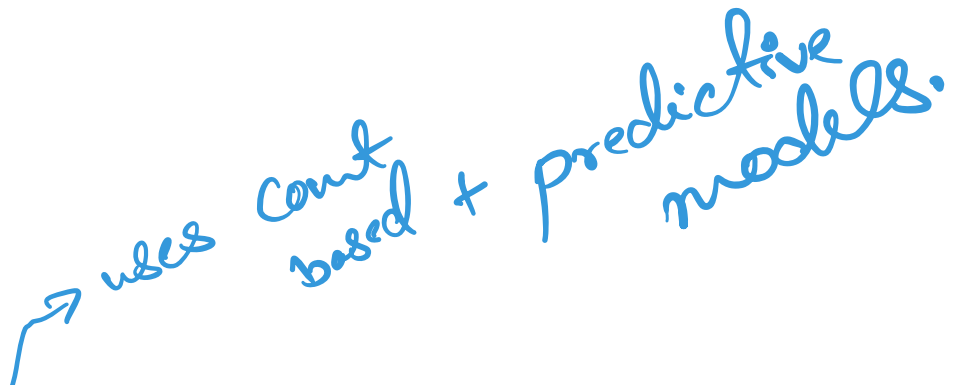
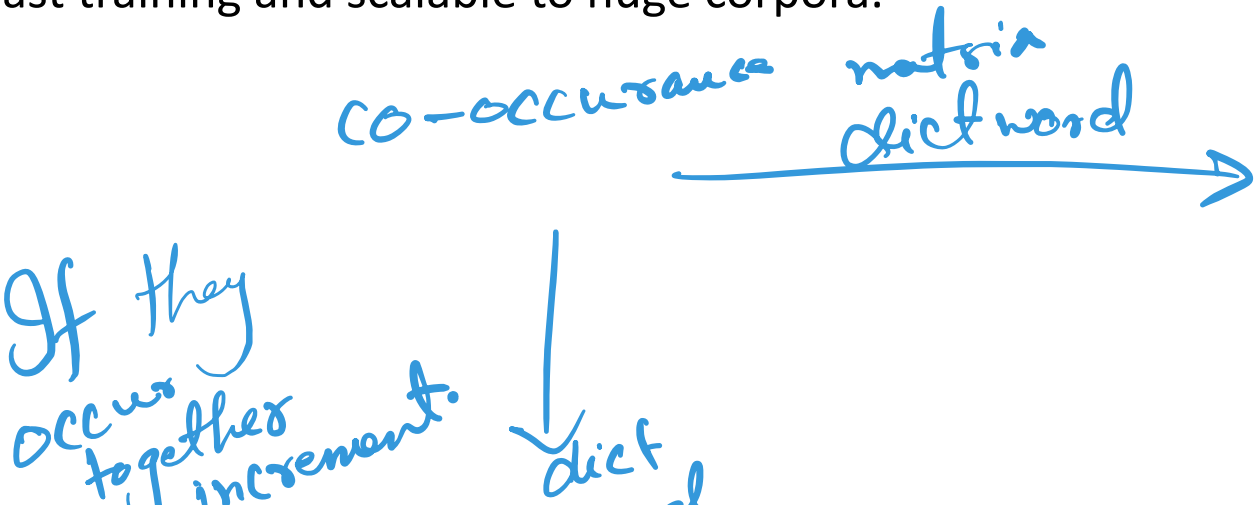
GloVe: Global Vectors for Word Representation



GloVe: **Global Vectors for Word Representation**

→ mix of predictive & count based models.

- **Predictive model** word2Vec learn vectors to improve the predictive ability.
- **Count-based models** learn vectors by doing dimensionality reduction on a co-occurrence counts matrix.
  • co-occurence matrix.
- GloVe is a best of both worlds.
- GloVe is Fast training and scalable to huge corpora.



GloVe: **Global Vectors for Word Representation**

- X: the matrix of word-word co-occurrence
- X_{ij} : number of times word j occurs in the context of word i

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

GloVe: **Global Vectors for Word Representation**

- X: the matrix of word-word co-occurrence
- X_{ij} : number of times word j occurs in the context of word i
- $X_i = \sum_k X_{ik}$: the number of times any word appears in the context of the word i

$P(X$

$X =$

	I	*like*	*enjoy*	*deep*	*learning*	*NLP*	*flying*	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

GloVe: **Global Vectors for Word Representation**

- X: the matrix of word-word co-occurrence
- X_{ij} : number of times word j occurs in the context of word i
- $X_i = \sum_k X_{ik}$: the number of times any word appears in the context of the word i
- $P_{ij} = P(j \mid i) = X_{ij}/X_i$: the probability that word j appears in the context of word i

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

GloVe: **Global Vectors for Word Representation**

- X_{ij} encodes important global information about the co-occurrence between i and j (global: because it is computed from the entire corpus)
- Why not learn word vectors which are faithful to this information?

$$v_i^T v_j = \log P(j|i)$$

$$= X_{ij}/X_i$$

GloVe: **Global Vectors for Word Representation**

- X_{ij} encodes important global information about the co-occurrence between i and j (global: because it is computed from the entire corpus)
- Why not learn word vectors which are faithful to this information?
- For example, enforce

$$v_i^T v_j = \log P(j \mid i)$$
$$= \log X_{ij} - \log (X_i)$$

$v_i^T v_j = \log P(j|i)$

GloVe: **Global Vectors for Word Representation**

- X_{ij} encodes important global information about the co-occurrence between i and j (global: because it is computed from the entire corpus)
- Why not learn word vectors which are faithful to this information?
- For example, enforce

$$\begin{aligned} v_i^T v_j &= \log P(j \mid i) \\ &= \log X_{ij} - \log (X_i) \end{aligned}$$

- Similarly,

$$v_j^T v_i = \log X_{ij} - \log X_j \quad (X_{ij} = X_{ji})$$

- Essentially we are saying that we want word vectors v_i and v_j such that $v_i^T v_j$ is faithful to the globally computed $P(j \mid i)$

GloVe: **Global Vectors for Word Representation**

- Adding the two equations we get

$$2v_i^T v_j = 2\log X_{ij} - \log X_i - \log X_j$$

$$v_i^T v_j = \log X_{ij} - \frac{1}{2}\log X_i - \frac{1}{2}\log X_j$$

GloVe: **Global Vectors for Word Representation**

- Adding the two equations we get

$$2v_i^T v_j = 2\log X_{ij} - \log X_i - \log X_j$$
$$v_i^T v_j = \log X_{ij} - \frac{1}{2}\log X_i - \frac{1}{2}\log X_j$$

- Note that $\log X_i$ and $\log X_j$ depend only on the words $i\&j$ and we can think of them as word specific biases which will be learned

$$v_i^T v_j = \log X_{ij} - b_i - b_j$$
$$v_i' v_j + b_i + b_j = \log X_{ij}$$

GloVe: **Global Vectors for Word Representation**

- We can then formulate this as the following optimization problem

$$\min_{v_i, v_j, b_i, b_j} \sum_{i,j} \Big(\underbrace{v_i^T v_j + b_i + b_j}_{\text{predicted values using model parameters}} - \underbrace{\log X_{ij}}_{\text{actual values computed from the given corpus}} \Big)^2$$

Glove results

Nearest words to frog:

1. frogs
2. toad
3. litoria
4. leptodactylidae
5. rana
6. lizard
7. eleutherodactylus

litoria

leptodactylidae

rana

eleutherodactylus

Encoder Decoder Network

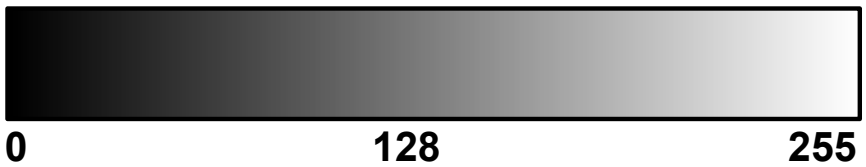
Distribution of Natural Images

https://timesofindia.indiatimes.com

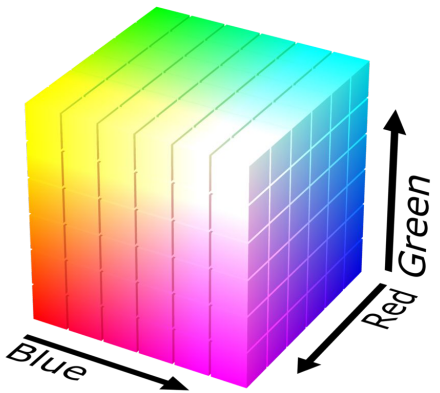
https://www.wikiwand.com/en/RGB_color_model

What is an image?

An image is a 2D grid of pixels



Each pixel can take 256 different intensity values



Each pixel consists of three color channels **Red**, **Green** and **Blue**

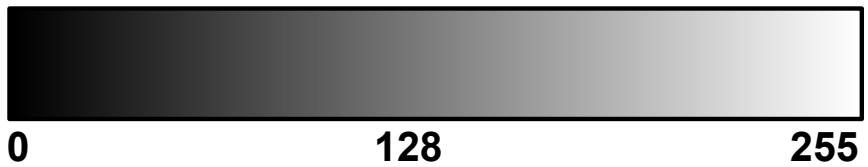
Distribution of Natural Images

https://timesofindia.indiatimes.com

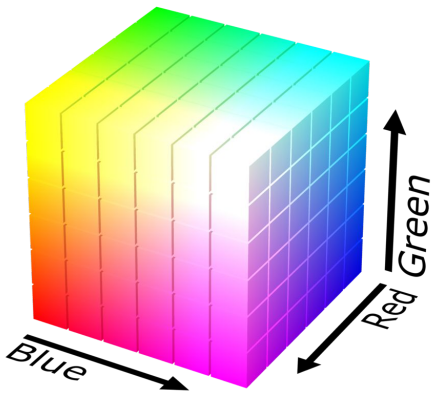
https://www.wikiwand.com/en/RGB_color_model

What is an image?

An image is a 2D grid of pixels



Each pixel can take 256 different intensity values



Each pixel consists of three color channels **Red**, **Green** and **Blue**

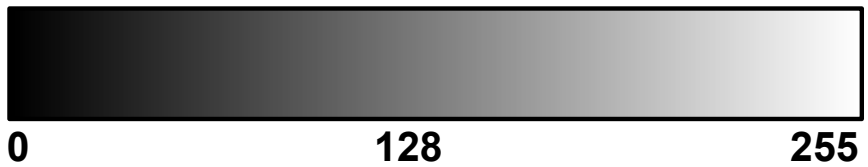
Distribution of Natural Images

https://timesofindia.indiatimes.com

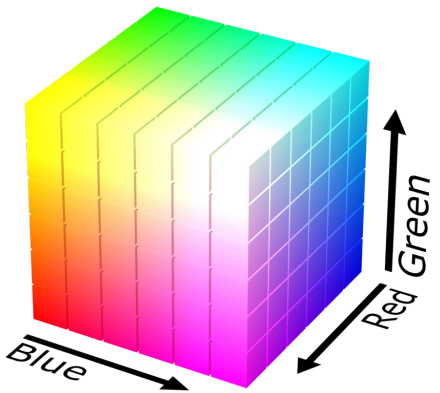
https://www.wikiwand.com/en/RGB_color_model

What is an image?

An image is a 2D grid of pixels



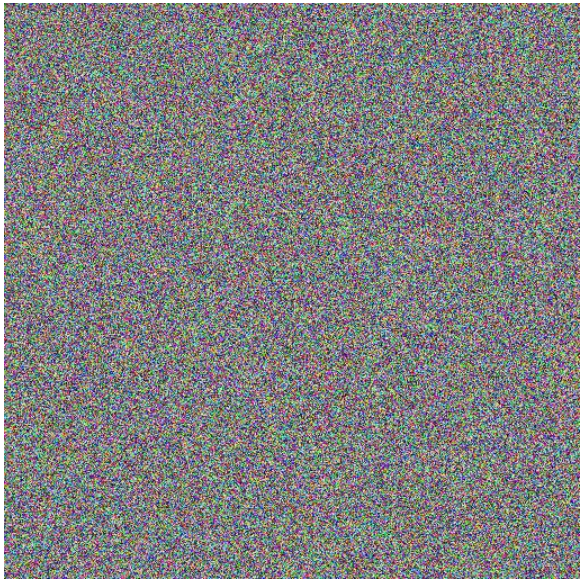
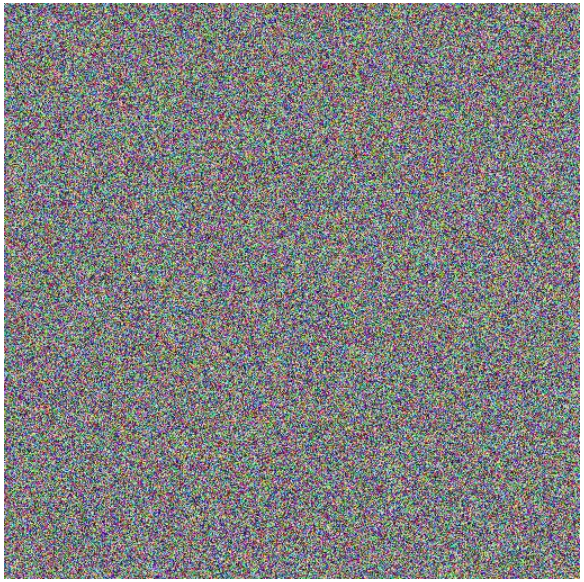
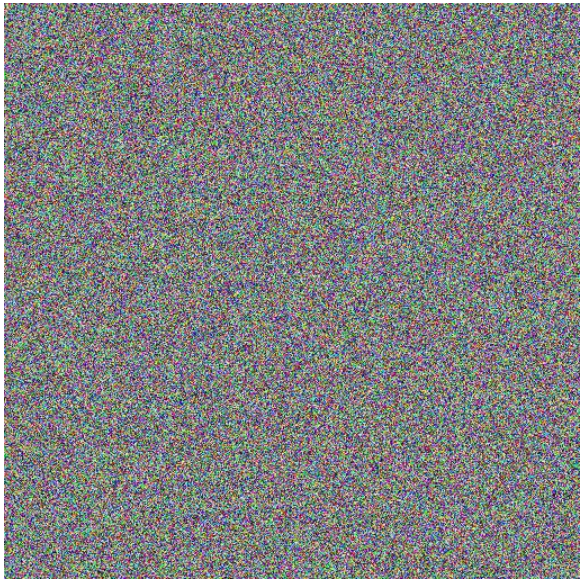
Each pixel can take 256 different intensity values



