DEEP FAKE GENERATION AND DETECTION

Christophe Charrier

GREYC Laboratory (UMR 6072)
UNICAEN – ENSICAEN - CNRS

christophe.charrier@unicaen.fr

GREYC

GREYC
Electronics and Computer Science Laboratory

Normandie Université

UNICAEN

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

cnrs

GREYC Lab

- Research in Digital Sciences
- Image processing, artificial intelligence, data science, instrumentation, theoretical computer science, cybersecurity, natural language processing …
- In Normandy, France

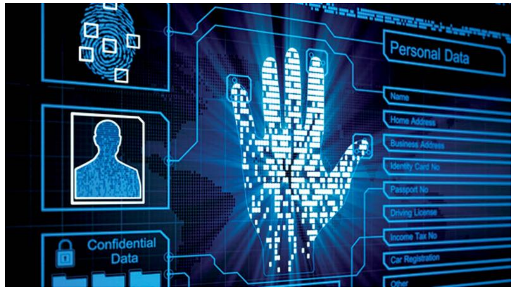
SAFE Team Members
Security, Architecture, Forensics, biomEtrics

- Faculty staff
  - 3 full PR
  - 5 associate PR (4 HdR)
  - 1 CNRS researcher (HdR)
  - 2 research ing.
- Team members
  - 8 PhD students
  - 7 associated researchers
  - 1 associated PR under contract (PAST)

Research activities

Biometrics

- *Biometric systems design*
  - New biometric systems
- *Evaluation of biometric systems*
  - Quality of biometrics data.
  - Presentation attacks detection
- *Biometrics data protection*
  - non-invertible transformation schemes



Security Architectures and models

- *Security of future SDN/5G/6G network technologies*
  - IoT
  - Junction of Physical and Cyber wolrds.
- *Detection of attacks and associated countermeasures*
- *Boolean functions for security*
  - correcting codes, Boolean functions, steganography.

Forensics

- *Automatic language processing*
  - analysis of digital text traces
  - automatic extraction of information
- *Analysis of digital traces*
  - linking digital identity and the real identity of individuals
  - analysis of societal interactions in the cyberspace
  - Deepfake, images/video forgery
- *Personal data protection*
  - Privacy protection.

1. What is a deepfake?
2. How is generated a deepfake?
    - Strategy 1: Autoencoder
    - Strategy 2: GAN
3. Deepfake misuse
4. Deepfake detection
5. Future trends

Normandie Université

GREYC
Electronics and Computer Science Laboratory

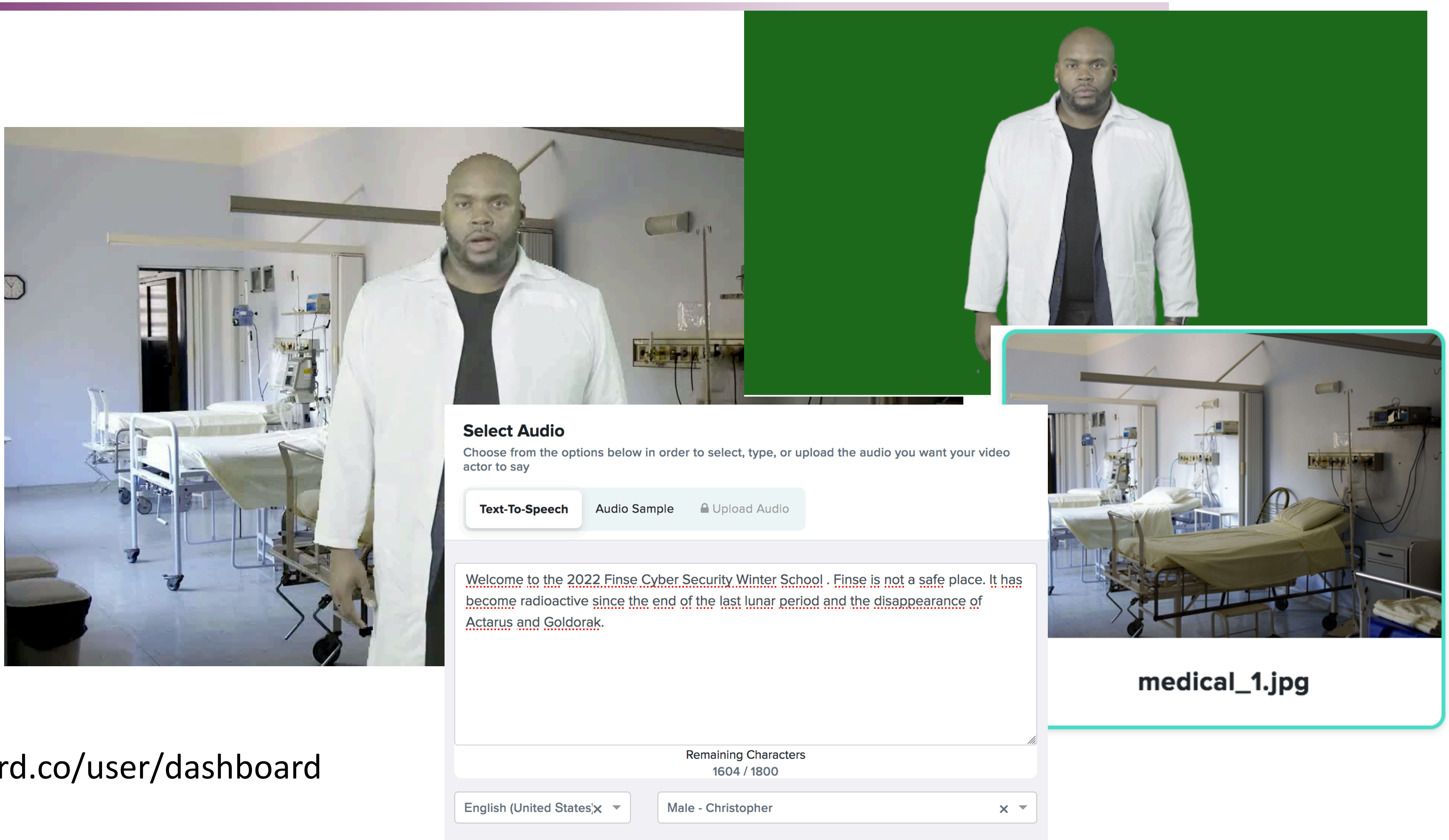
WHAT IS A DEEPFAKE?

cnrs

ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

UNICAEN

Alert message from the health authorities



https://login.deepword.co/user/dashboard

What is a deepfake?

Have you ever

- Come across quite strange tiktok videos with celebrities?
  - Tom cruise

- seen a person imitate different celebrities?
  - Eg. Robin Williams impersonating Jack Nicholson

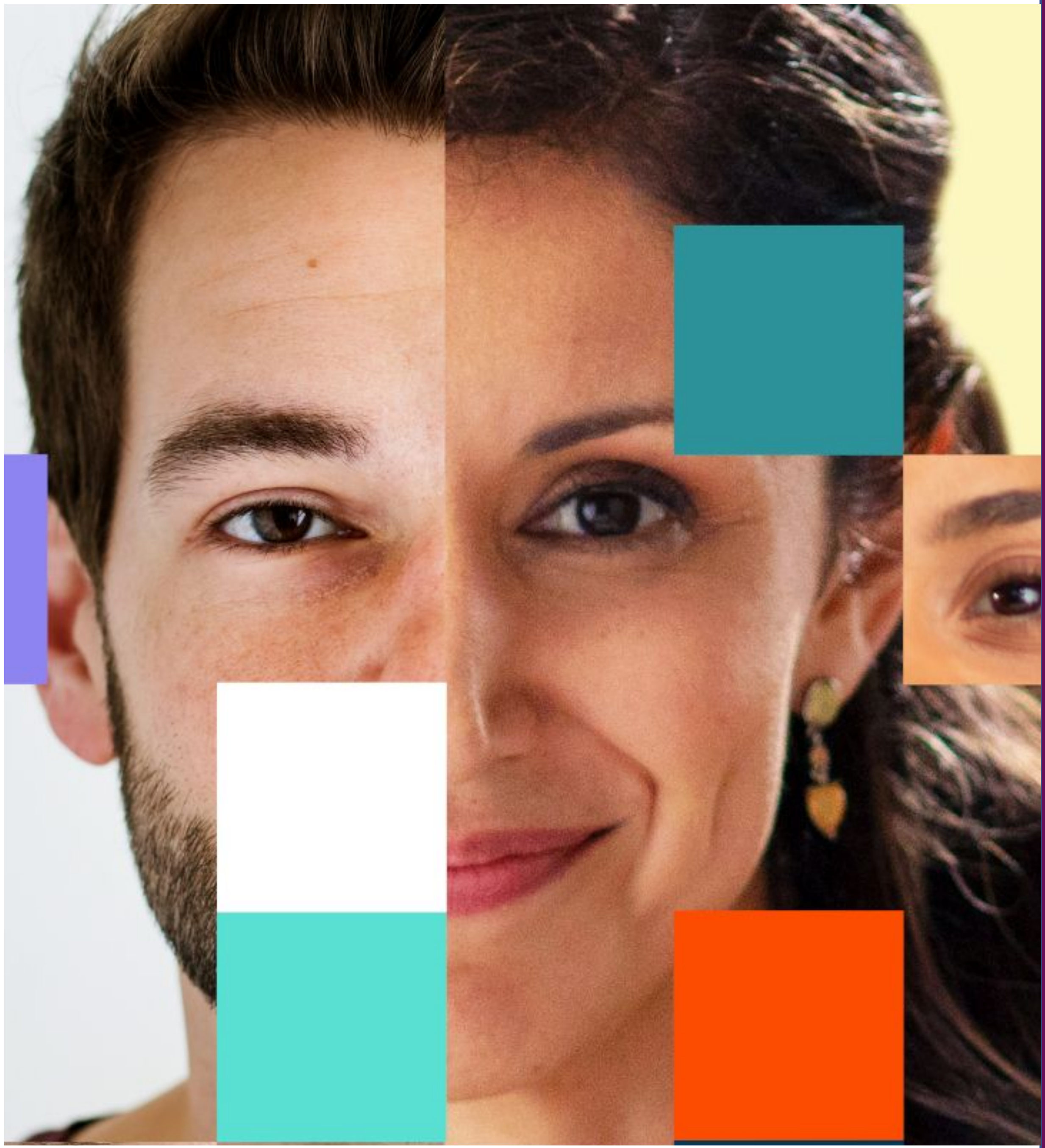
- noticed something strange in a person's voice or face?

What is a deepfake?

- **Deepfakes** are synthetic media in which a person in an existing video or image is replaced by someone else likeness.
- While the act of faking content is not new, deepfakes led in powerful techniques from machine learning and artificial intelligence to manipulate video and audio content with a high potential to deceive.
- Example : Obama’s pubic service annoucement

What is a deepfake?

- Many app exist to create deepfake
- Among them, we can cite
  - Reface app (voice, face swap)
  - FaceApp
  - Zao
  - SpeakPic
  - DeepFaceLab
  - FakeApp
  - Reflect
  - Deepfake Web



Normandie Université

GREYC
Electronics and Computer Science Laboratory

HOW IS GENERATED A DEEPFAKE?

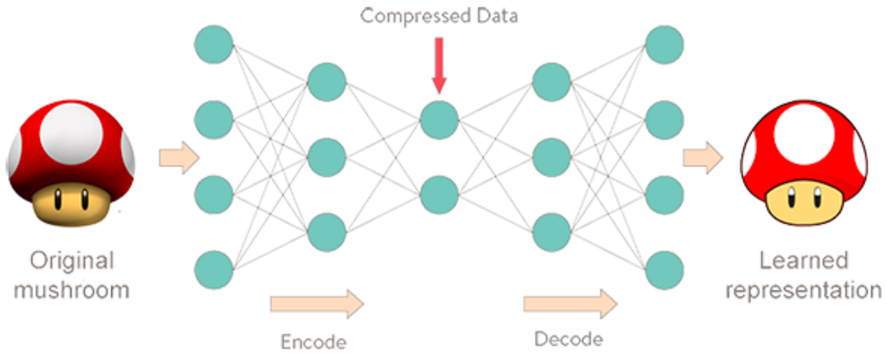
cnrs

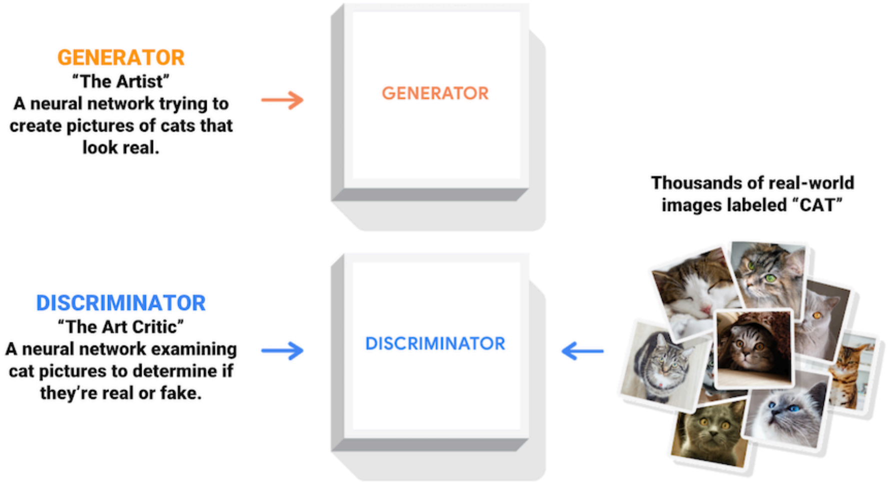
ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

UNICAEN

How does a deepfake work?

