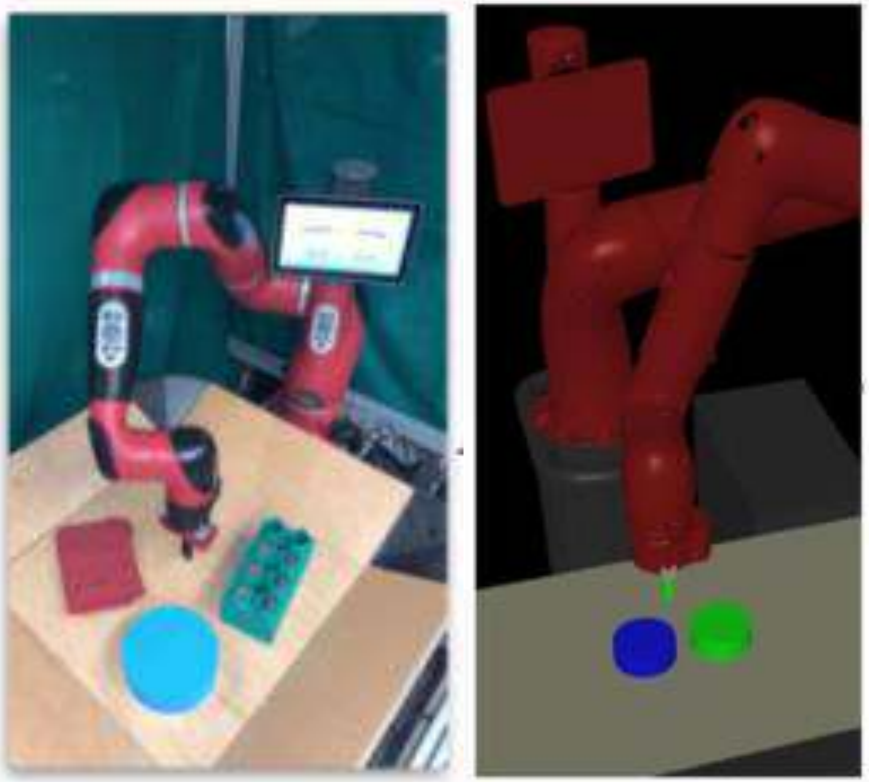
$$
\begin{aligned}
z &\sim q_\phi(z|s) = \mathcal{N}(z; \mu_\phi(s), \sigma^2_\phi(s)) \\
\mathcal{L}_{\beta\text{-VAE}} &= -\mathbb{E}_{z\sim q_\phi(z|s)}[\log p_\psi(s|z)] + \beta D_{\text{KL}}(q_\phi(z|s) \| p_\psi(s)) \\
e(s) &\triangleq \mu_\phi(s)
\end{aligned}
$$

$$
r(s, g) = -\|e(s) - e(g)\|
$$

Ashvin Nair, et al. "Visual reinforcement learning with imagined goals" NeuriPS. 2018.

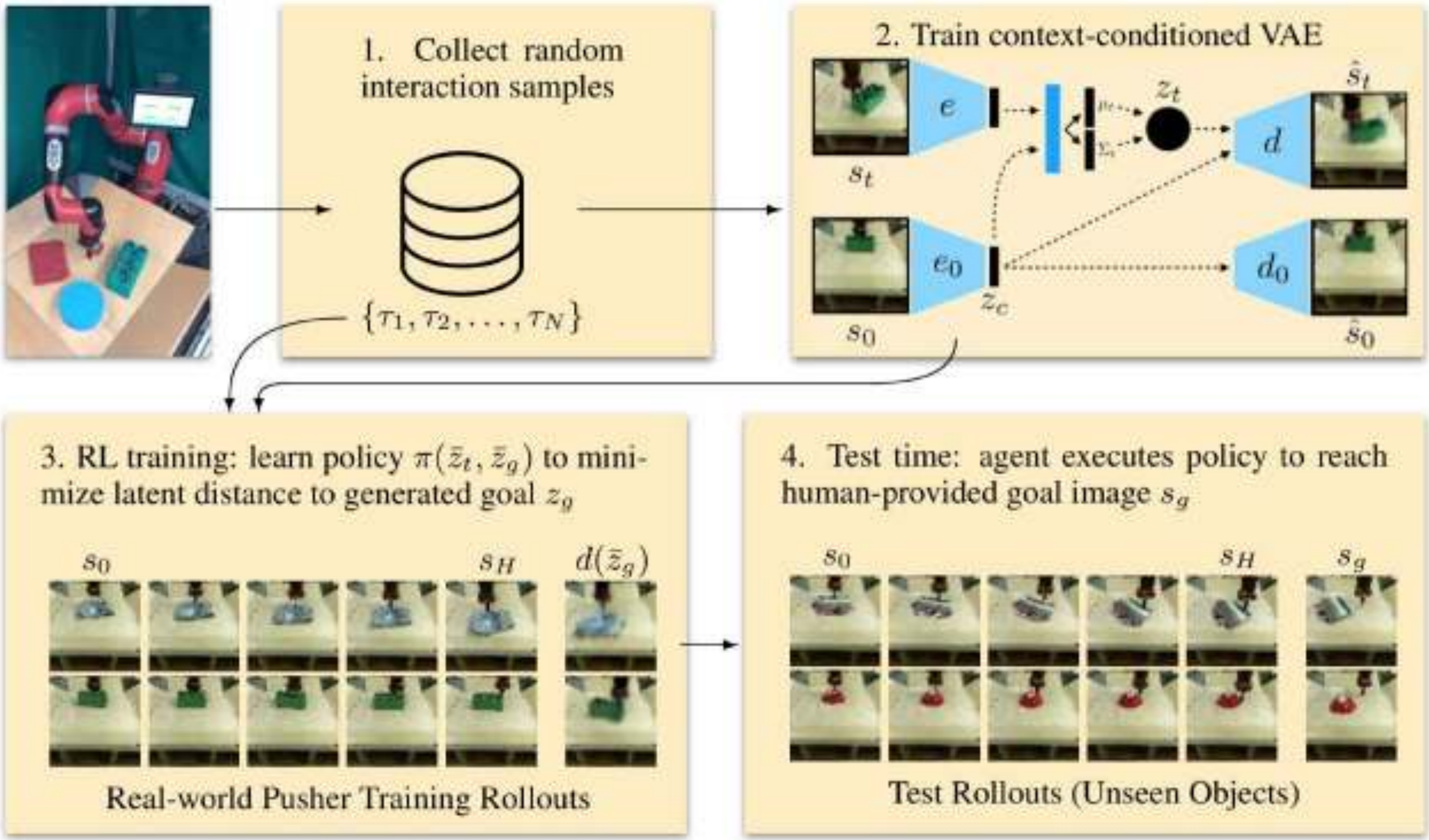
CC-VAE (Context-Conditioned VAE)