Each pixel consists of three color channels **Red**, **Green** and **Blue**

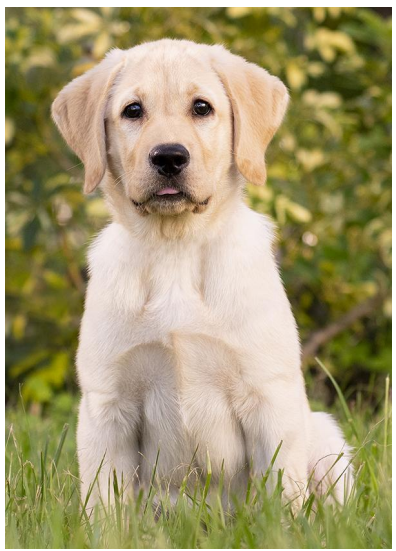
Distribution of Natural Images

Three randomly sampled images from the space of all images of size (512 X 512)

Distribution of Natural Images

But the same space also contains these natural images

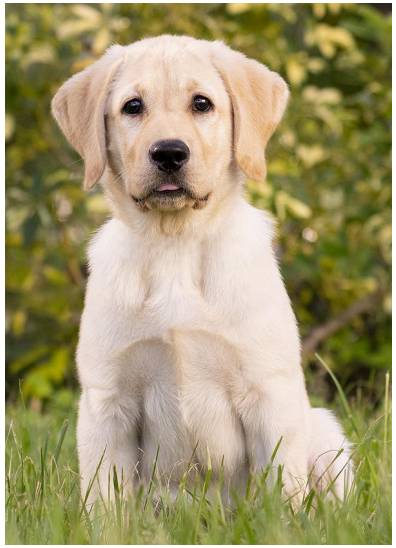


https://guidedogs.org/

https://www.thescottishsun.co.uk/news/3608326/when-is-international-mountain-day-2018-and-whats-the-theme-this-year/

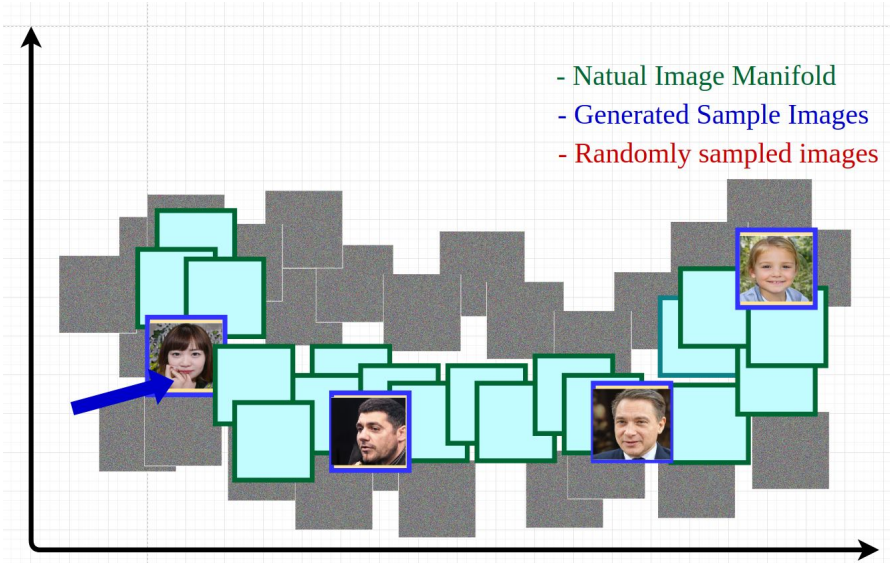
Distribution of Natural Images

Natural Images occupy an extremely tiny fraction in the space of all images

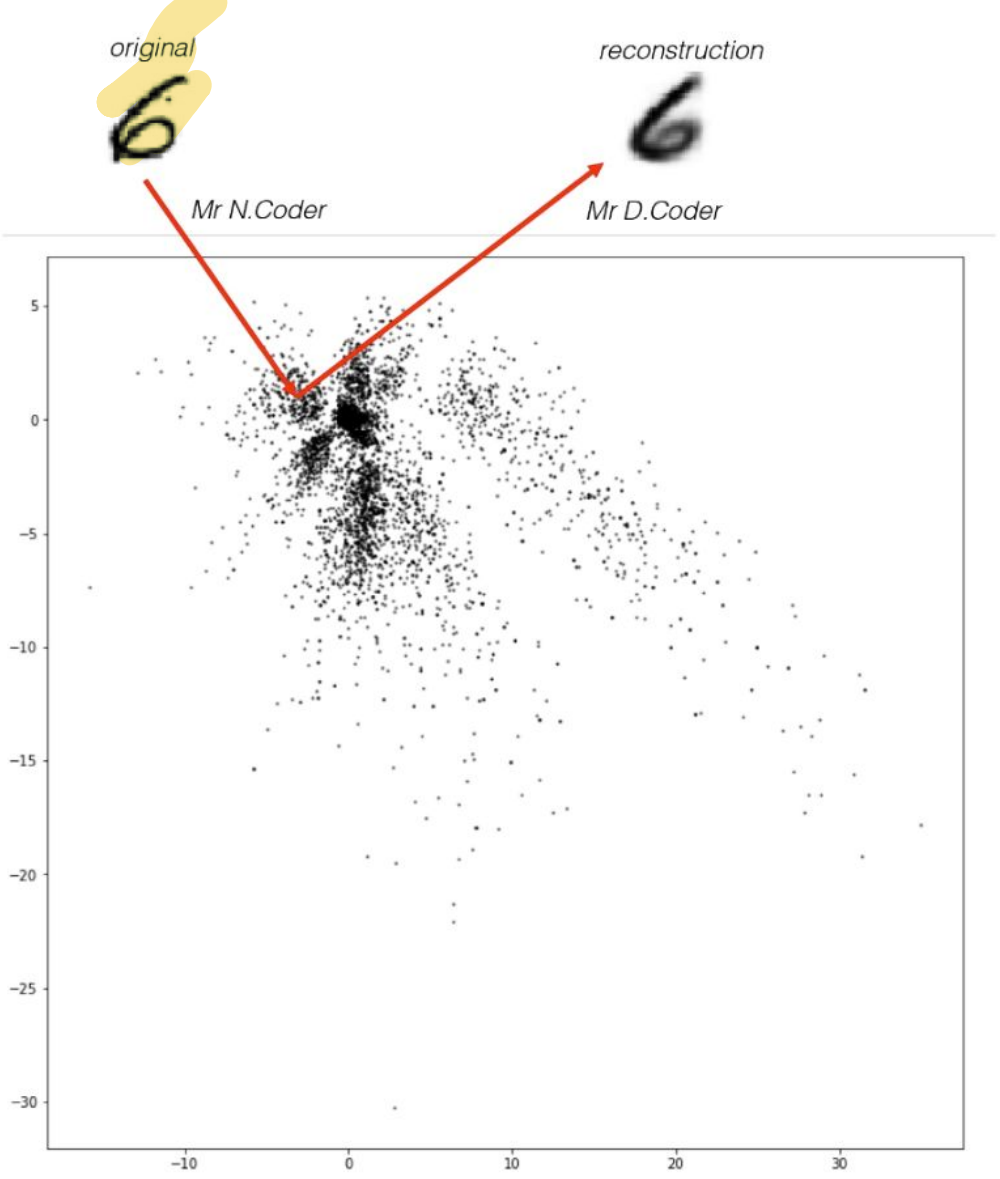


https://guidedogs.org/

https://www.thescottishsun.co.uk/news/3608326/when-is-international-mountain-day-2018-and-whats-the-theme-this-year/

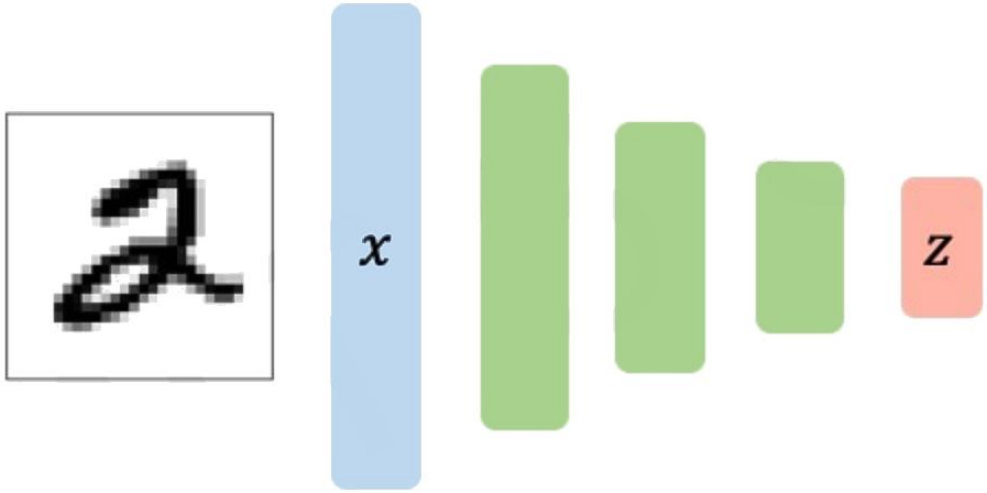


The Art Exhibition



Autoencoder -> Unsupervised.

Unsupervised approach for learning a **lower-dimensional** feature representation from unlabeled training data

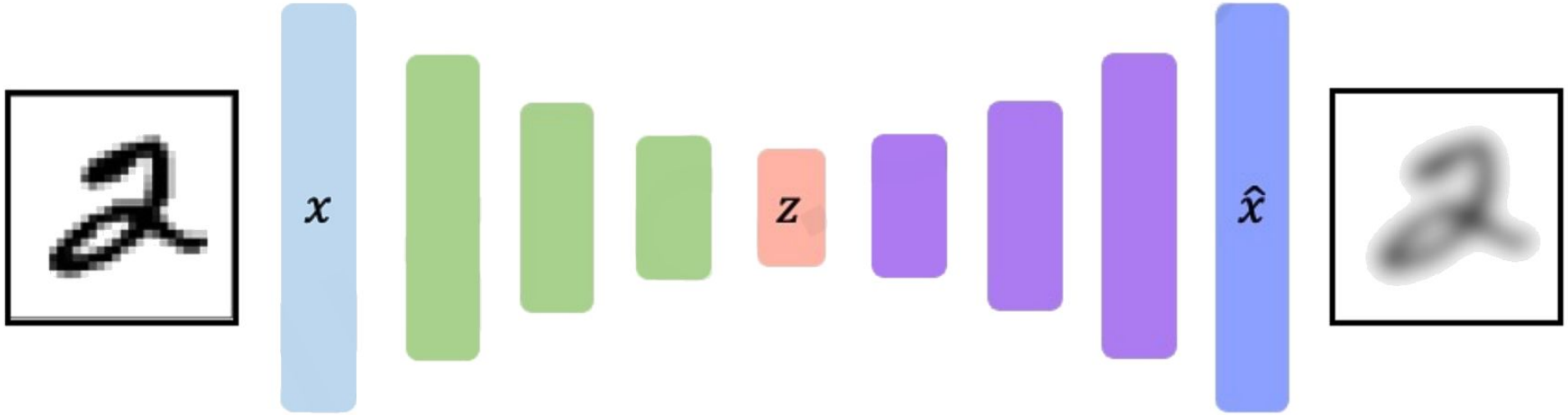


Why do we care about a low-dimensional z?

"Encoder" learns mapping from the data, x, to a low-dimensional latent space, z

Autoencoder

How can we learn this latent space?
Train the model to use these features to **reconstruct the original data**

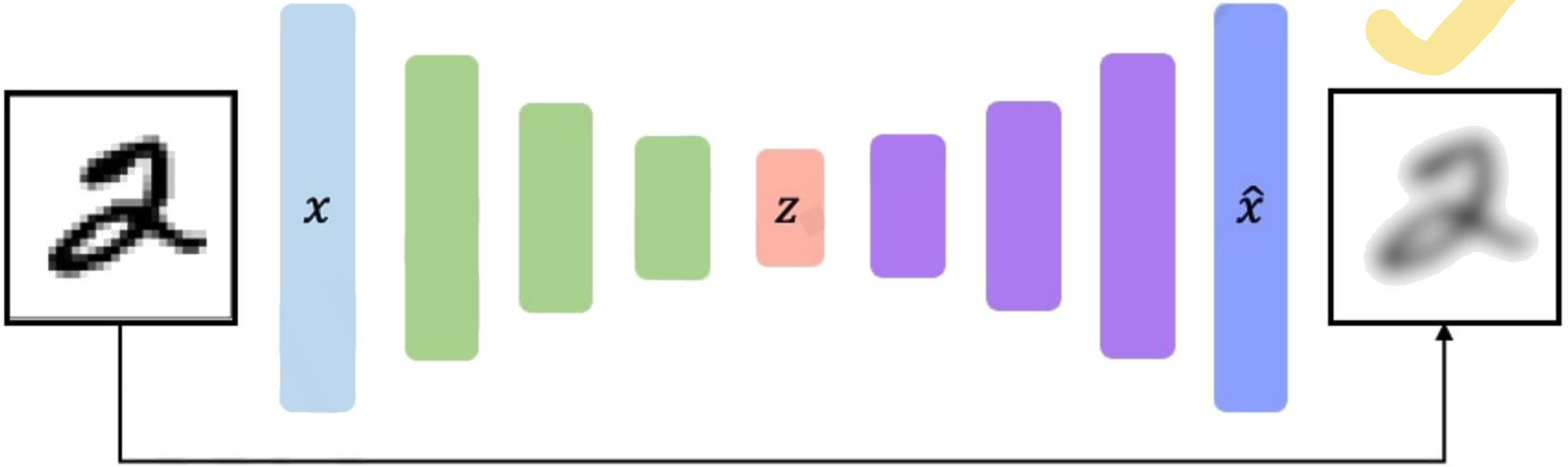


"Decoder" learns mapping back from latent space, z,
to a reconstructed observation, $\hat{x}$

Autoencoder

How can we learn this latent space?