- The main machine learning methods used to create deepfakes are based on deep learning approaches and involve training Generative Neural Network (GNN) architectures, such as **autoencoder** or **Generative Adversarial Networks** (GAN).



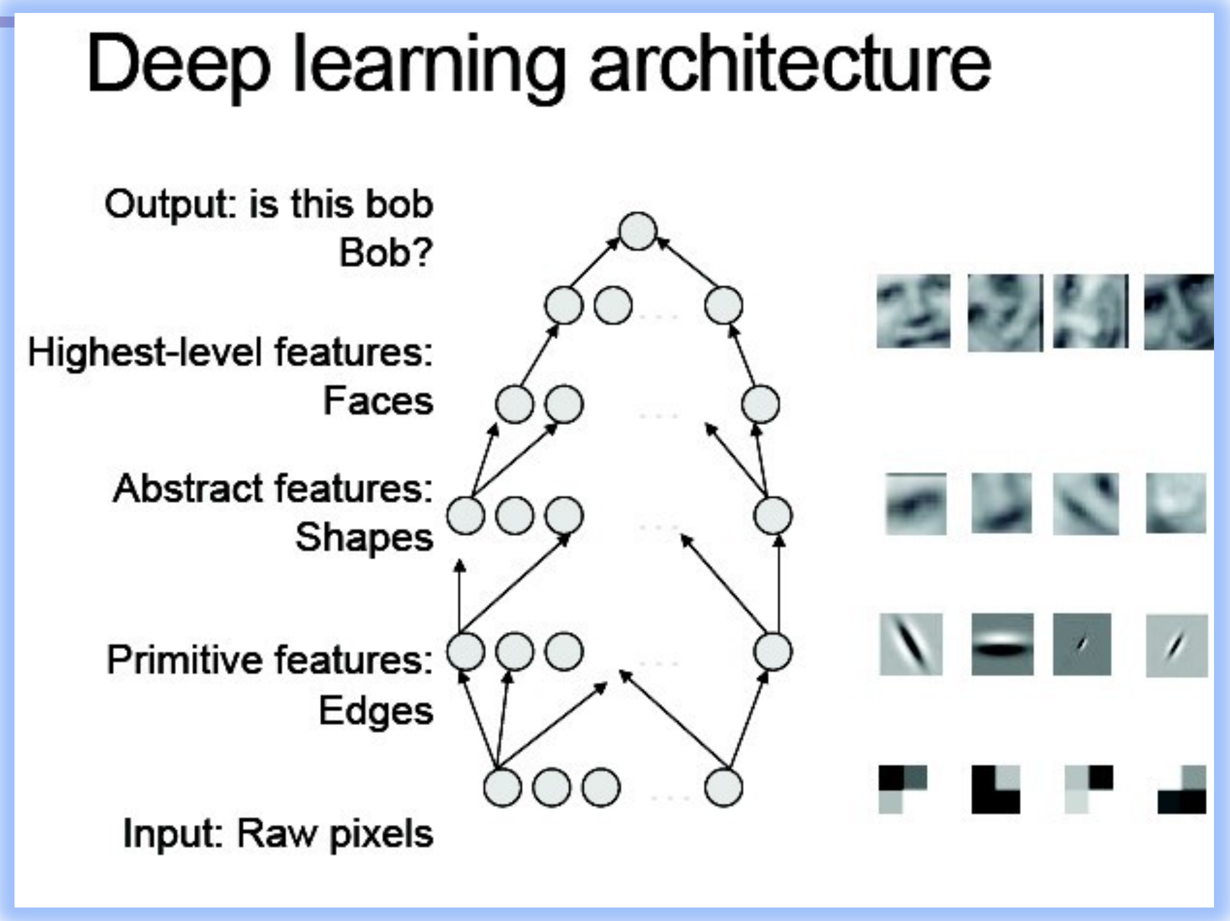


Hierarchical representations

"Deep learning methods aim at **learning feature hierarchies** with features from higher levels of the hierarchy formed by the composition of lower level features.

Automatically learning features at multiple levels of abstraction allows a system to learn **complex functions** mapping the input to the output directly from data, without depending completely on human-crafted features."

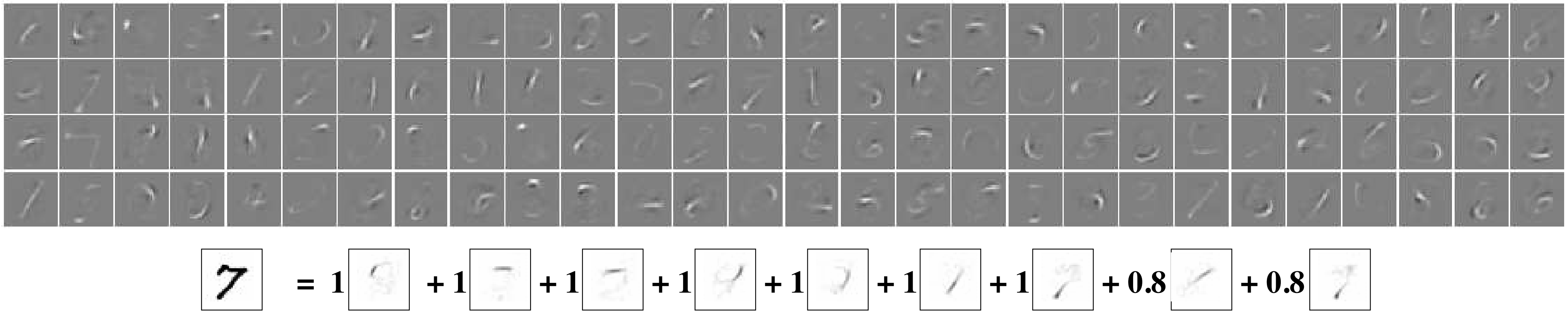
— Yoshua Bengio



[Bengio, "On the expressive power of deep architectures", *Talk at ALT*, 2011]
[Bengio, *Learning Deep Architectures for AI*, 2009]

Sparse and/or distributed representations

Biological motivation: V1 visual cortex



Example on MNIST handwritten digits
An image of size 28x28 pixels can be represented using a small combination of **codes** from a **basis set**.

[Ranzato, Poultney, Chopra & LeCun, "Efficient Learning of Sparse Representations with an Energy-Based Model ", *NIPS*, 2006; Ranzato, Boureau & LeCun, "Sparse Feature Learning for Deep Belief Networks ", *NIPS*, 2007]

Supervised vs unsupervised learning
What about data?

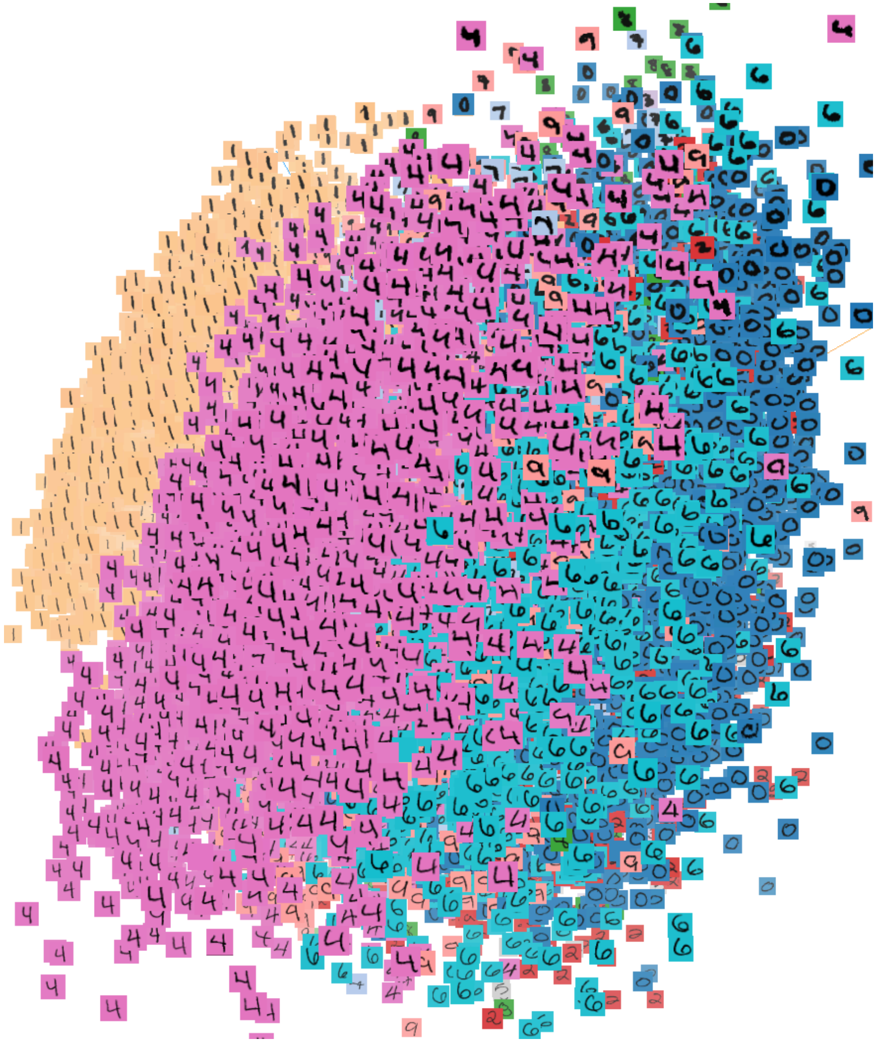
- To date, always supervised
- Need at lot of labeled data
- What to do if huge amount of unlabeled data?
- Supervised learning towards unsupervised learning

"We expect unsupervised learning to become far more important in the longer term. Human and animal learning is largely unsupervised: we discover the structure of the world by observing it, not by being told the name of every object." – LeCun, Bengio and Hinton

LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521,** 436–444 (2015)

Unsupervised Learning

- ▶Data: X (no labels!)
- ▶Goal: Learn the structure of the data (learn correlations between features)