- The problem with RIG is a **lack of object variations** in the imagined goal pictures. If β-VAE is only trained with **a black puck**,
- it would not be able to create a goal with other objects like **blocks of different shapes and colors**. A follow-up improvement replaces β-VAE with a **CC-VAE**



Ashvin Nair, et al. “Contextual imagined goals for self-supervised robotic learning” CoRL. 2019.

CC-VAE (Context-Conditioned VAE)



Ashvin Nair, et al. “Contextual imagined goals for self-supervised robotic learning” CoRL. 2019.

CC-VAE (Context-Conditioned VAE)

encoder and decoder parameters ϕ and ψ, we jointly optimize both objectives when minimizing the negative evidence lower bound:

$$\mathcal{L}_{\text{VAE}} = -\mathbb{E}_{q_\phi(z|s)}[\log p(s|z)] + \beta D_{KL}(q_\phi(z|s)||p(z)). \tag{2}$$

3.3 Conditional Variational Auto-Encoders

Instead of a generative model that learns to generate the dataset distribution, one might instead desire a more structured generative model that can generate samples based on structured input. One example of this is a conditional variational auto-encoder (CVAE) that conditions the output on some input variable c and samples from $p(x|c)$ [40]. For example, to train a model that generates images of digits given the desired digit, the input variable c might be a one-hot encoded vector of the desired digit.

A CVAE trains $q_\phi(z|s, c)$ and $q_\psi(s|z, c)$, where both the encoder and decoder has access to the input variable c. The CVAE then minimizes:

$$\mathcal{L}_{\text{CVAE}} = -\mathbb{E}_{q_\phi(z|s,c)}[\log p(s|z, c)] + \beta D_{KL}(q_\phi(z|s, c)||p(z)). \tag{3}$$

Samples are generated by first sampling a latent $z \sim p(z)$. Based on c, we can then decode z with $q_\psi(s|z, c)$ and visualize the output, which is in our case an image. In our framework $c = s_0$.

Ashvin Nair, et al. “Contextual imagined goals for self-supervised robotic learning” CoRL. 2019.

CC-VAE (Context-Conditioned VAE)

Other than the state encoder $e(s) \triangleq \mu_\phi(s)$, CC-VAE trains a second convolutional encoder $e_0(.)$ to translate the starting state s_0 into a compact context representation $c = e_0(s_0)$. Two encoders, $e(.)$ and $e_0(.)$, are intentionally different without shared weights, as they are expected to encode different factors of image variation. In addition to the loss function of CVAE, CC-VAE adds an extra term to learn to reconstruct c back to s_0, $\hat{s}_0 = d_0(c)$.

$$\mathcal{L}_{\text{CC-VAE}} = \mathcal{L}_{\text{CVAE}} + \log p(s_0|c)$$

Ashvin Nair, et al. “Contextual imagined goals for self-supervised robotic learning” CoRL. 2019.

Common Observations

- Combining multiple pretext tasks improves performance;
- Deeper networks improve the quality of representation;
- Supervised learning baselines still beat all of them by far.

Thank you

- End