Train the model to use these features to **reconstruct the original data**

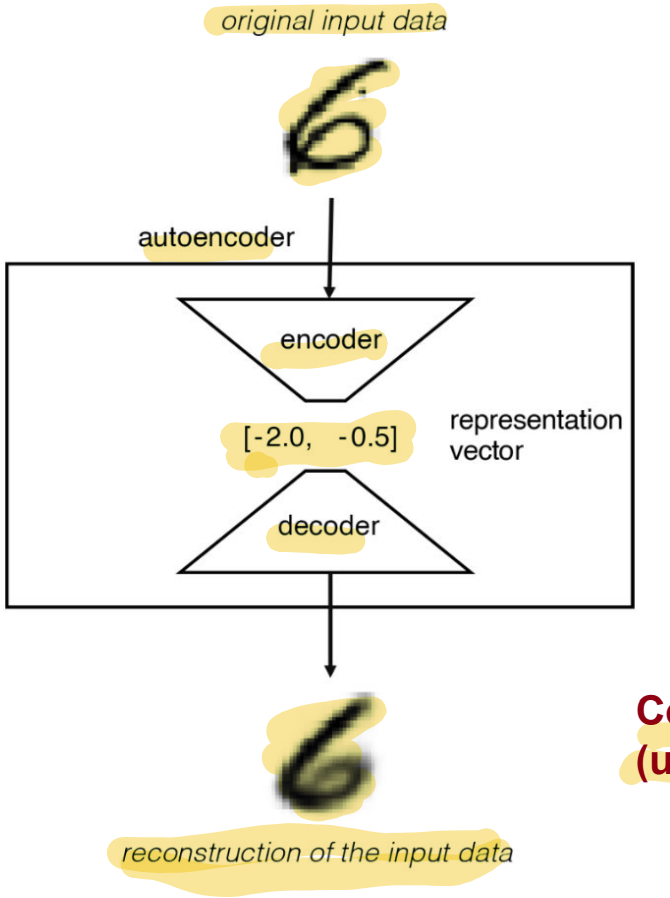


$$\mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2$$

Loss function doesn't use any labels!!

Encoder Architecture

Convolution Layers

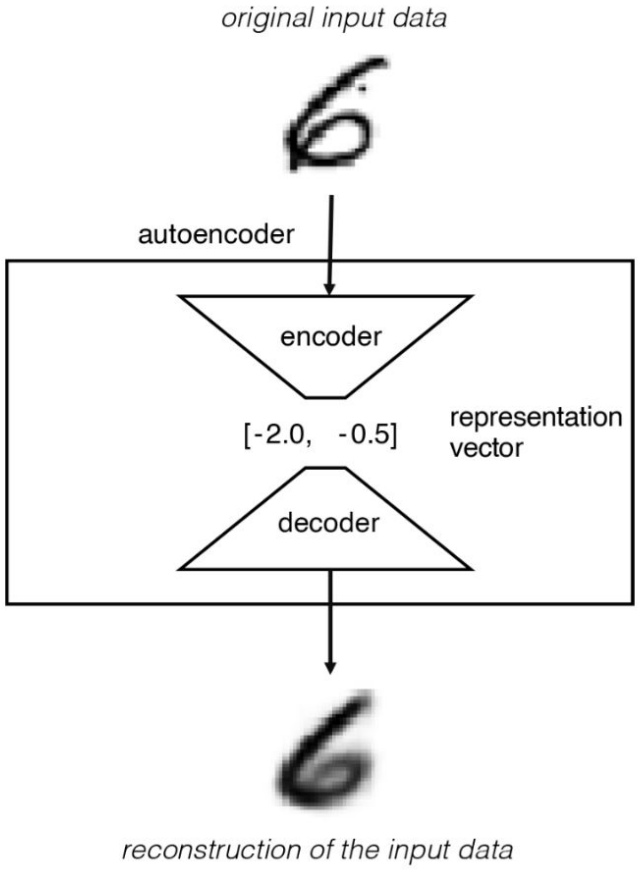


Decoder Architecture

Convolution Transpose Layers (upsampling layers)

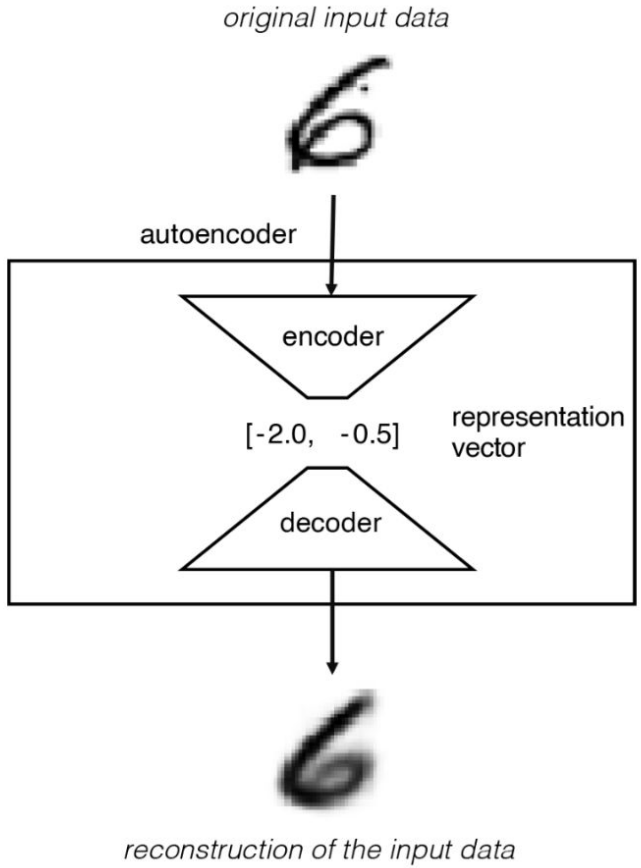
Encoder Architecture

Layer (type)	Output Shape
encoder_input (InputLayer)	(None, 28, 28, 1)
encoder_conv_0 (Conv2D)	(None, 28, 28, 32)
leaky_re_lu_1 (LeakyReLU)	(None, 28, 28, 32)
encoder_conv_1 (Conv2D)	(None, 14, 14, 64)
leaky_re_lu_2 (LeakyReLU)	(None, 14, 14, 64)
encoder_conv_2 (Conv2D)	(None, 7, 7, 64)
leaky_re_lu_3 (LeakyReLU)	(None, 7, 7, 64)
encoder_conv_3 (Conv2D)	(None, 7, 7, 64)
leaky_re_lu_4 (LeakyReLU)	(None, 7, 7, 64)
flatten_1 (Flatten)	(None, 3136)
encoder_output (Dense)	(None, 2)

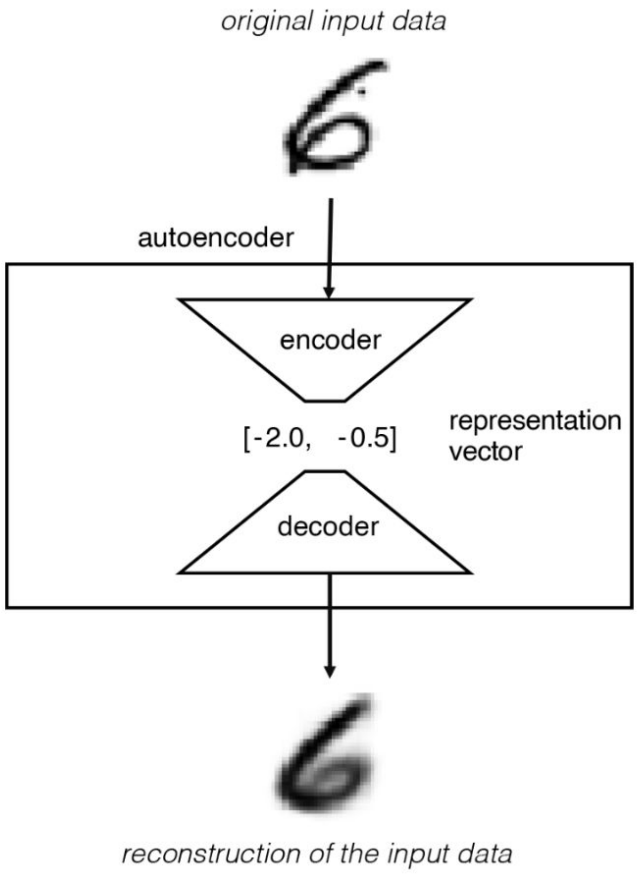


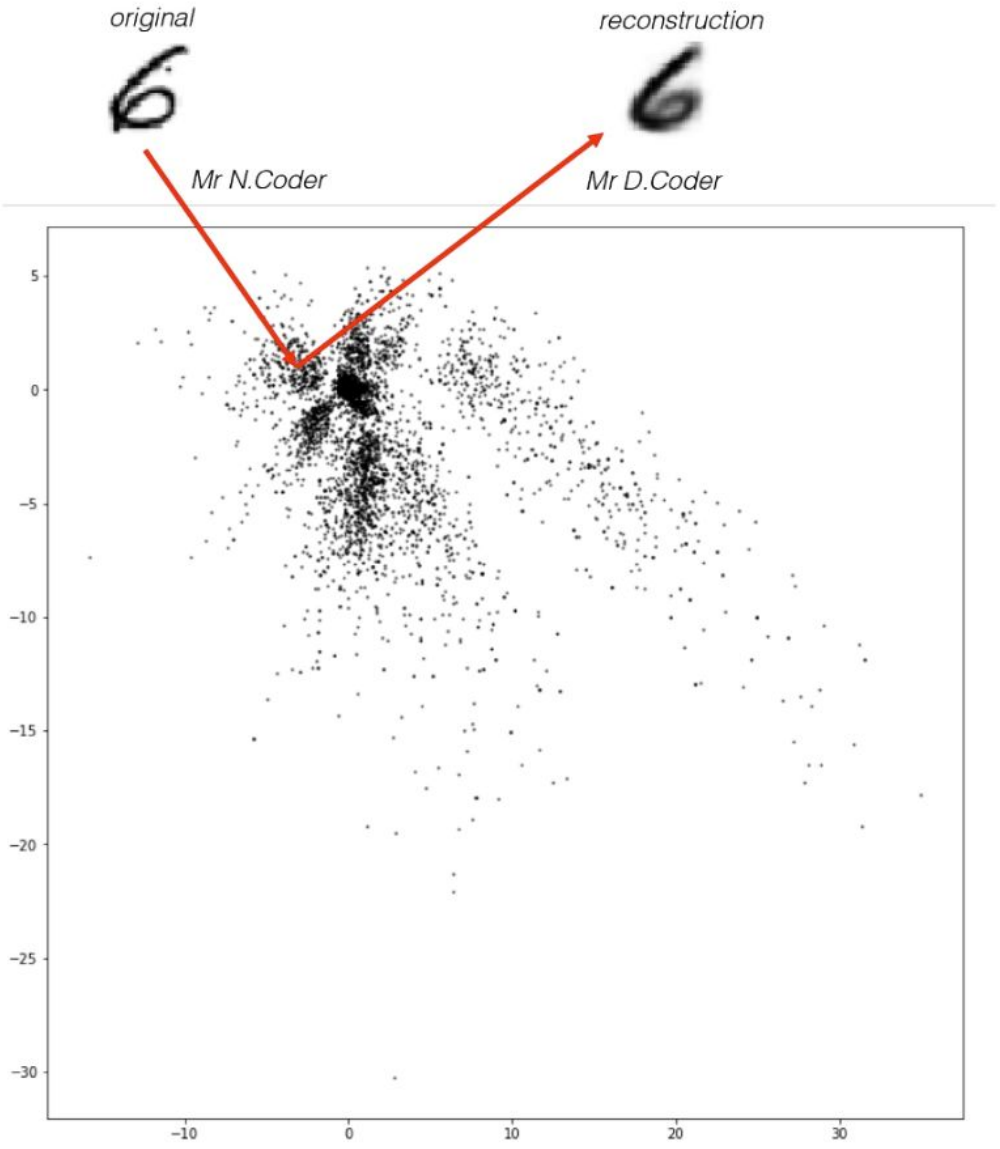
Decoder Architecture

Layer (type)	Output Shape
decoder_input (InputLayer)	(None, 2)
dense_1 (Dense)	(None, 3136)
reshape_1 (Reshape)	(None, 7, 7, 64)
decoder_conv_t_0 (Conv2DTran	(None, 7, 7, 64)
leaky_re_lu_5 (LeakyReLU)	(None, 7, 7, 64)
decoder_conv_t_1 (Conv2DTran	(None, 14, 14, 64)
leaky_re_lu_6 (LeakyReLU)	(None, 14, 14, 64)
decoder_conv_t_2 (Conv2DTran	(None, 28, 28, 32)
leaky_re_lu_7 (LeakyReLU)	(None, 28, 28, 32)
decoder_conv_t_3 (Conv2DTran	(None, 28, 28, 1)
activation_1 (Activation)	(None, 28, 28, 1)