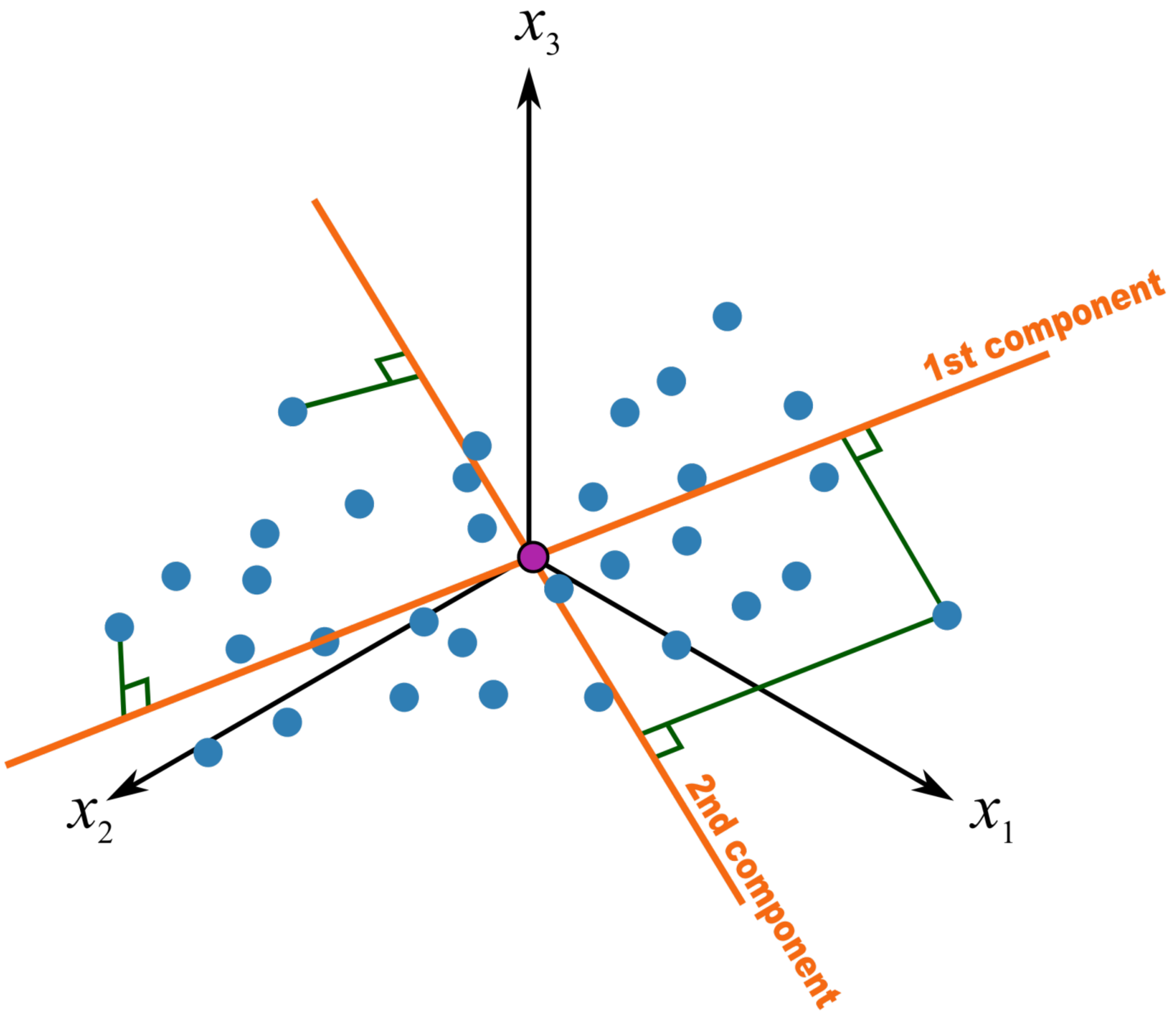
Unsupervised Learning

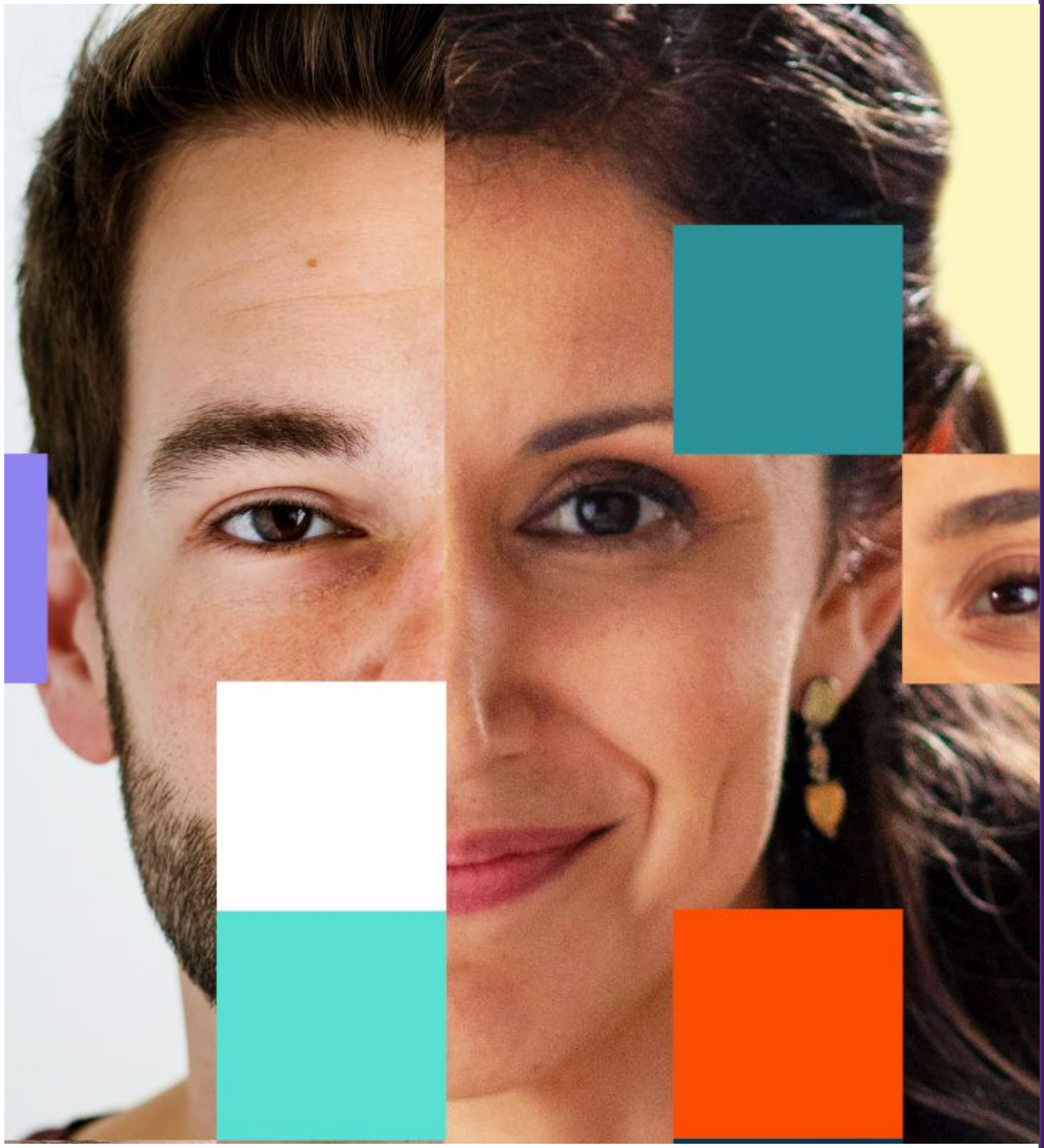
- Examples: Clustering, **Compression**, **Feature & Representation learning, Dimensionality reduction**, Generative models ,etc.

PCA – Principal Component analysis

- Statistical approach for data compression and visualization
- Invented by Karl Pearson in 1901

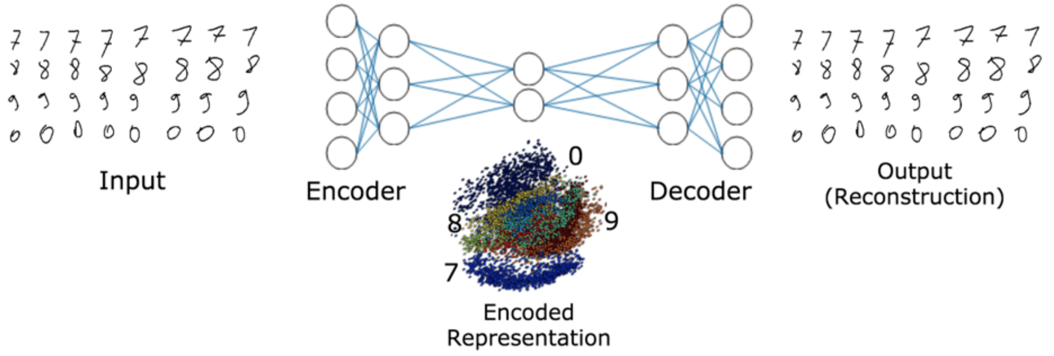
- Weakness:
  - linear components only.



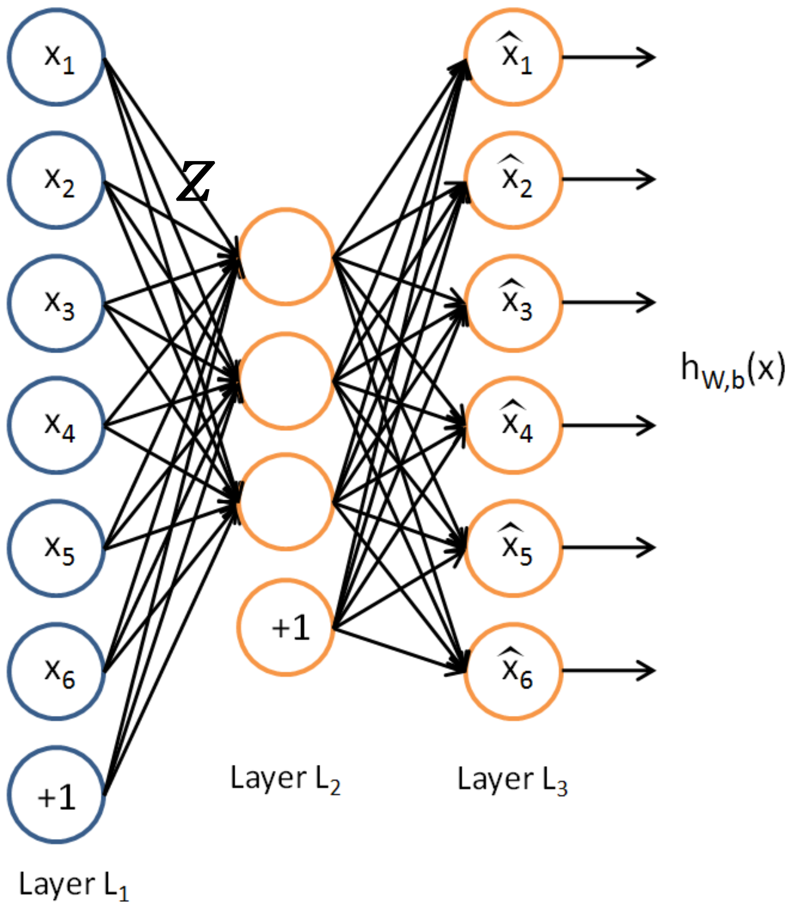


HOW IS GENERATED A DEEPFAKE?

Strategy 1: the autoencoders



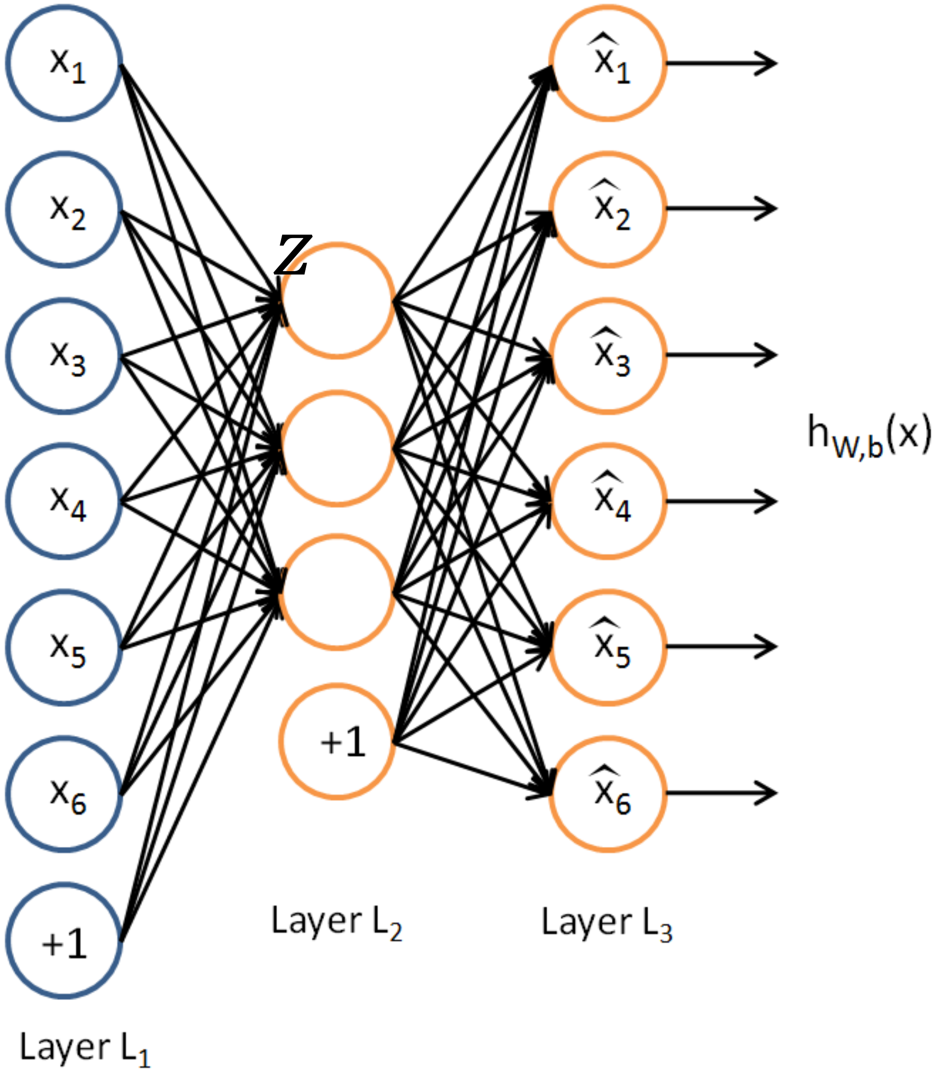
Traditional Autoencoder



Traditional Autoencoder

- Unlike the PCA now we can use activation functions to achieve non-linearity.