Reconstruction





Reconstruction for Art Exhibition

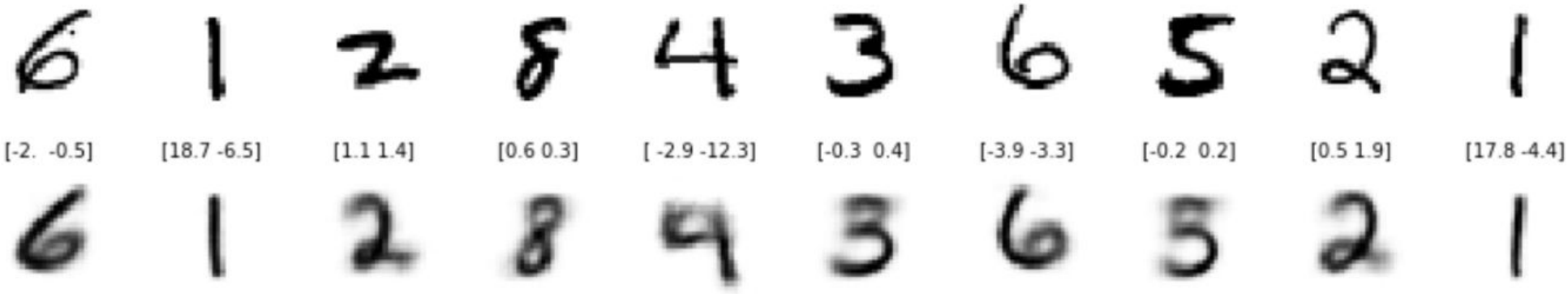


Figure 3-2. More examples of reconstructed paintings

Applications of Autoencoder

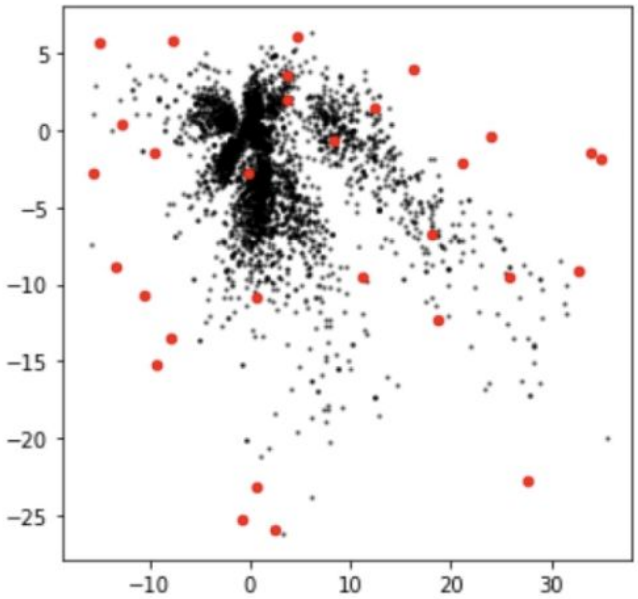
Image Restoration: For example, Denoising Autoencoder remove noise from images through learned encoding and decoding.

Representation Learning: Learn meaningful representations for input into other machine learning models.

Semantic Segmentation: Segment images into meaningful regions based on learned features.

Dimensionality Reduction: Reduce data dimensionality for visualization and speed.

Image Generation: Generate realistic images using Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs).



Variational Autoencoder

Generation as an application

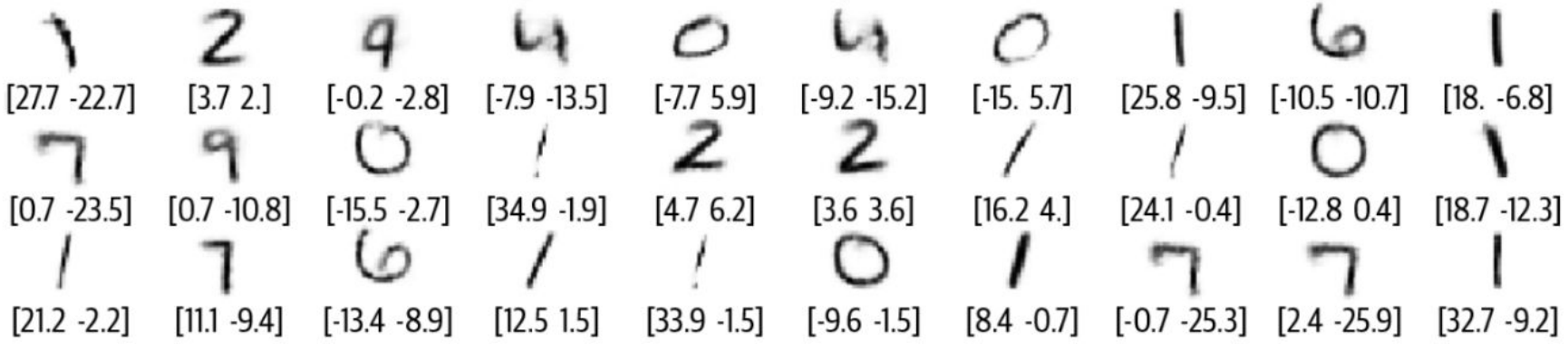
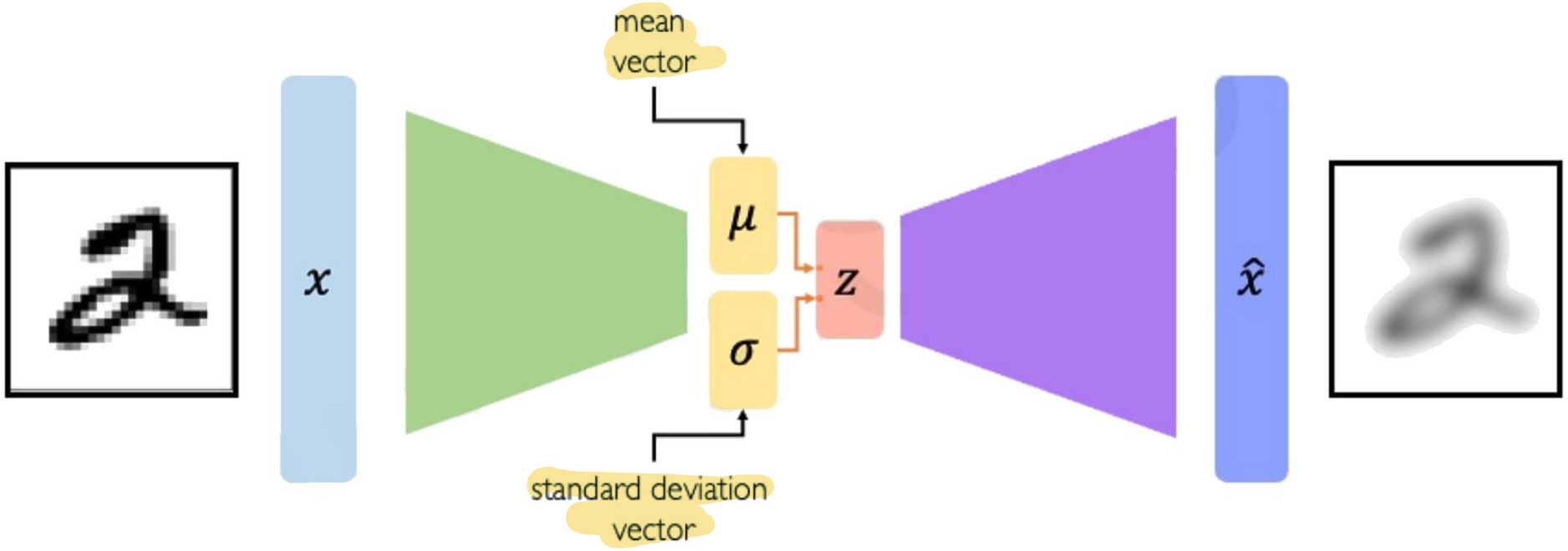


Figure 3-3. The new generative art exhibition

VAEs: key difference with traditional autoencoder



Variational autoencoders are a probabilistic twist on autoencoders!

Sample from the mean and standard deviation to compute latent sample

Autoencoders

Variational Autoencoders

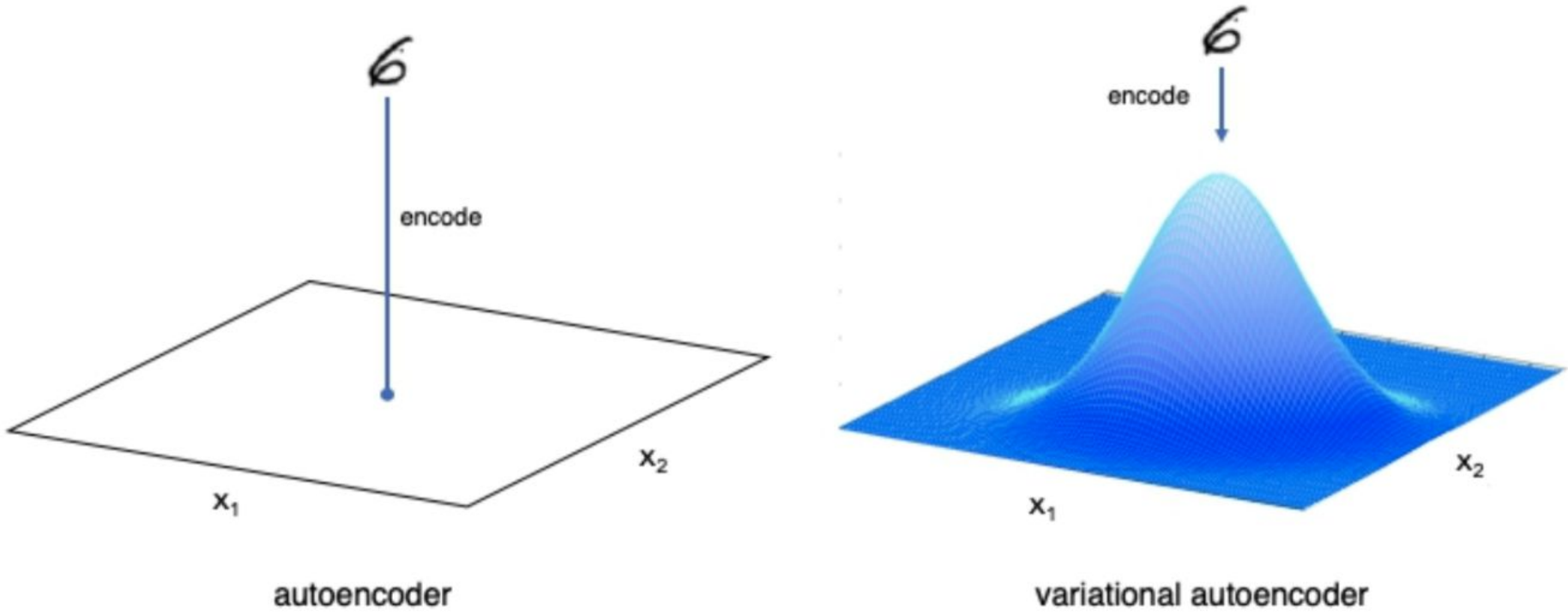


Figure 3-11. The difference between the encoder in an autoencoder and a variational autoencoder

Mapping to a point

Mapping to a Distribution (learning μ , σ)

Variational Autoencoders (VAE)

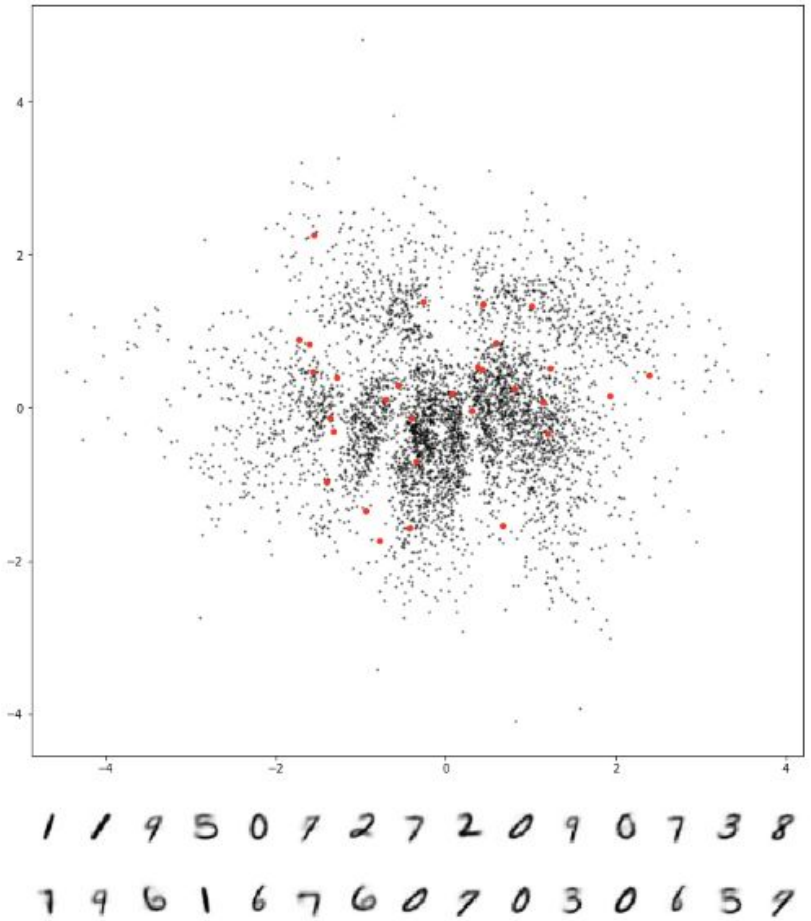
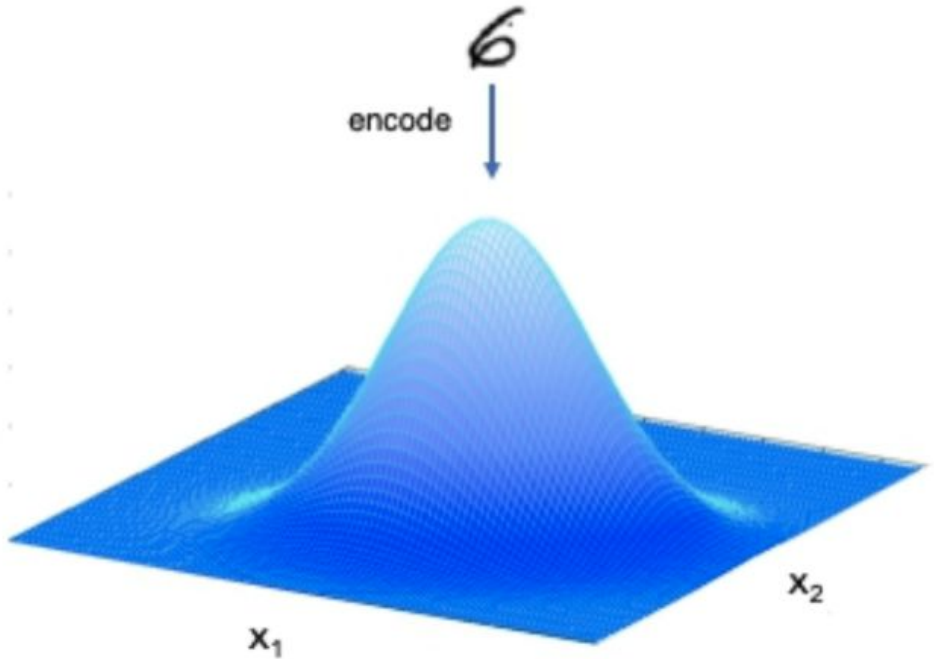
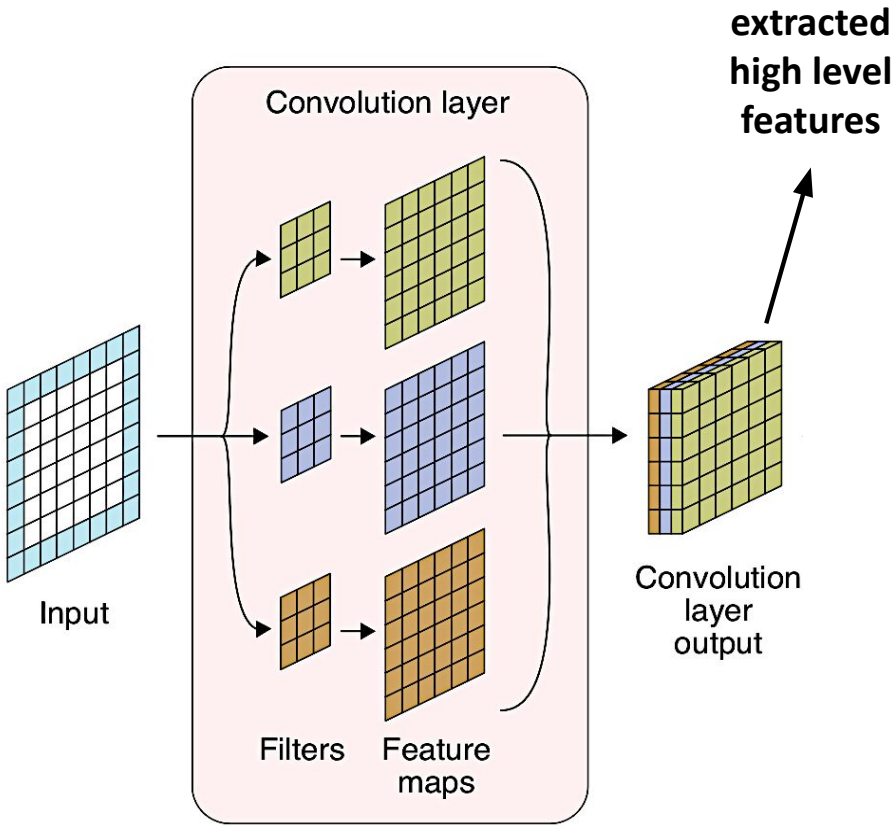


Figure 3-10. Artwork from the new exhibition

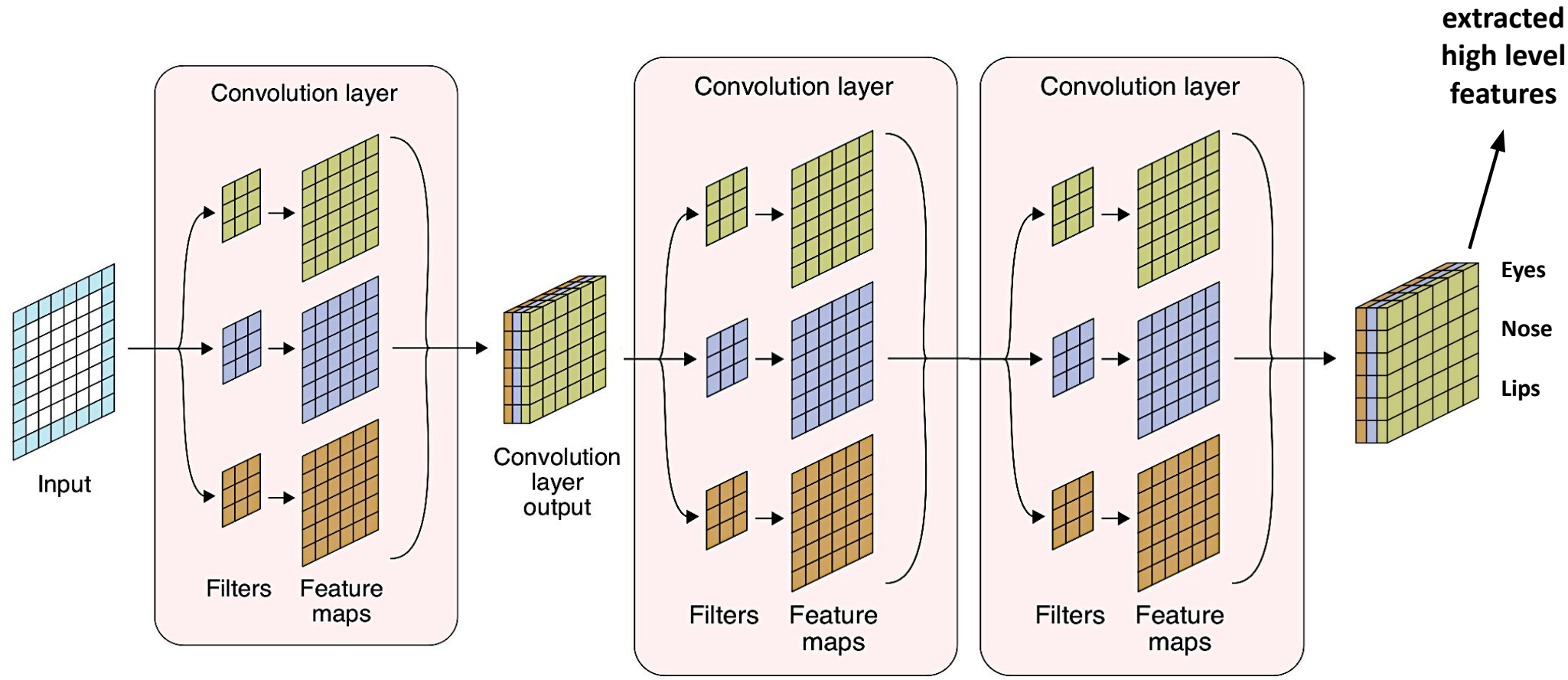


Visualizing CNNs

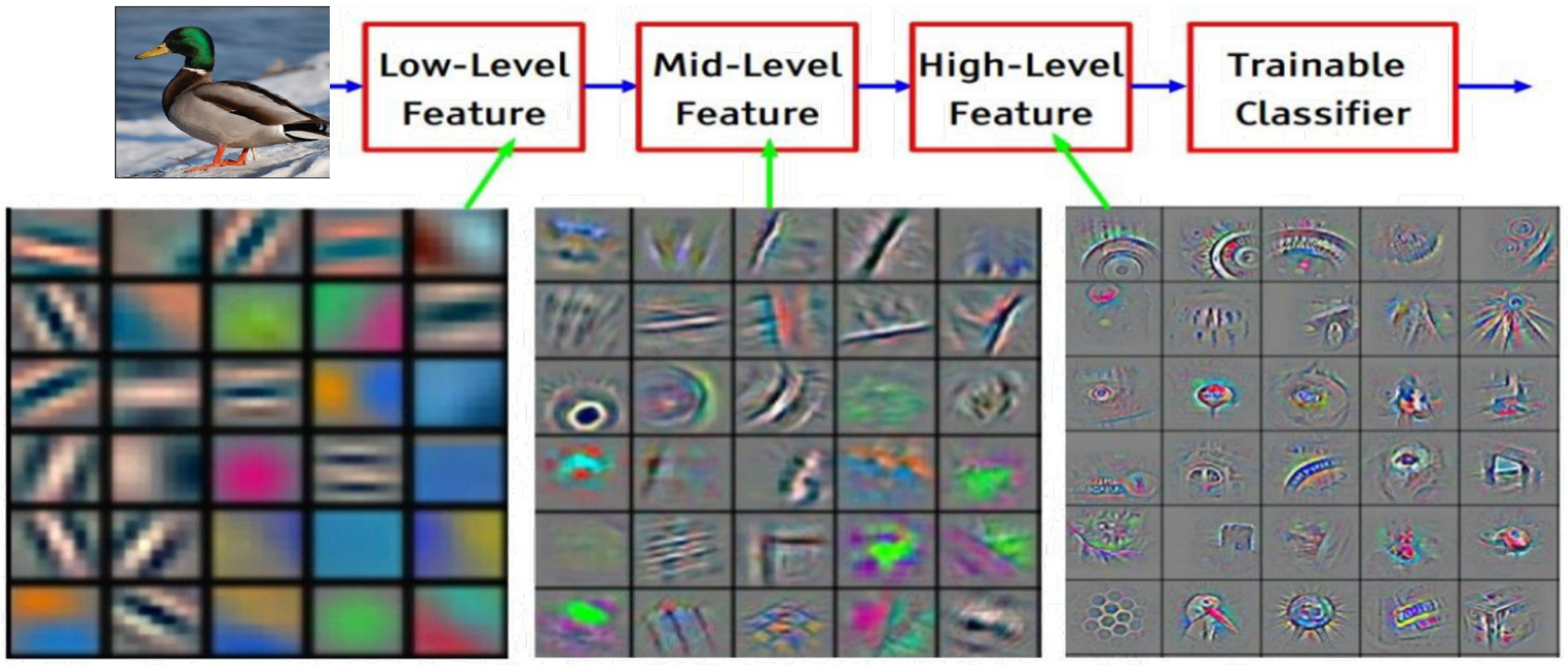
Feature Extraction Idea



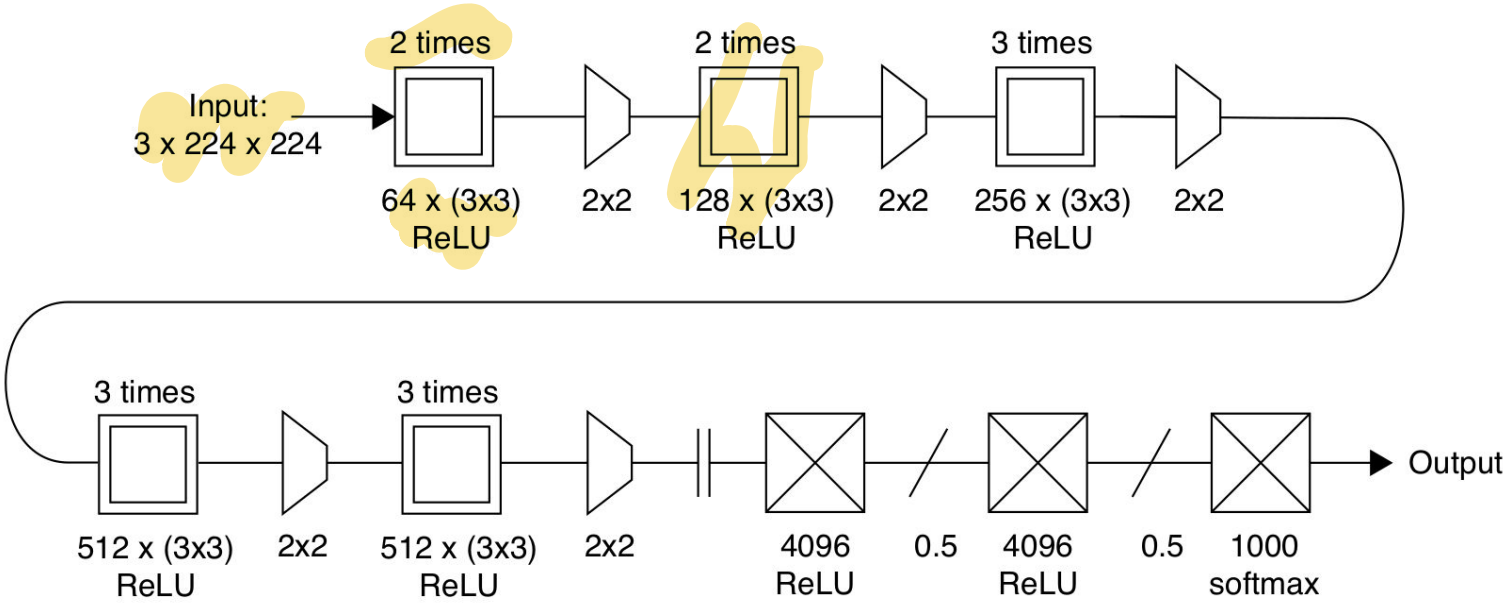
Feature Extraction Idea



CNN Learned Filters



Let us Visualizing Filters of VGG16



Let us Visualizing Filters of VGG16

Figure 17-18 The drake image that we use to visualize filter outputs

Let us Visualizing Filters of VGG16

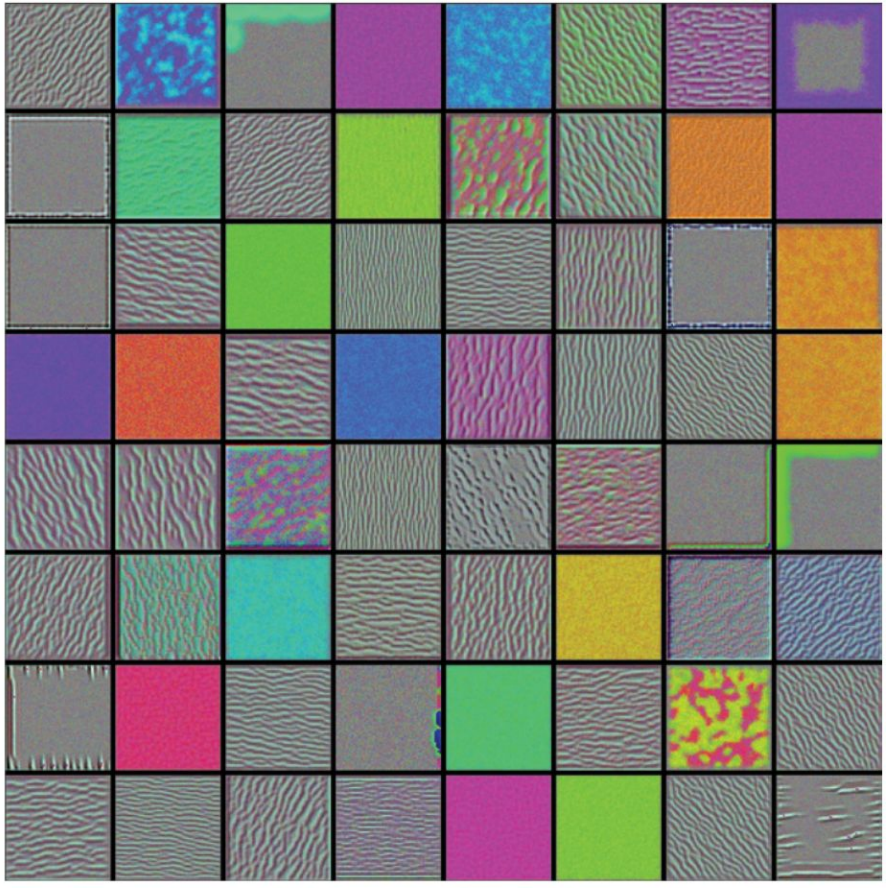


Figure 17-13: Images that get the biggest response from each of the 64 filters in the `block1_conv2` layer of VGG16