- It has been shown that an AE without activation functions achieves the PCA capacity.



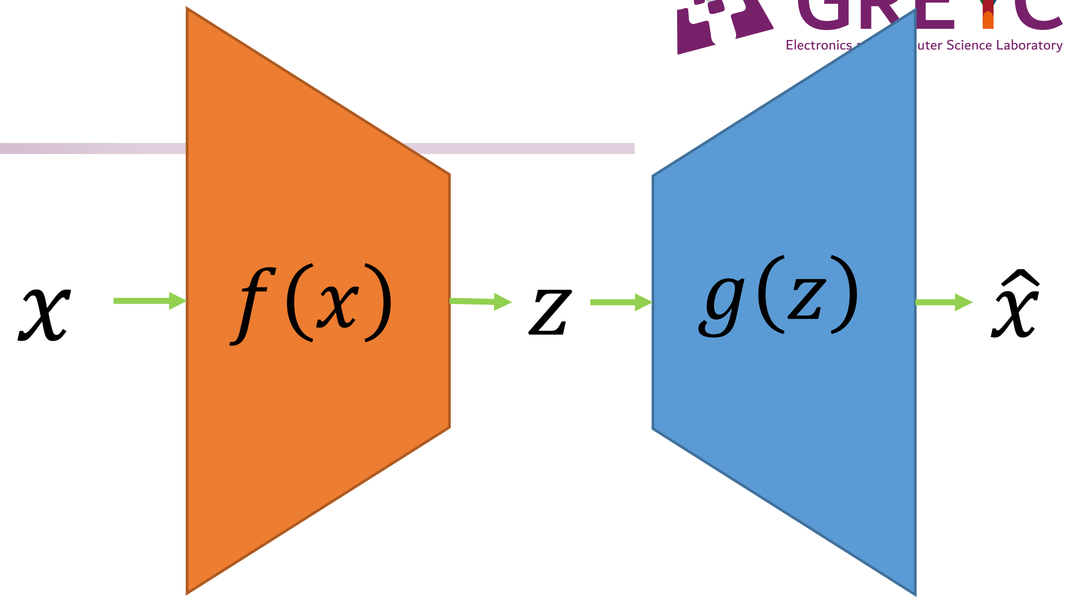
Uses

- The autoencoder idea was a part of NN history for decades (LeCun et al, 1987).
- Traditionally an autoencoder is used for dimensionality reduction and feature learning.
- Recently, the connection between autoencoders and latent space modeling has brought autoencoders to the front of generative modeling.

- **Not used for compression.**
  - **-Data specific compression.**
  - **-Lossy.**

Simple Idea

- Given data x (no labels) we would like to learn the functions f (encoder) and g (decoder) where:



$$f(x)=s(wx+b)=z$$

and

$$g(z)=s(w^{\prime} z+b^{\prime})=\hat{x}$$

s.t $h(x)=g(f(x))=\hat{x}$

where h is an approximation of the identity function.

(z is some **latent** representation or **code** and s is a non-linearity such as the sigmoid)

($\hat{x}$ is x's reconstruction)

Simple Idea

Learning the identity function seems trivial, but with added constraints on the network (such as limiting the number of hidden neurons or regularization) we can learn information about the structure of the data.

Trying to capture the distribution of the data (data specific!)

Training the AE

- Using **Gradient Descent** we can simply train the model as any other FC NN with:
  - Traditionally with squared error loss function

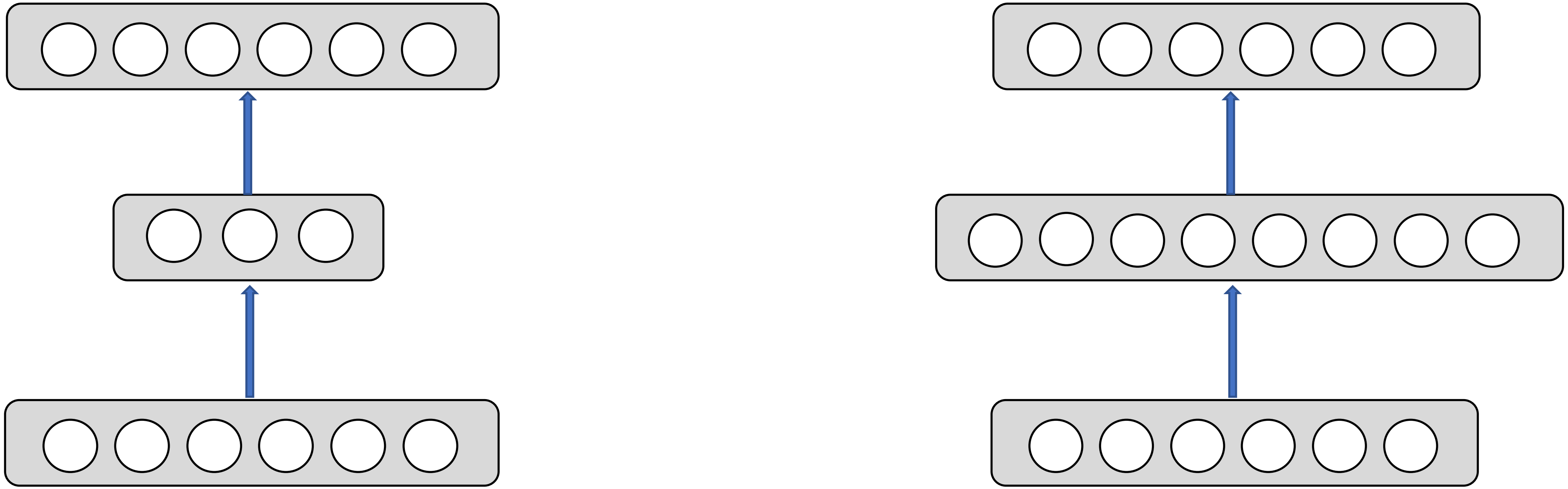
$$L(x, \hat{x}) = ||x - \hat{x}||^2$$

  - If our input is interpreted as bit vectors or vectors of bit probabilities the cross entropy can be used

$$H(p, q) = -\sum_{x} p(x) \log q(x)$$

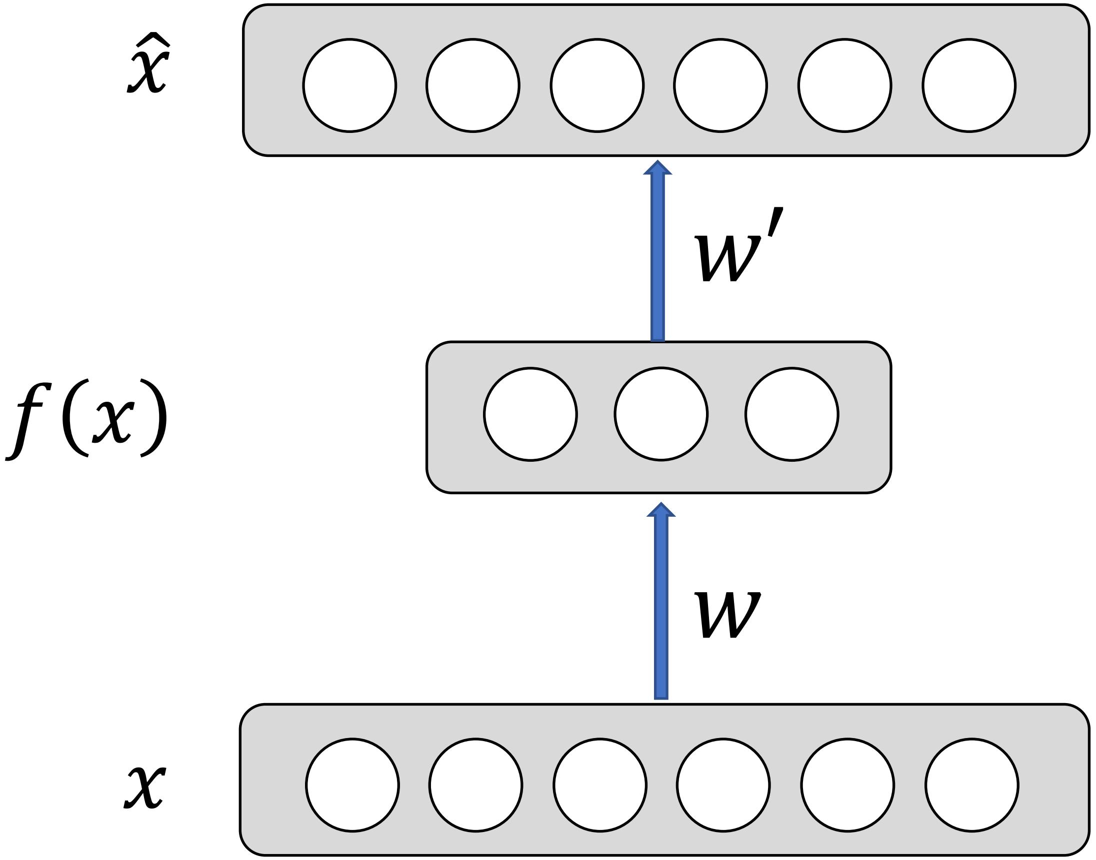
Undercomplete AE VS overcomplete AE

- We distinguish between two types of AE structures:



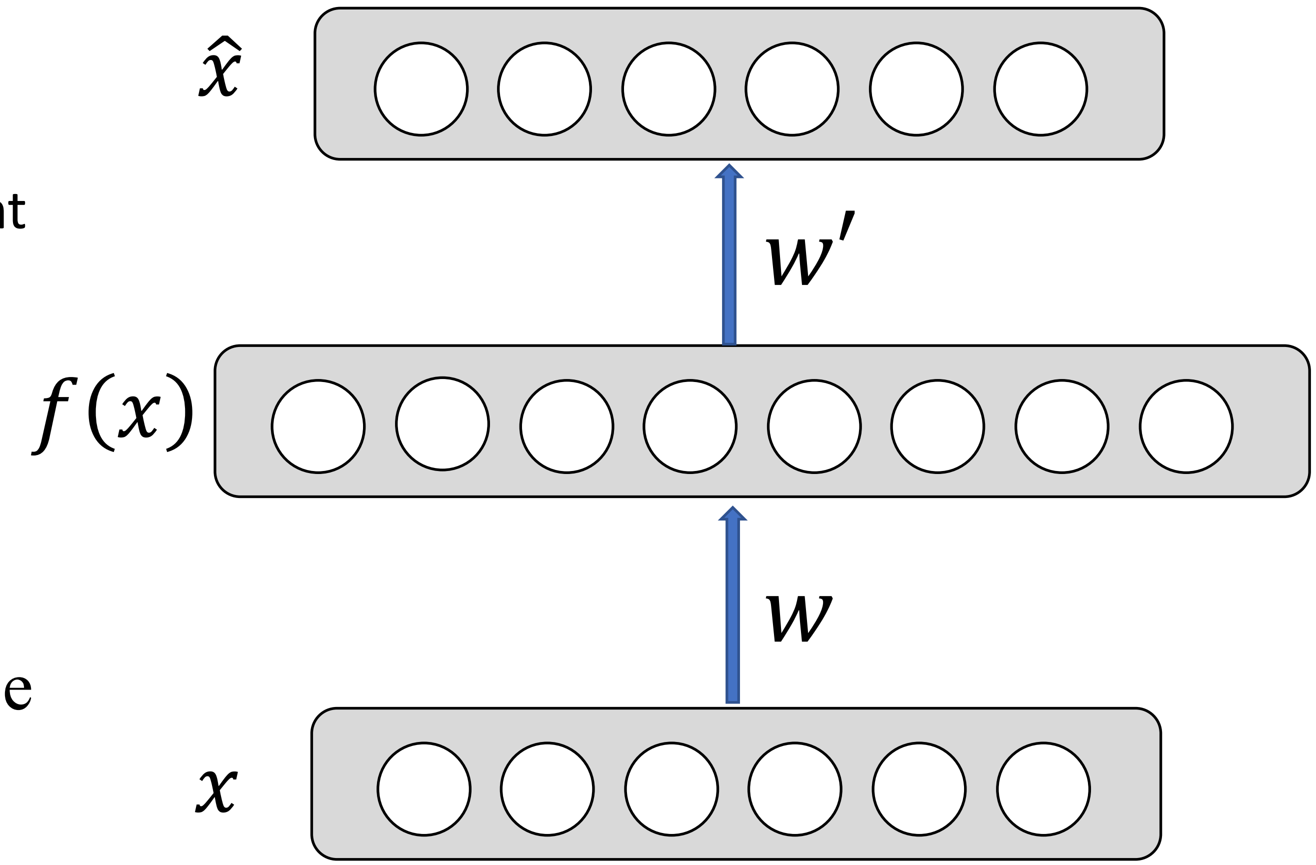
Undercomplete AE

- Hidden layer is **Undercomplete** if smaller than the input layer
  - Compresses the input
  - Compresses well only for the training dist.
- Hidden nodes will be
  - Good features for the training distribution.
  - Bad for other types on input