Let us Visualizing Filters of VGG16

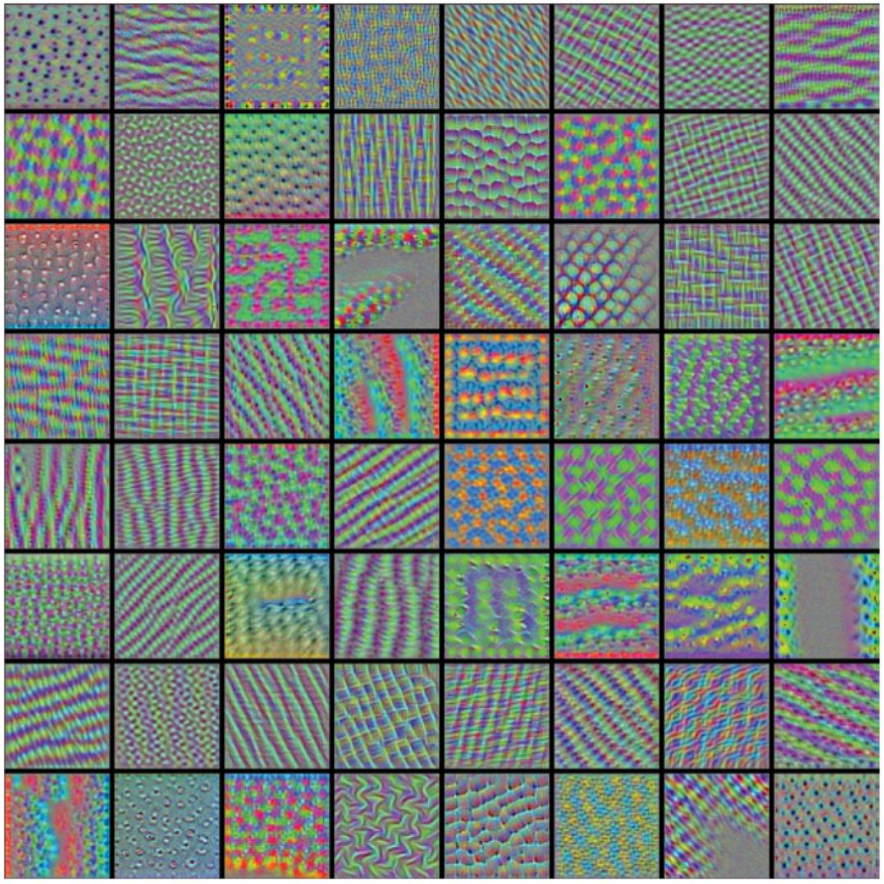


Figure 17-14: Images that get the biggest response from the first 64 filters in the `block3_conv1` layer of VGG16

Let us Visualizing Filters of VGG16

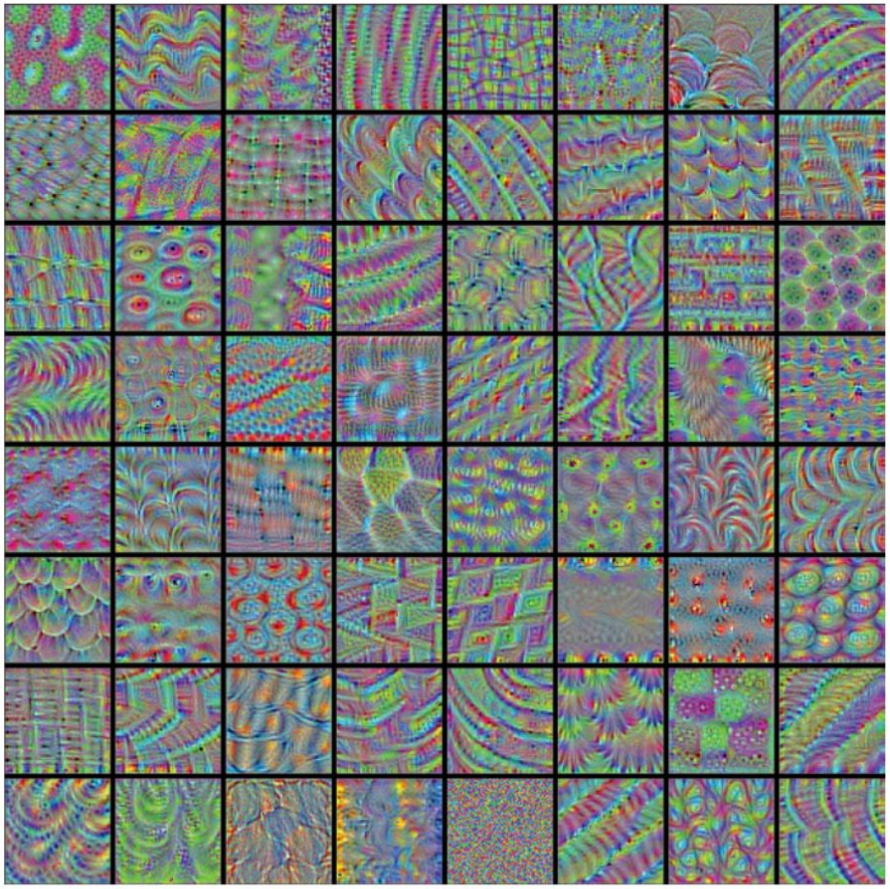


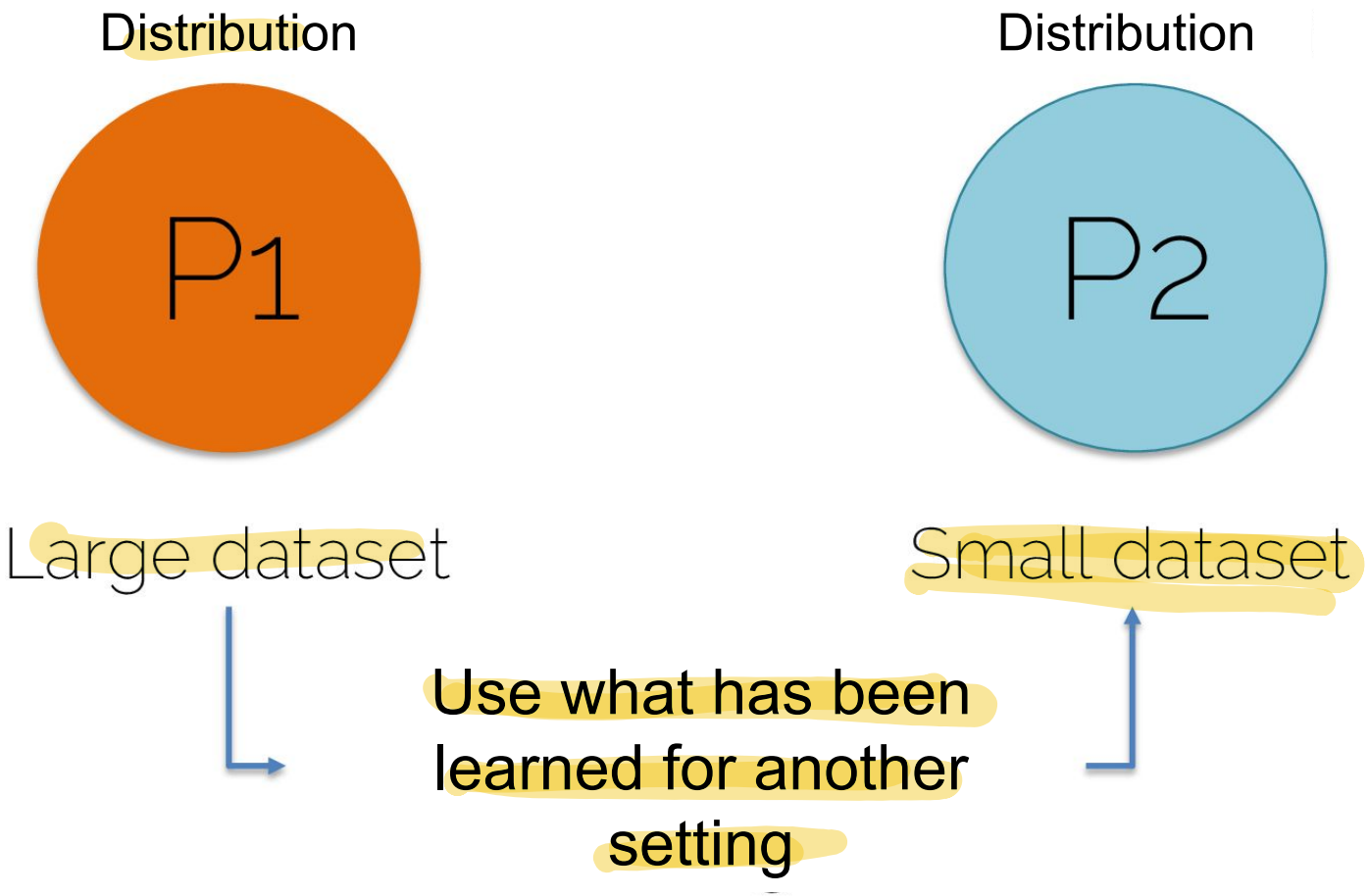
Figure 17-15: Images that get the biggest response from the first 64 filters in the `block4_conv1` layer of VGG16

Transfer Learning

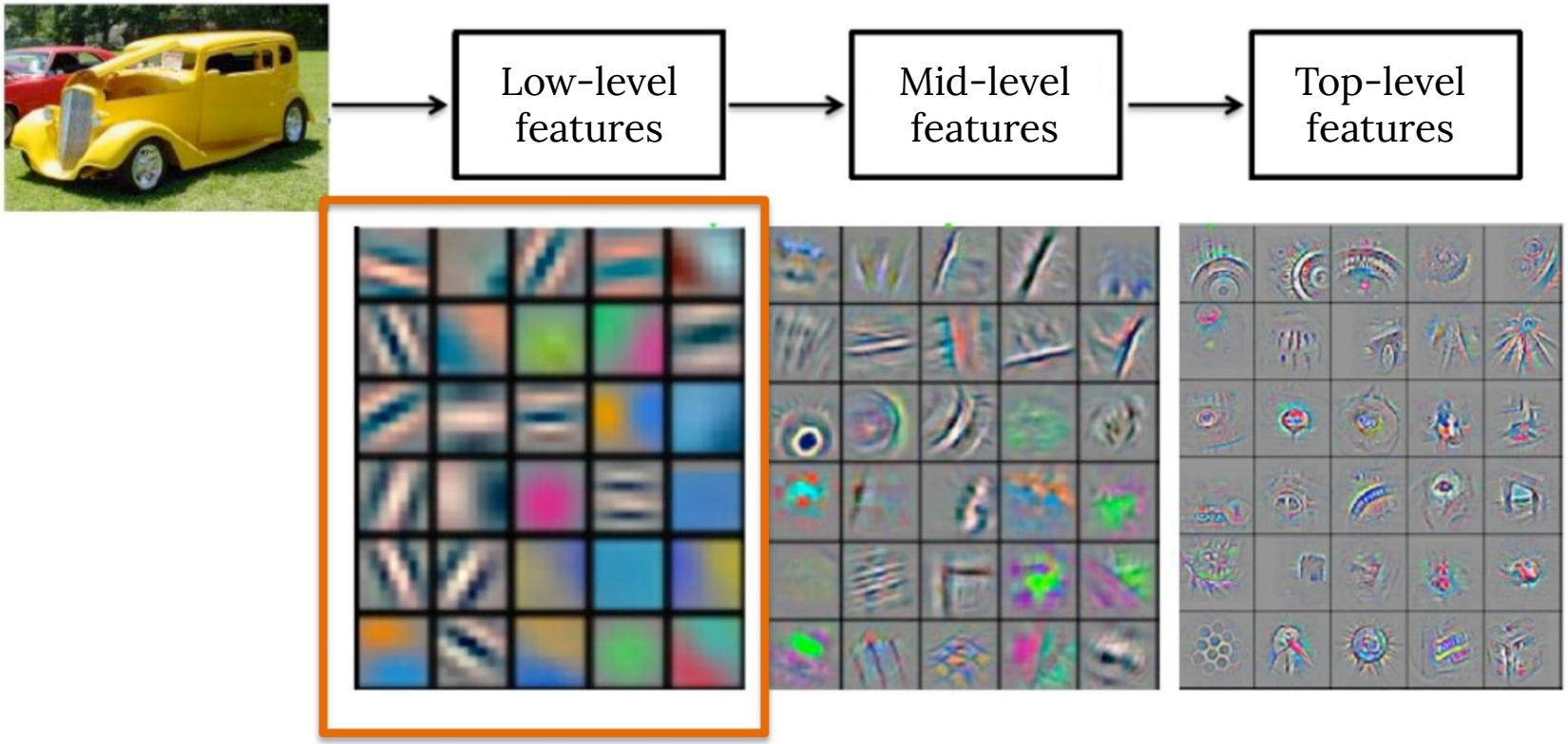
Transfer Learning

- Training your own model can be difficult with limited data and other resources, e.g.,
- It is a laborious task to manually annotate your own training dataset
- Why not reuse already pre-trained models?