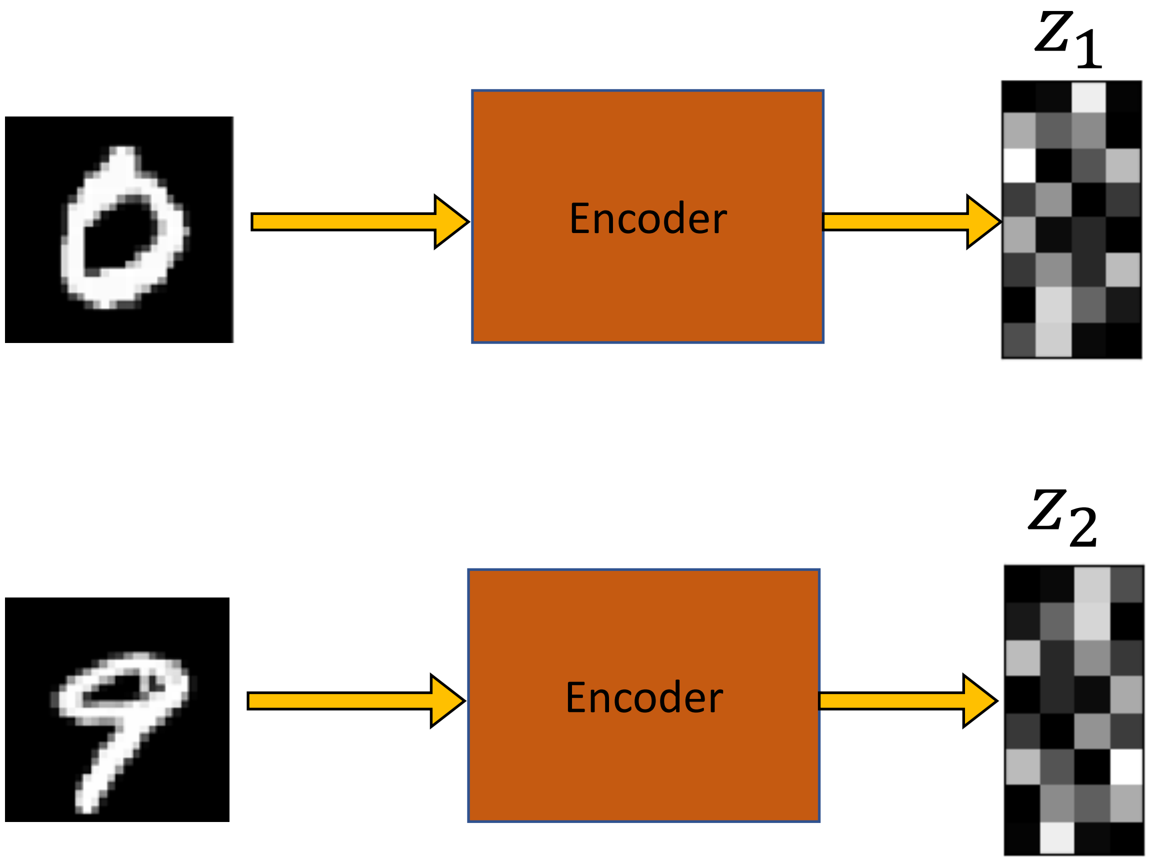


Overcomplete AE

- Hidden layer is **Overcomplete** if greater than the input layer
  - No compression in hidden layer.
  - Each hidden unit could copy a different input component
- No guarantee that the hidden units will extract meaningful structure
- Adding dimensions is good for training a linear classifier (XOR case example).
- A higher dimension code helps model a more complex distribution.

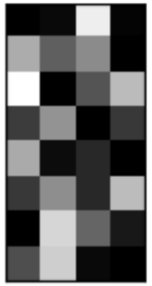
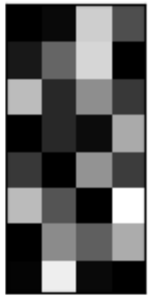


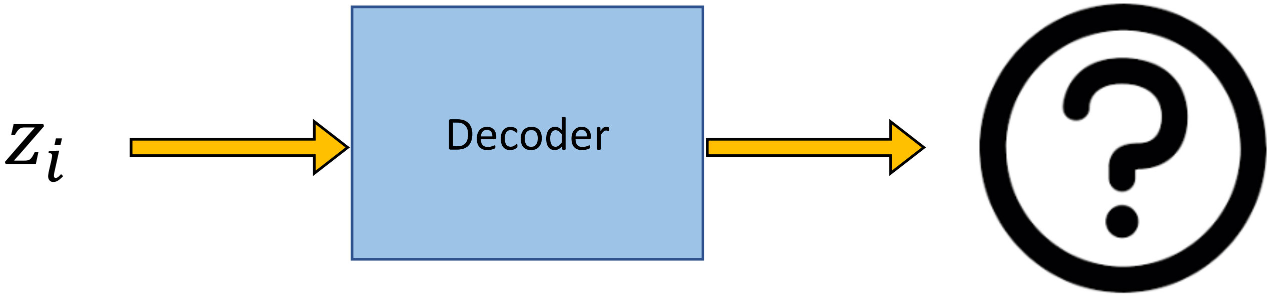
Simple latent space interpolation



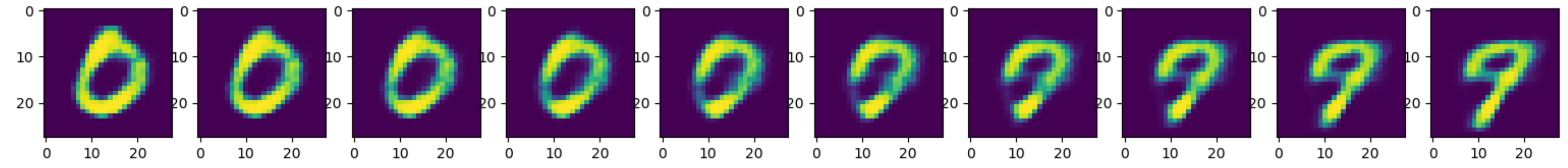
Simple latent space interpolation

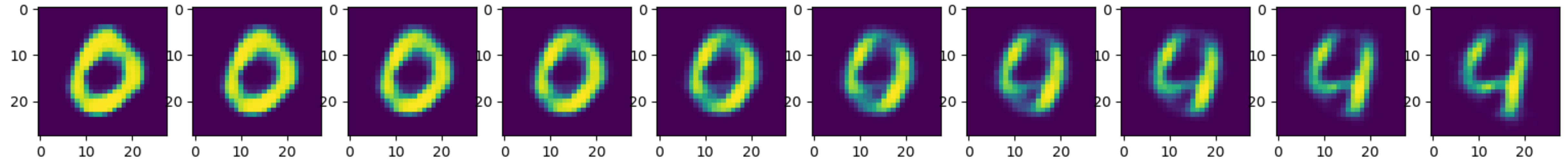
z_1 z_2

$z_i = \alpha$  $+ (1 - \alpha)$ 

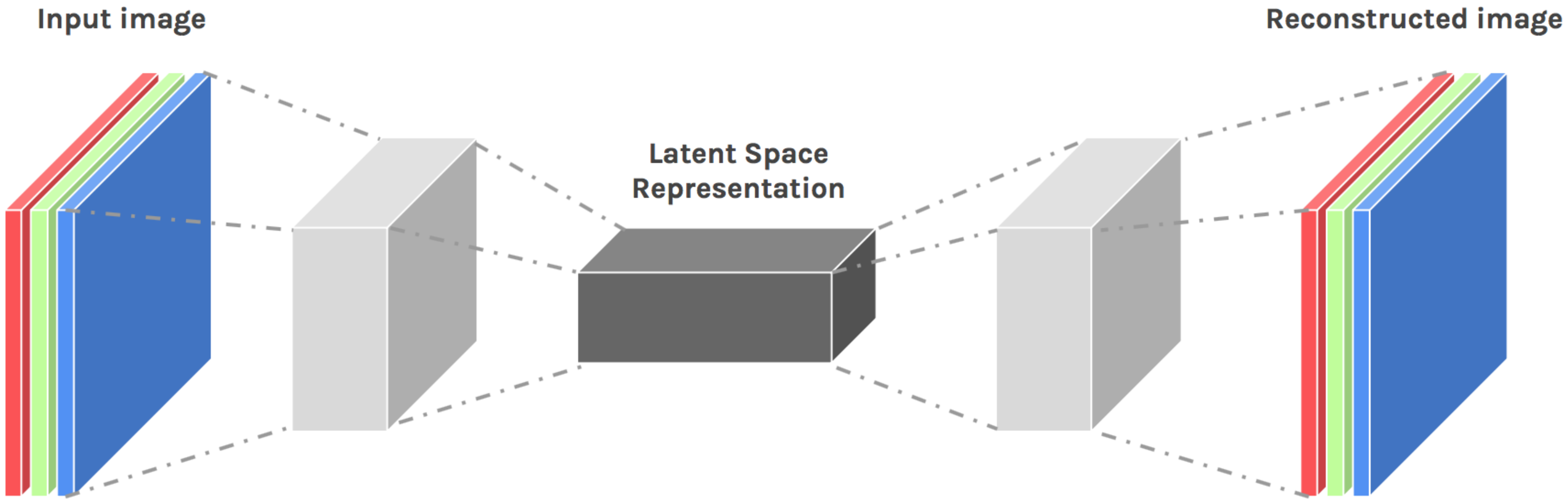


Simple latent space interpolation



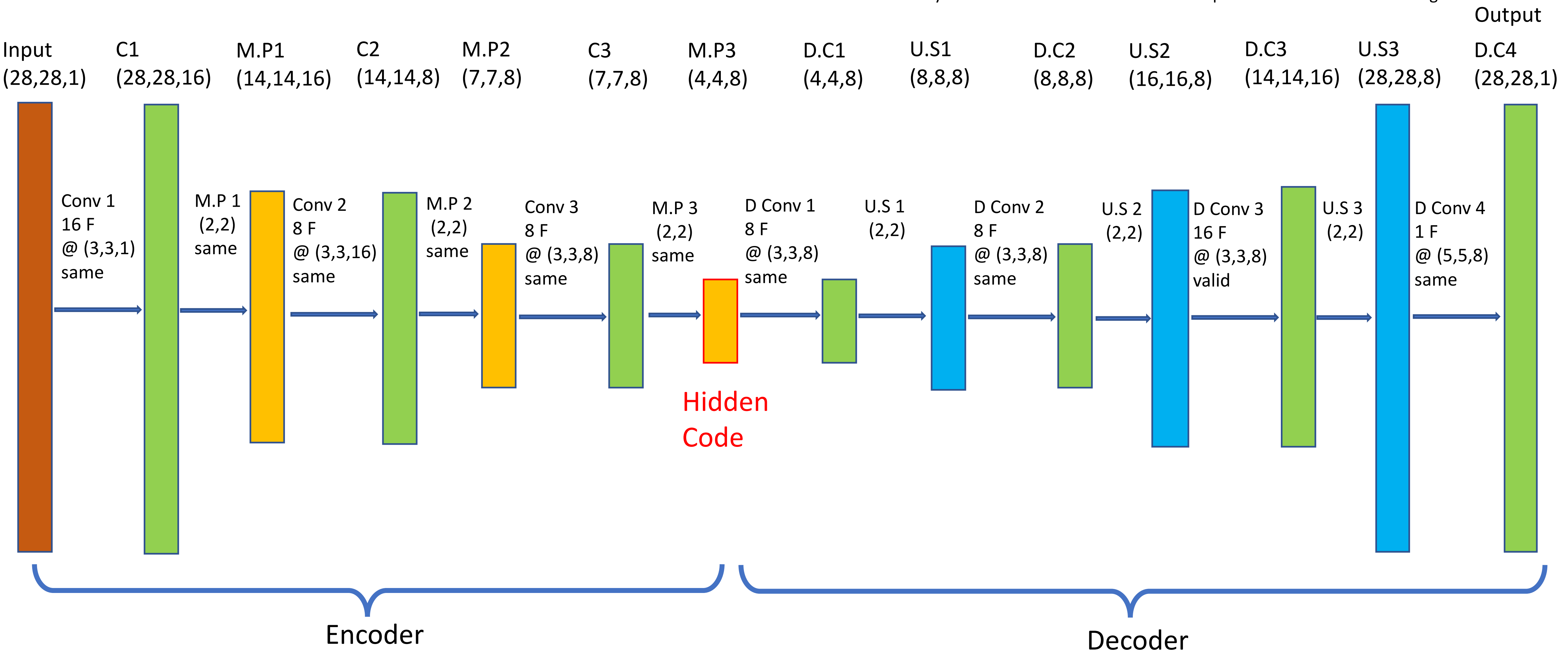


Convolutional AE



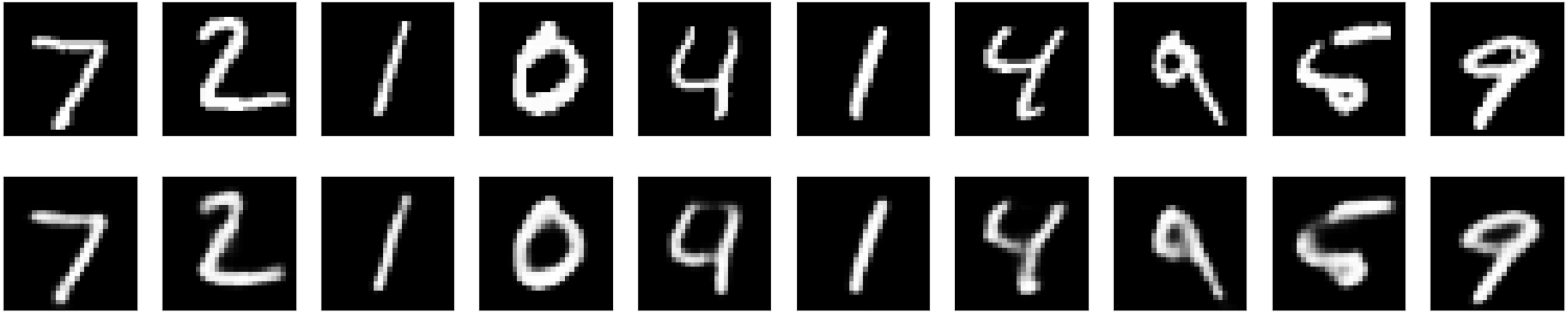
Convolutional AE

* Input values are normalized
* All of the conv layers activation functions are relu except for the last conv which is sigm



Convolutional AE – Keras example results

- 50 epochs.
- 88% accuracy on validation set.



Regularization

▸Motivations

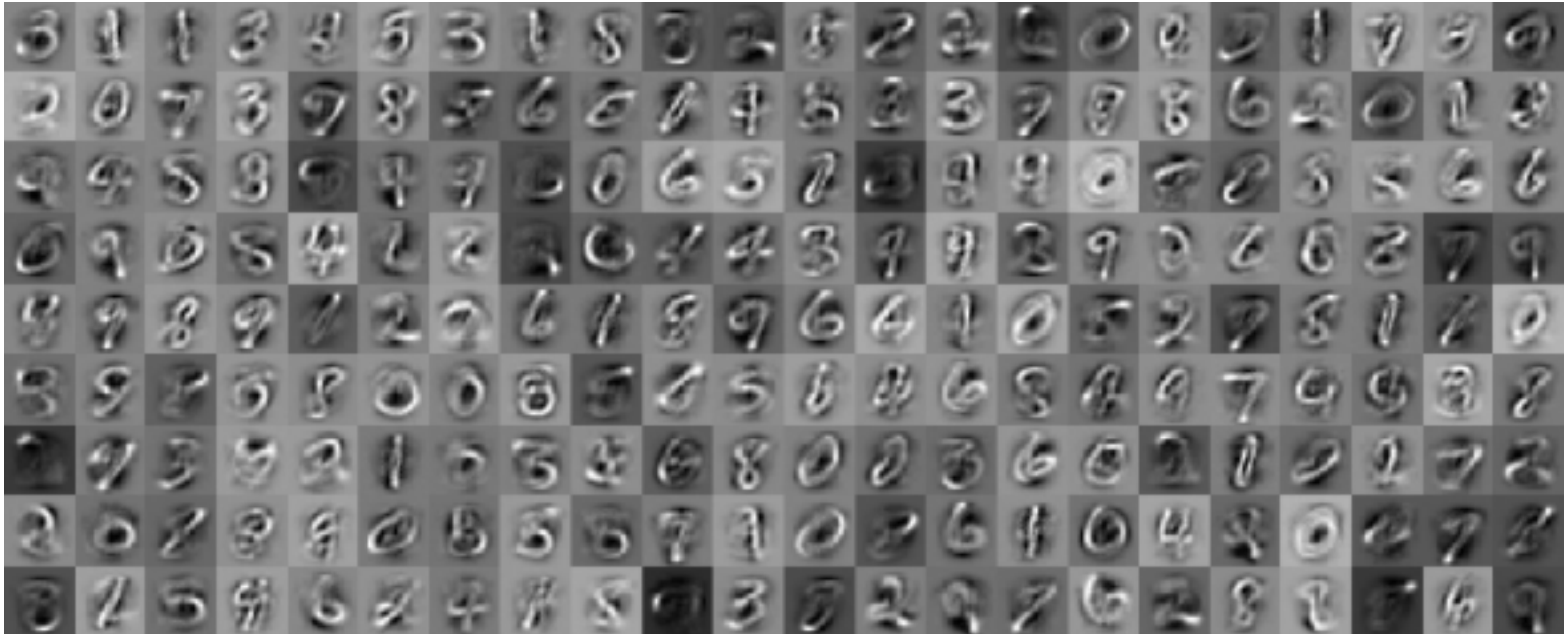
- We would like to learn meaningful features **without** altering the code's dimensions (Overcomplete or Undercomplete)
- We would like to avoid uninteresting solutions

▸The solution: **imposing other constraints on the network**.

Sparsely Regulated Autoencoders

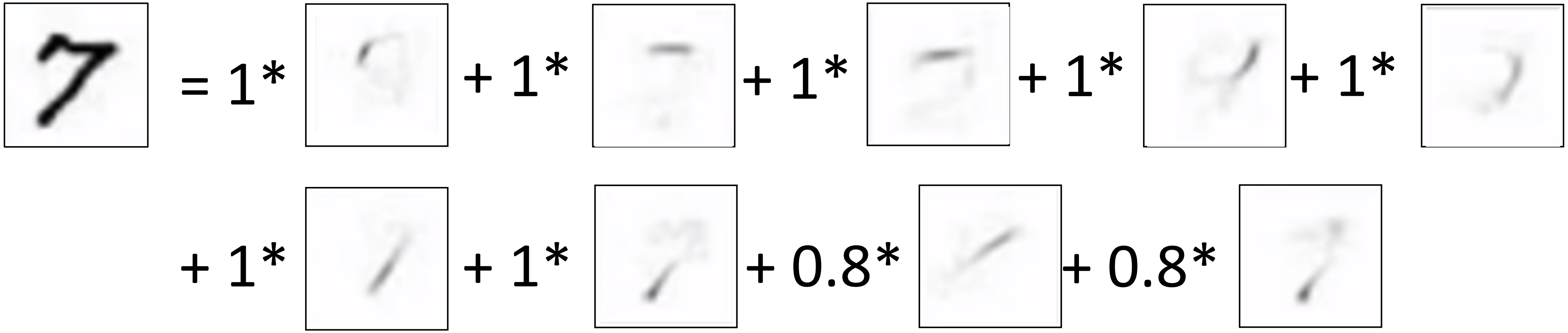
A bad example:

Activation Maps



Sparsely Regulated Autoencoders

- We want our learned features to be as sparse as possible.
- With sparse features we can generalize better.



Sparsely Regulated Autoencoders

- Recall:
  - $a_j^{(\mathrm{Bn})}$ is defined to be the activation of the jth hidden unit (bottleneck) of the autoencoder.
  - Let $a_j^{(\mathrm{Bn})}(x)$ be the activation of this specific node on a given input x.
  - Let

$$\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m}\left[a_j^{(\mathrm{Bn})}\left(x^{(i)}\right)\right]$$

  be the average activation hidden unit *j* (over the training)

- We would like to force the constraint

$$\hat{\rho}_j = \rho$$

  where ρ is a “sparsity parameter”, typically small.
- In other words, we want the average activation of each neuron j to be close to ρ.

Sparsely Regulated Autoencoders

- We need to penalize $\hat{\rho}_j$ for deviating from ρ.
- Many choices of the penalty term will give reasonable results.
- For example:

$$\sum_{j=1}^{Bn} KL(\rho|\hat{\rho}_j)$$

where $KL(\rho|\hat{\rho}_j)$ is a Kullback-Leibler divergence function.

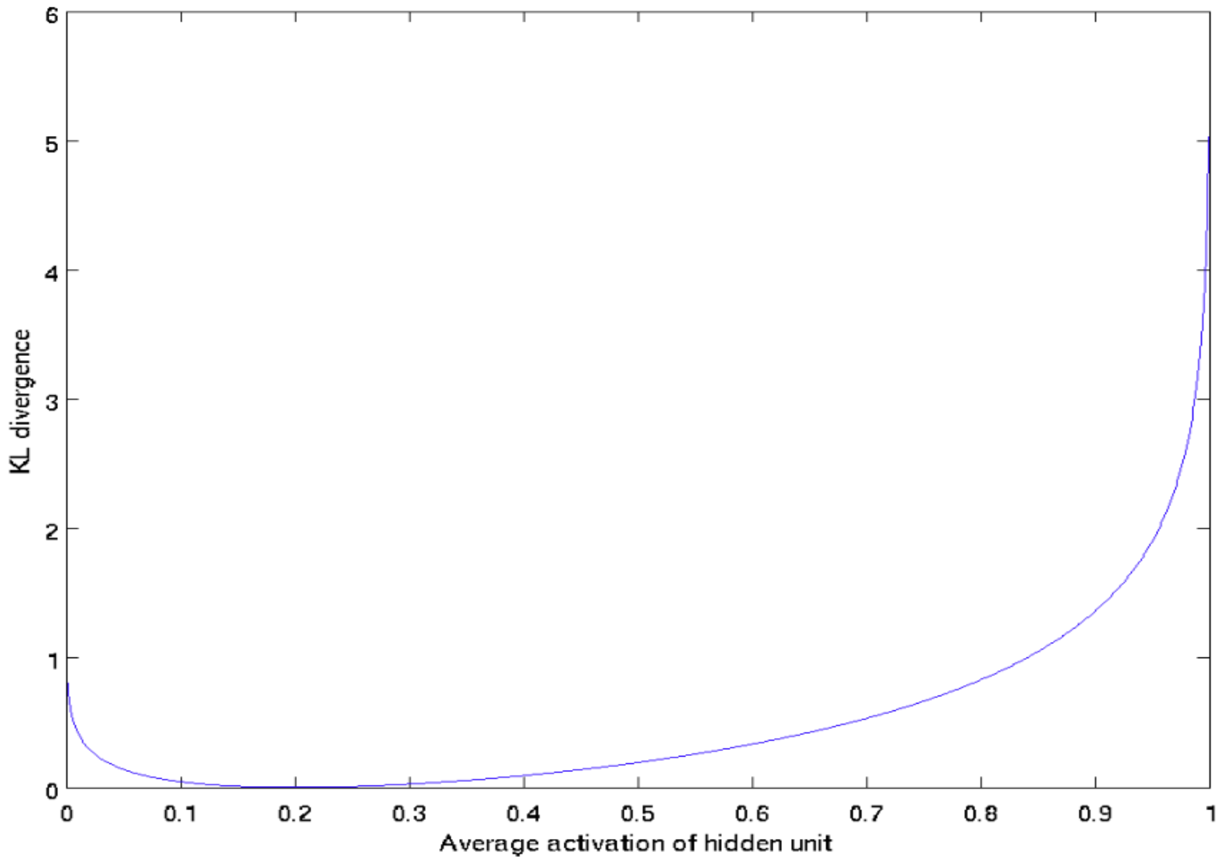
Sparsely Regulated Autoencoders

Reminder

- KL is a standard function for measuring how different two distributions are, which has the properties:

$$KL\left(\rho|\hat{\rho}_j\right) = 0 \text{ if } \hat{\rho}_j = \rho$$

otherwise it is increased monotonically.