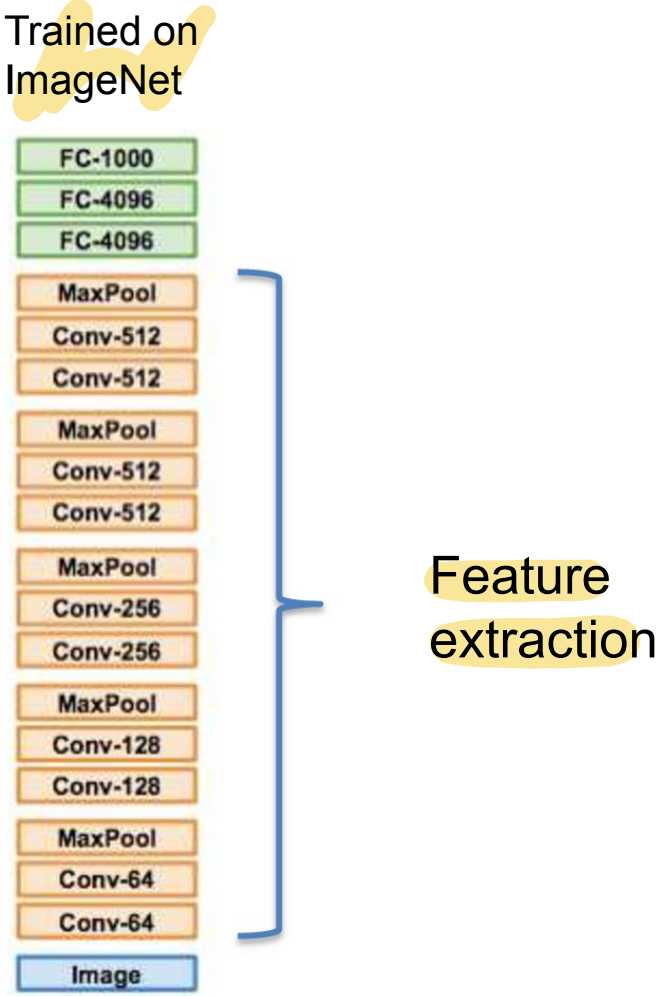
Transfer Learning



Transfer Learning for Images

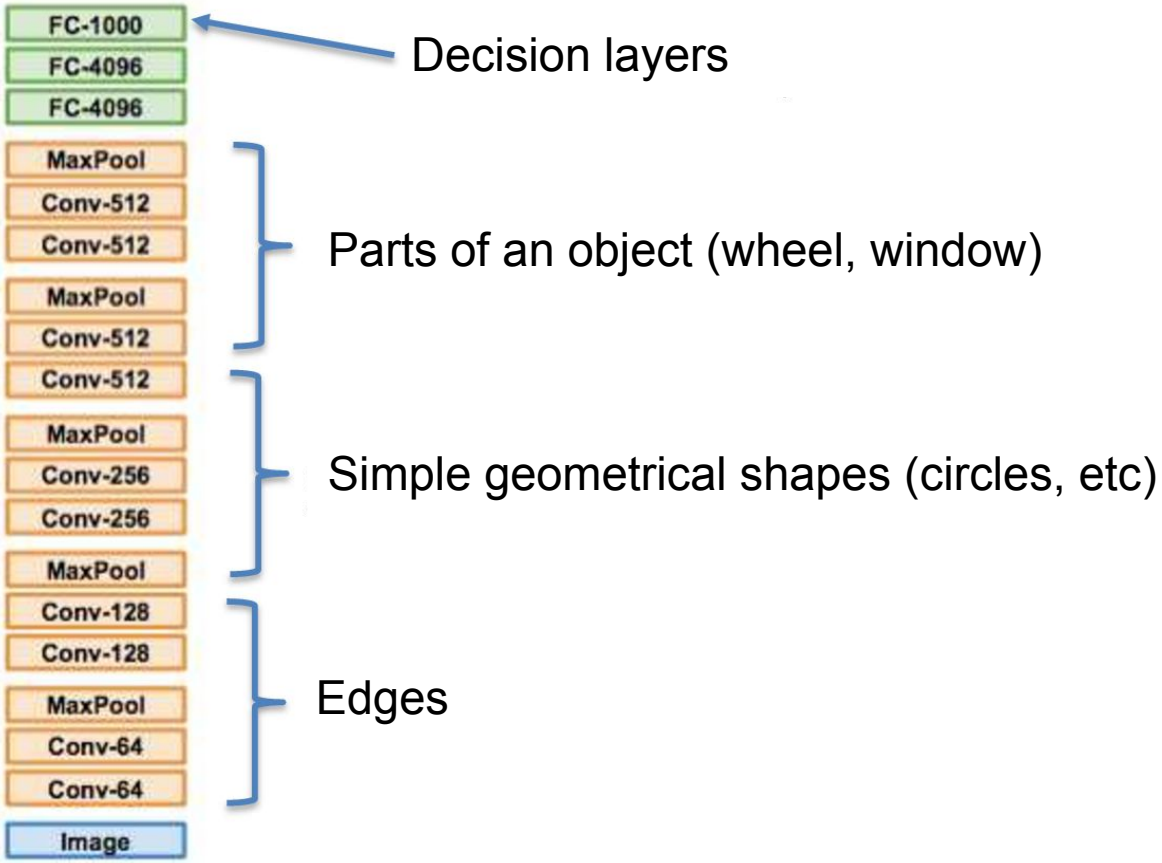


Transfer Learning



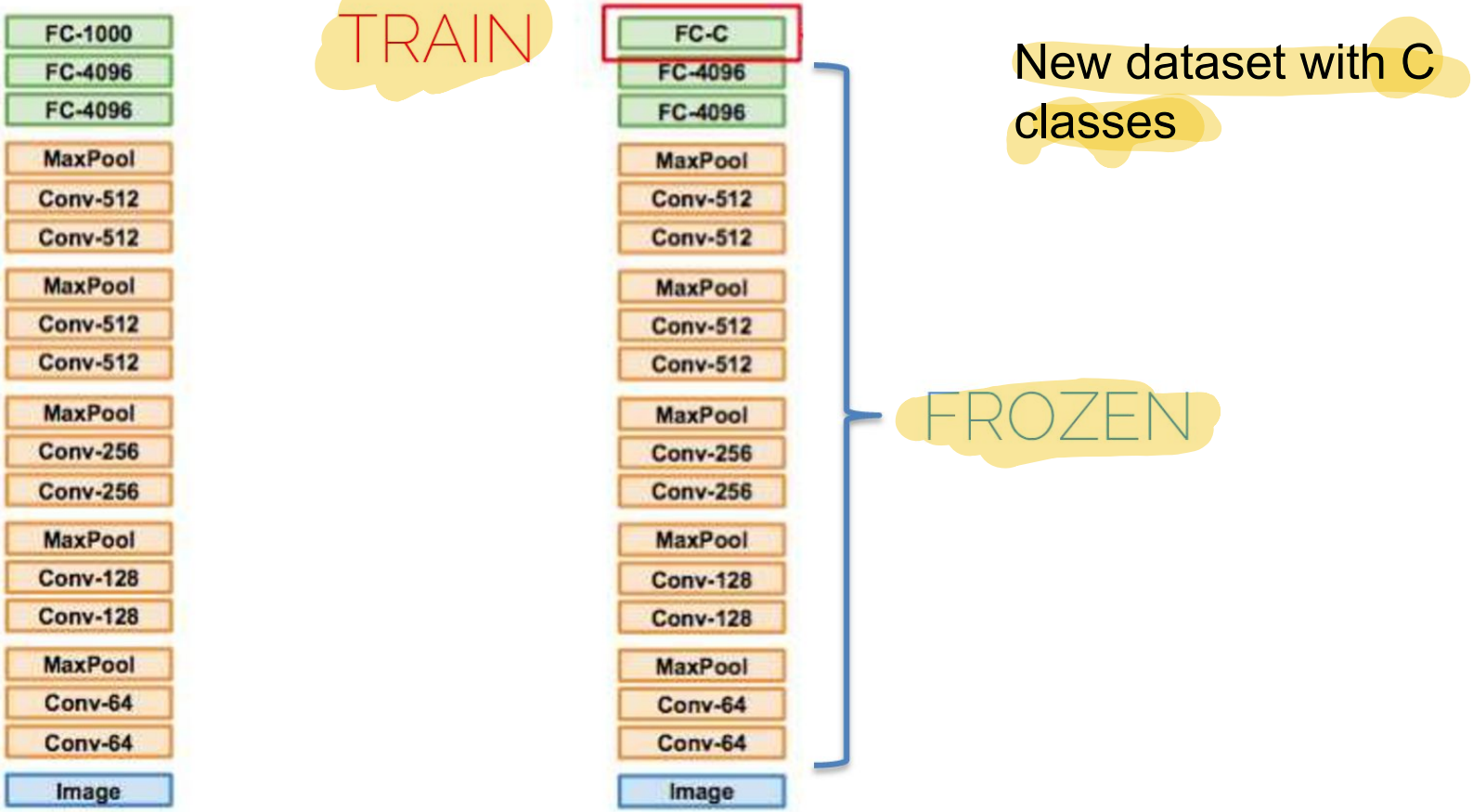
Transfer Learning

Trained on ImageNet



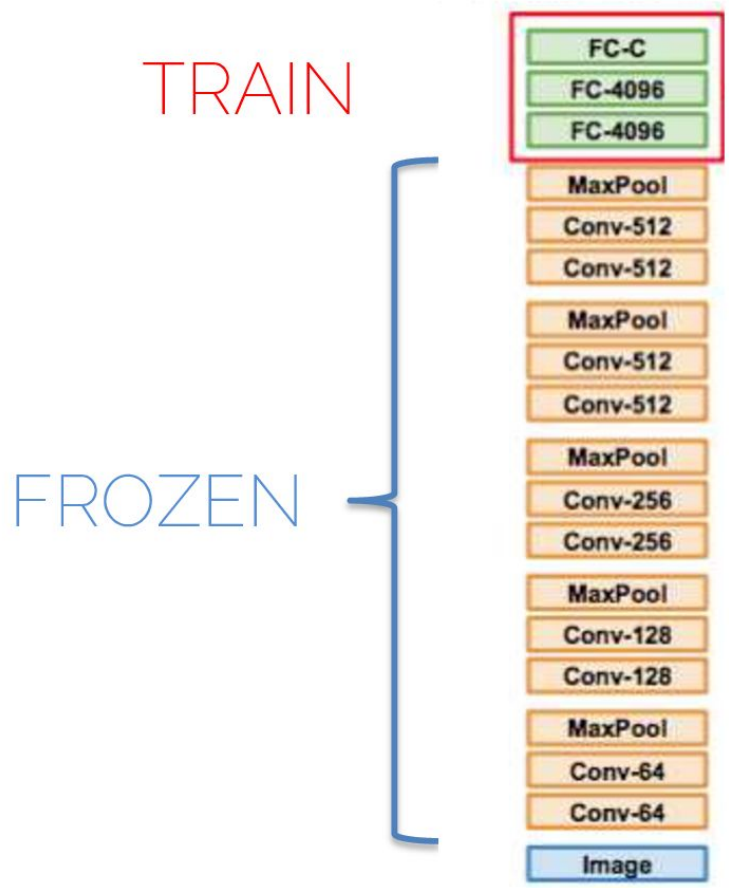
Transfer Learning

Trained on ImageNet



Transfer Learning

If the dataset is big enough train more layers

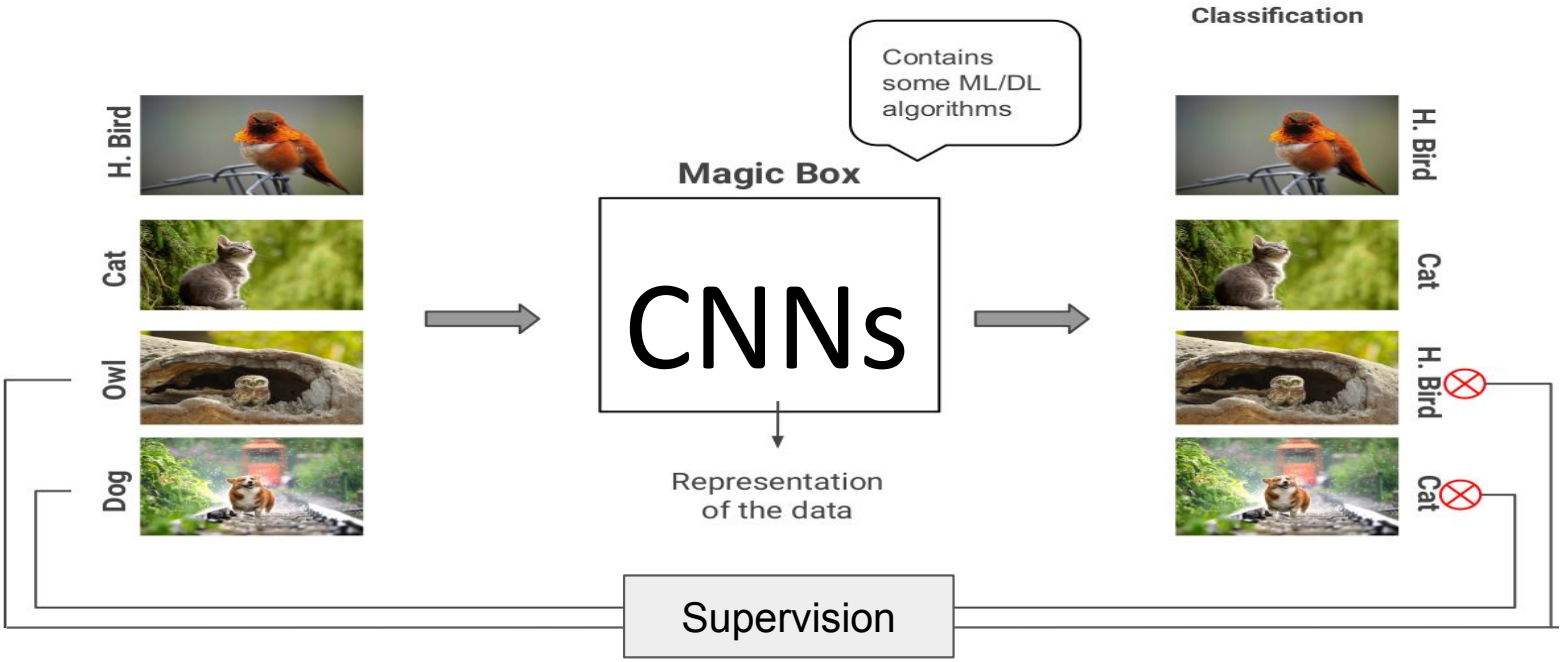


When Transfer Learning makes Sense

- When task T1 and T2 have the same input (e.g. an RGB image)
- When you have more data for task T1 than for task T2
- When the low-level features for T1 could be useful to learn T2