$\rho = 0.2$

Sparsely Regulated Autoencoders

▶Our overall cost functions is now:

$$J_S(W, b) = J(W, b) + \beta \sum_{j=1}^{Bn} KL(p|\hat{\rho}_j)$$

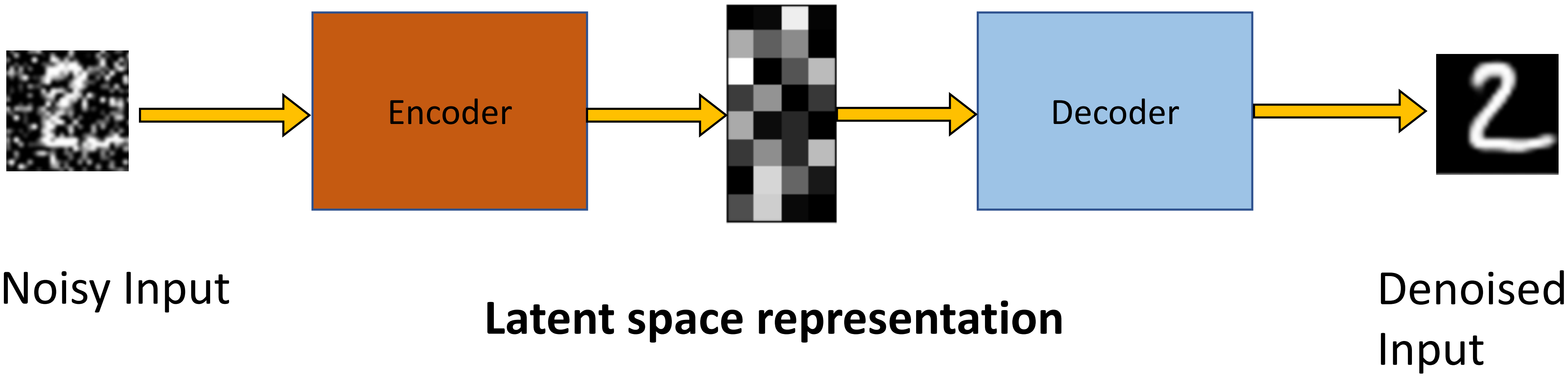
*Note: We need to know $\hat{\rho}_j$ before hand, so we have to compute a forward pass on all the training set.

Denoising Autoencoders

<u>Intuition:</u>

- We still aim to encode the input and to NOT mimic the identity function.
- We try to undo the effect of corruption process stochastically applied to the input.

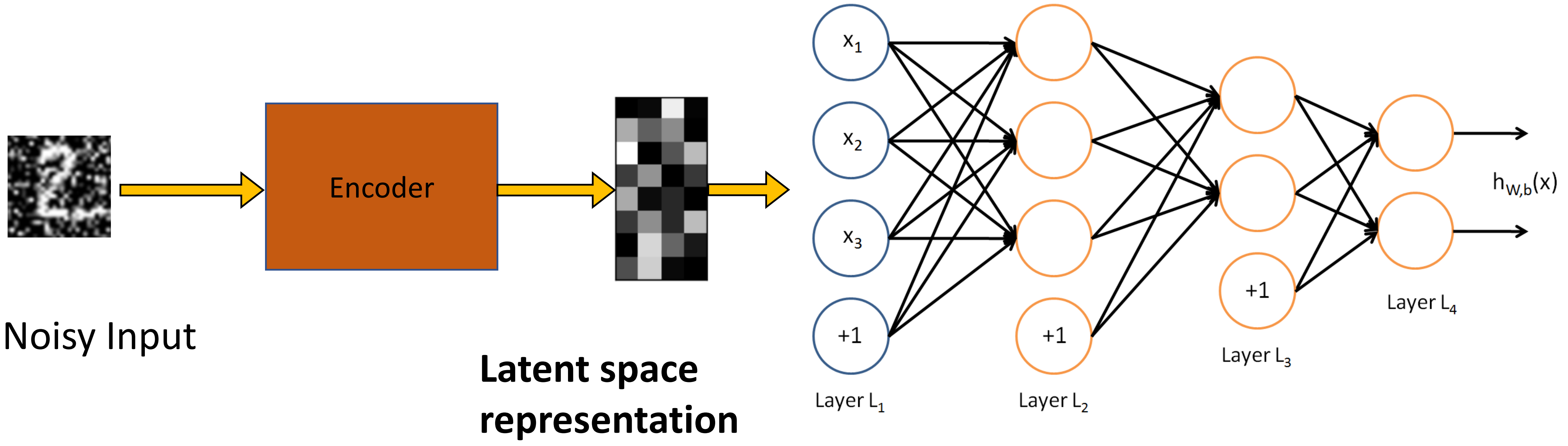
A more robust model



Denoising Autoencoders

Use Case:

▶ Extract robust representation for a NN classifier.



Denoising Autoencoders

Instead of trying to mimic the identity function by minimizing:

$$L\left(x, g\left(f(x)\right)\right)$$

where L is some loss function

A **DAE** instead minimizes:

$$L\left(x, g\left(f(\tilde{x})\right)\right)$$

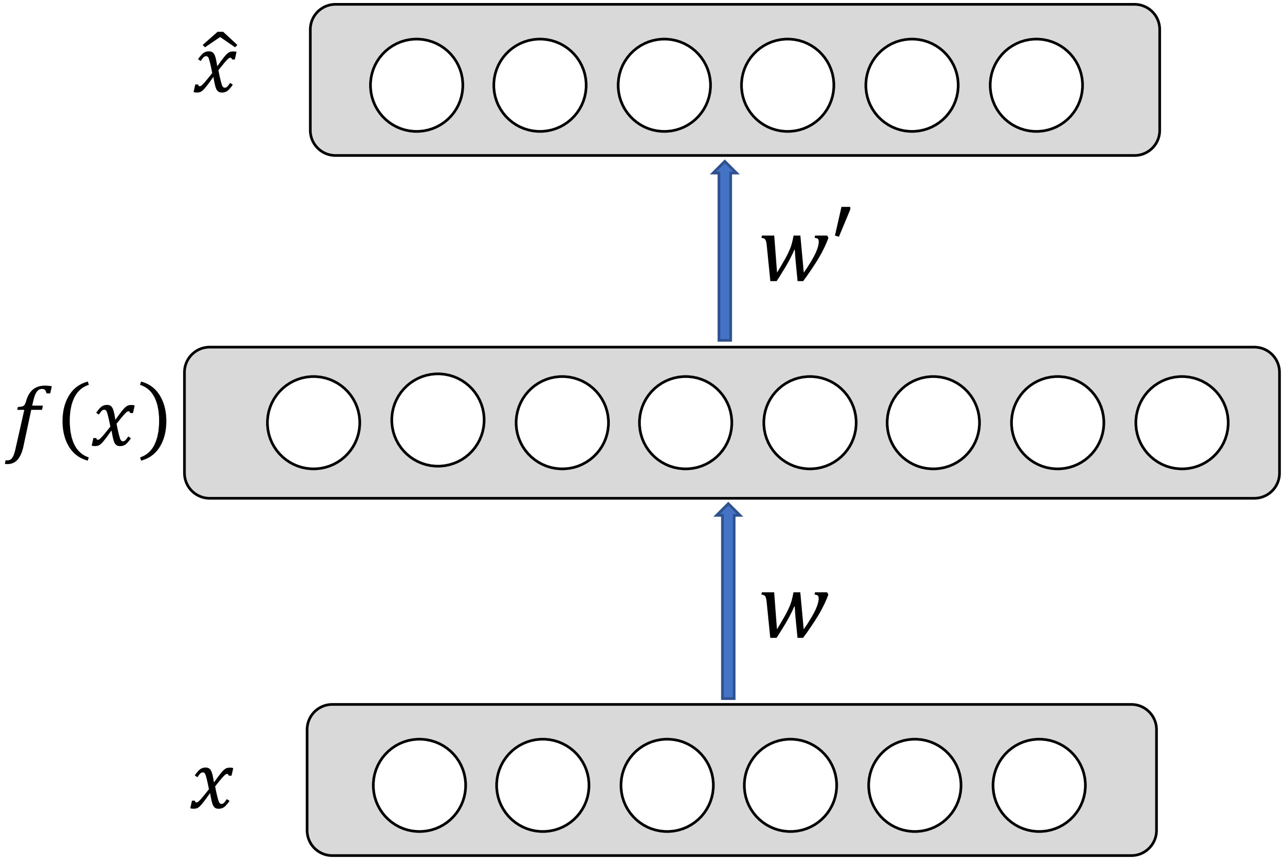
where $\tilde{x}$ is a copy of x that has been corrupted by some form of noise.

Denoising Autoencoders

Idea: A robust representation against noise

- Random assignment of subset of inputs to 0, with probability v.
- Gaussian additive noise.

(a) (b)

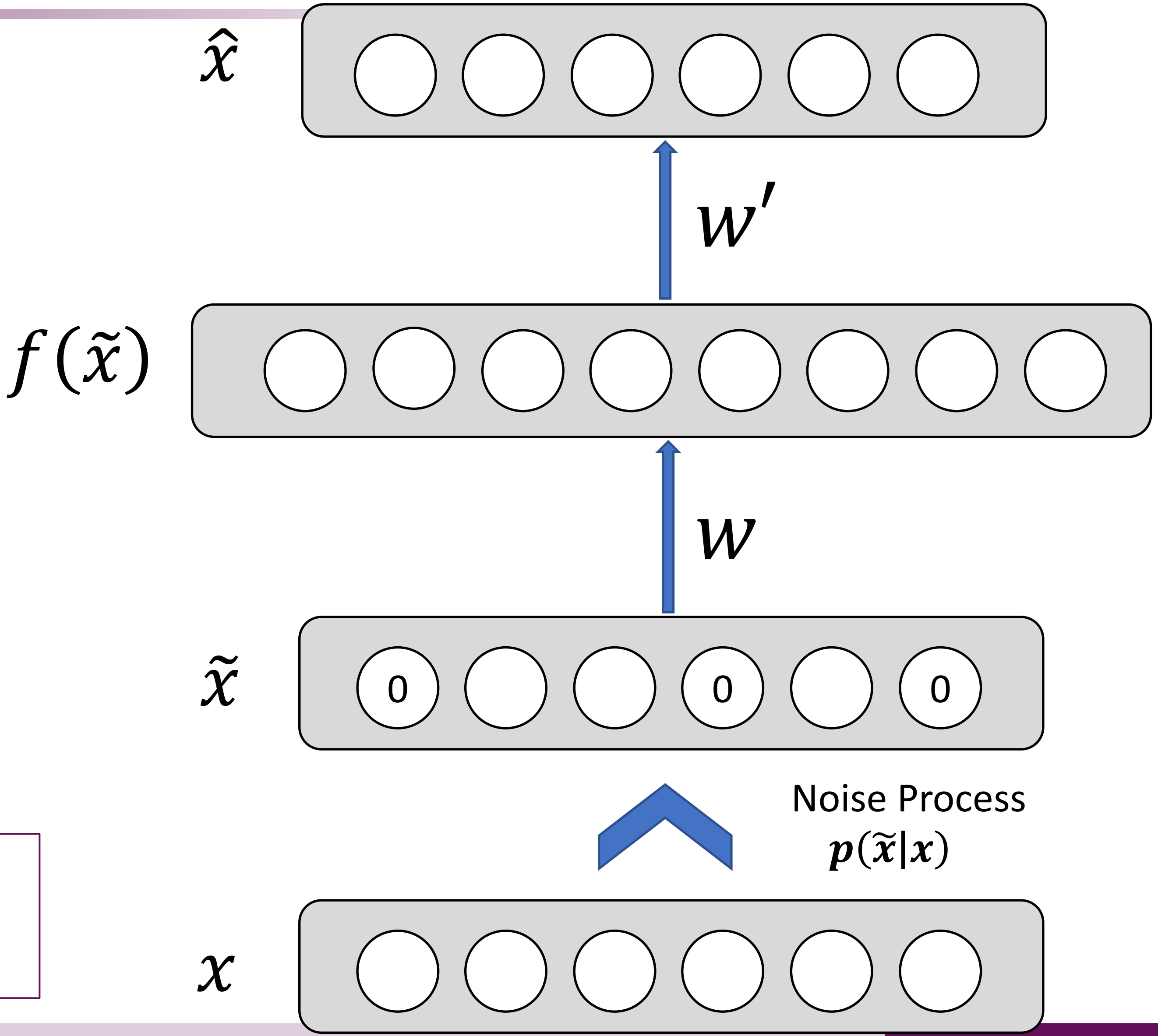


Denoising Autoencoders

- Reconstruction $\hat{x}$ computed from the corrupted input $\tilde{x}$.
- Loss function compares $\hat{x}$ reconstruction with the noiseless x.

- The autoencoder cannot fully trust each feature of x independently so it must learn the correlations of x's features.
- Based on those relations we can predict a more 'not prune to changes' model.

We are forcing the hidden layer to learn a generalized structure of the data.