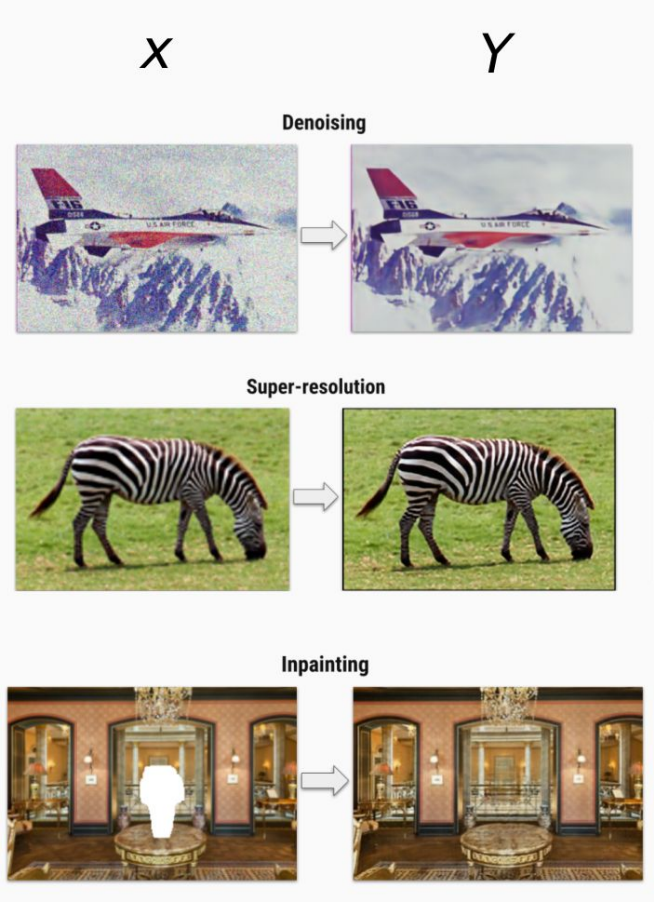
Self-supervised learning

Supervised Approaches



— Hard and expensive to obtain annotations

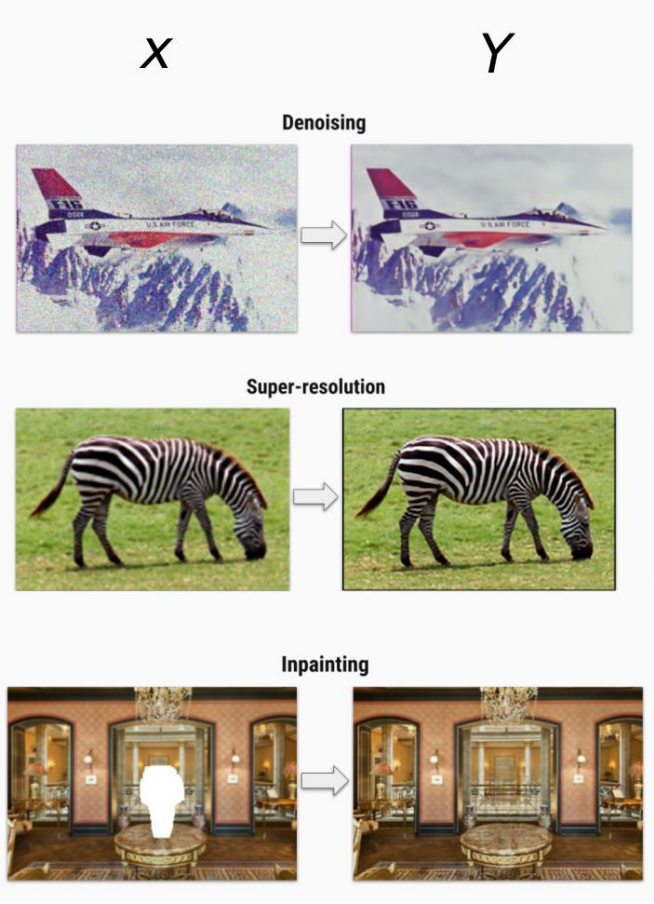
Unsupervised Approaches



Denoising

$$\min_{Y} ||Y - X||$$

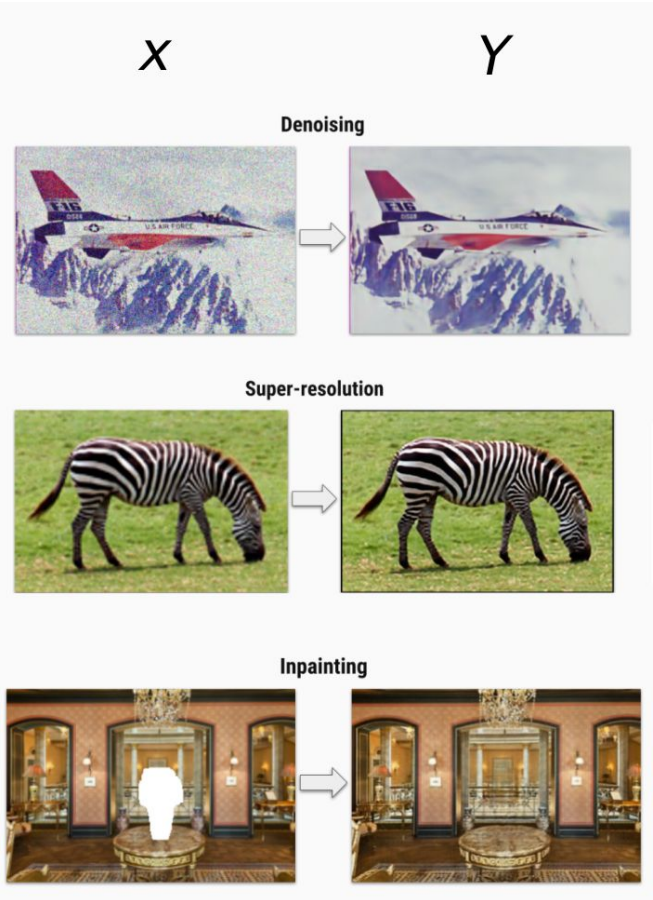
Unsupervised Approaches



Super-resolution

$$\min_{Y} ||d(Y) - X||$$

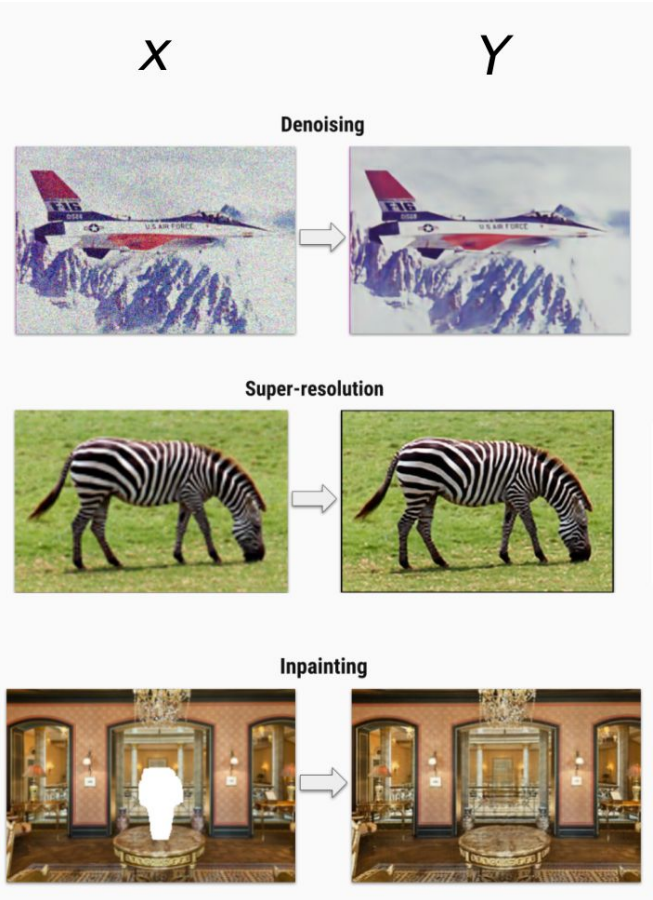
Unsupervised Approaches



Inpainting

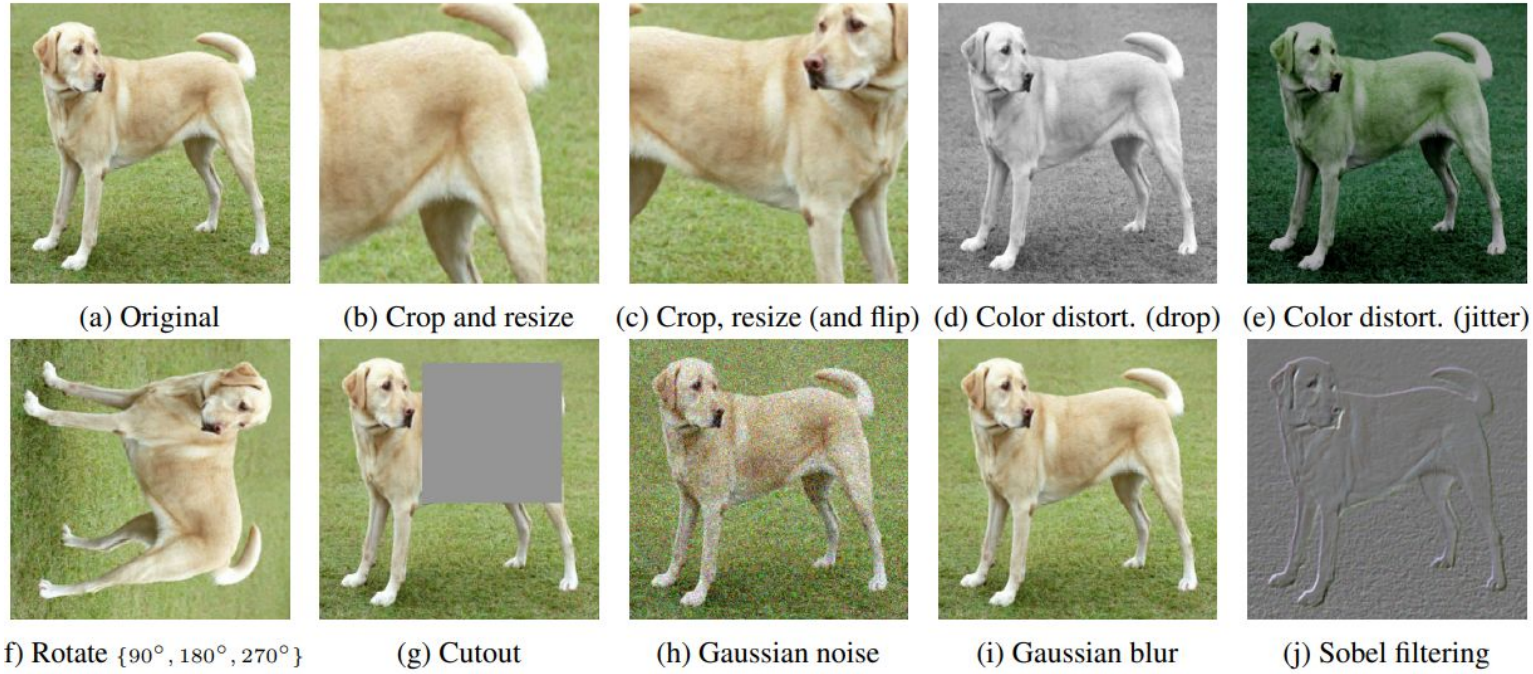
$$\min_{Y} ||(Y - X) \odot M||$$

Unsupervised Approaches



Limited Applications

Self-supervision by Augmentation



SSL is a paradigm where a model is trained on a task using the data itself to generate supervisory signals, rather than relying on external labels provided by humans.

Self-supervised Approaches

Why self-supervised?

— Hard and expensive to obtain annotations

— Make the most out of the existing unlabelled data

- Instagram: >1 billion images uploaded / day

- YouTube: >300 hrs of video uploaded / minute

— Alternative to the strong supervisions (labels)

Self-supervised learning (SSL)

- **Contrastive** self-supervised learning: training data can be divided into positive examples and negative examples.
- **Non-contrastive** self-supervised learning: uses only positive examples.

Similarity Learning: when and why?

Learn a similarity function

A

B

Low similarity score

A

B

High similarity score

Contrastive or Non-Contrastive Learning