Denoising Autoencoders - process

Taken some input x

Apply Noise

$\tilde{x}$

Denoising Autoencoders - process

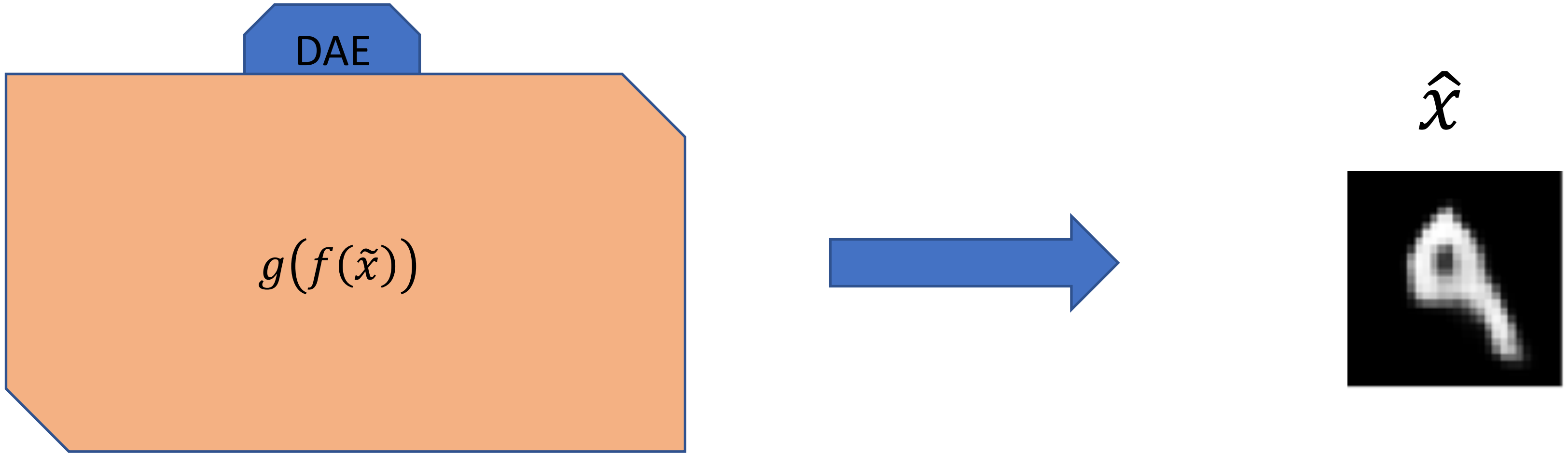
$\tilde{x}$

Encode And Decode

DAE

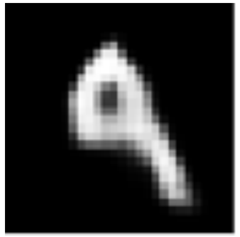
$g\left(f(\tilde{x})\right)$

Denoising Autoencoders - process



Denoising Autoencoders - process

$\hat{x}$

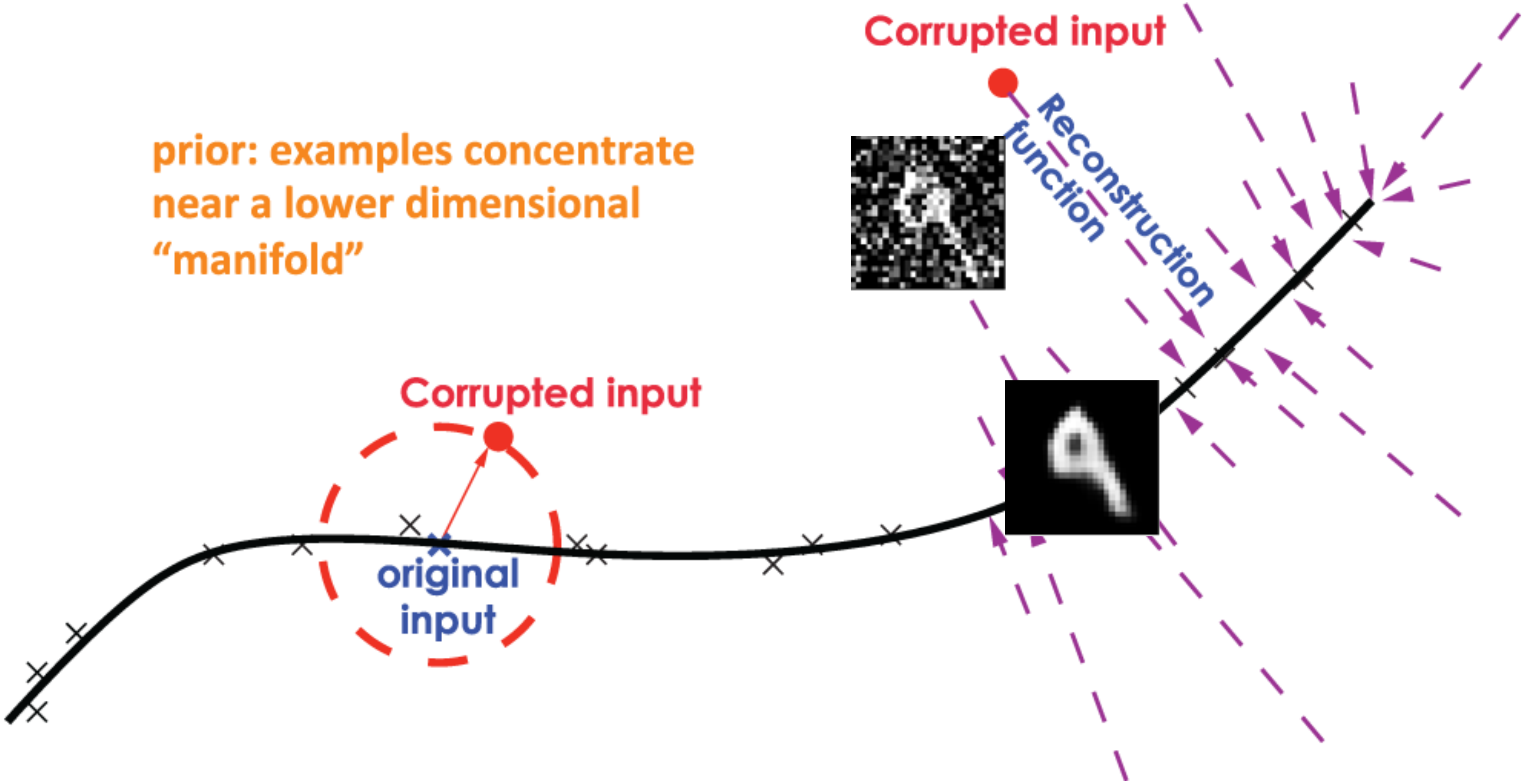


Compare

$=$

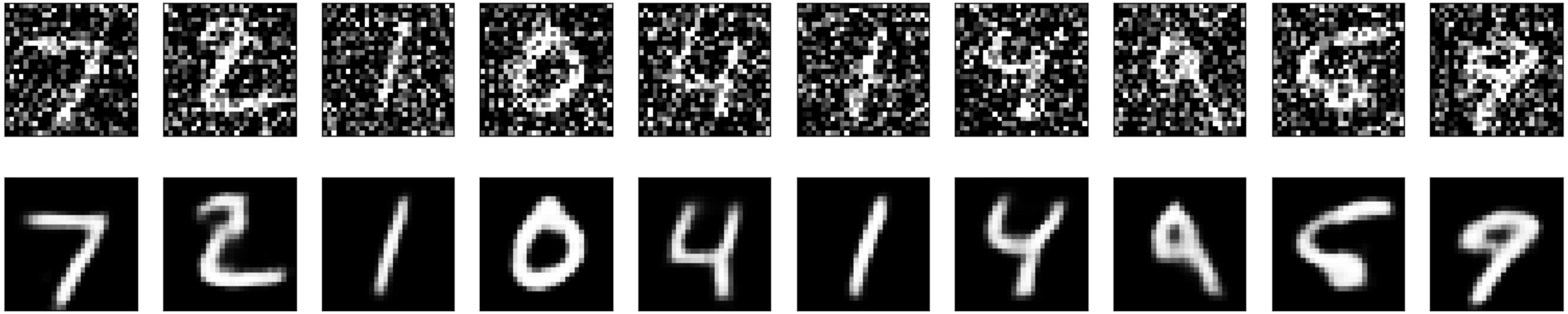
x

Denoising autoencoders



Denoising convolutional AE – keras

- 50 epochs.
- Noise factor 0.5
- 92% accuracy on validation set.



Stacked AE

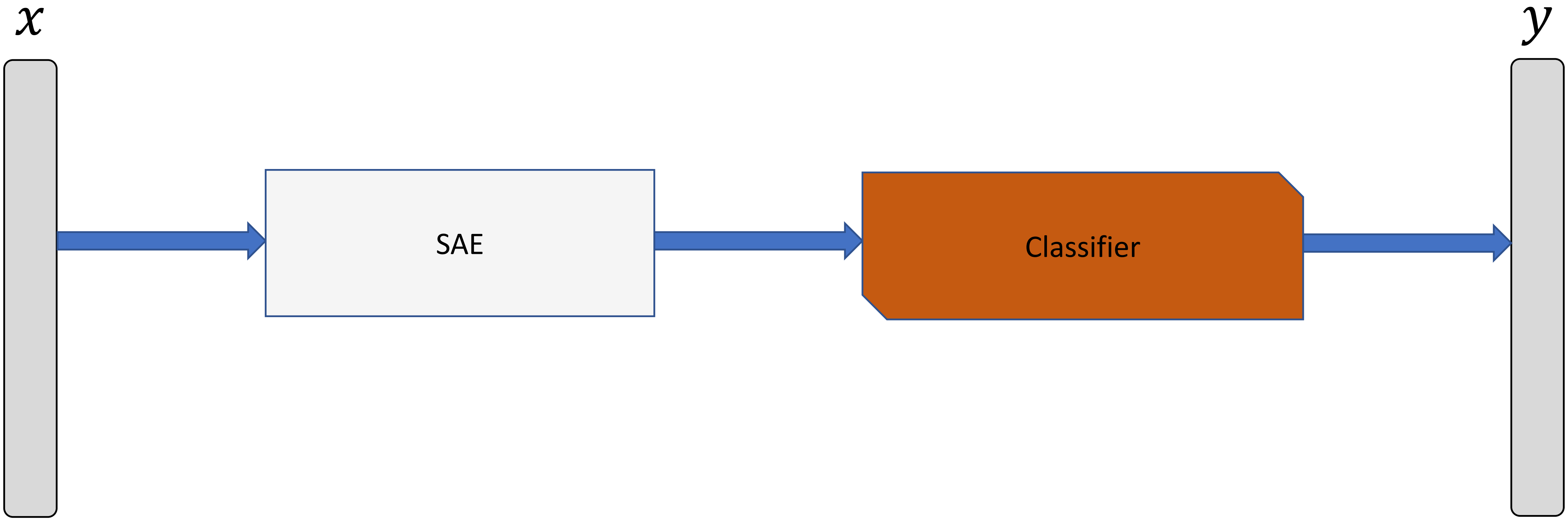
Motivations

- We want to harness the feature extraction quality of a AE for our advantage.
- *For example:* we can build a deep supervised classifier where it's input is the output of a SAE.
- The benefit: our deep model's W are not randomly initialized but are rather "smartly selected"
- Also using this unsupervised technique lets us have a larger unlabeled dataset.

Stacked AE

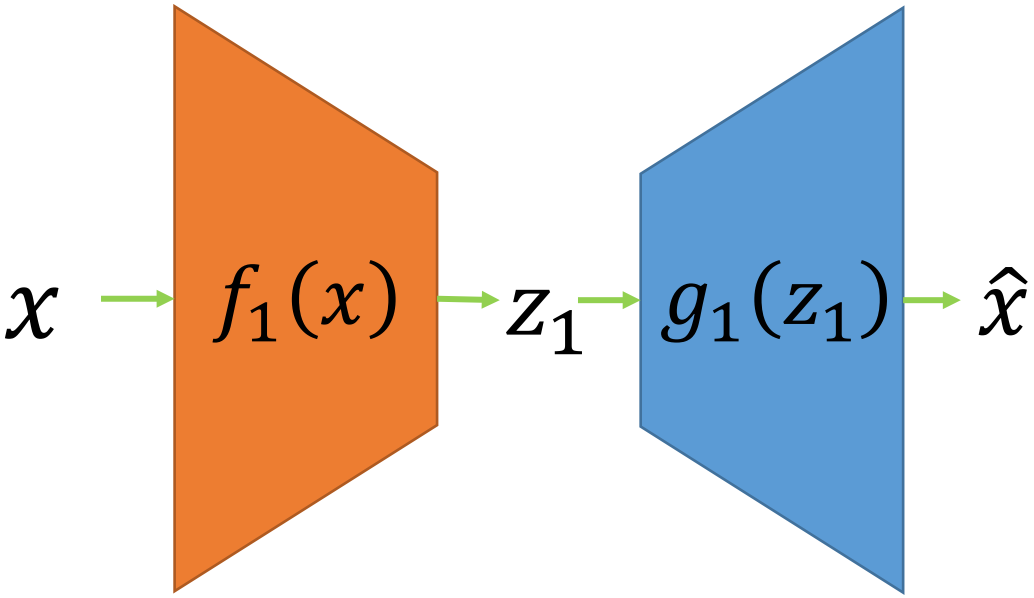
- Building a SAE consists of two phases:
  1. Train each AE layer one after the other.
  2. Connect any classifier (SVM / FC NN layer etc.)

Stacked AE



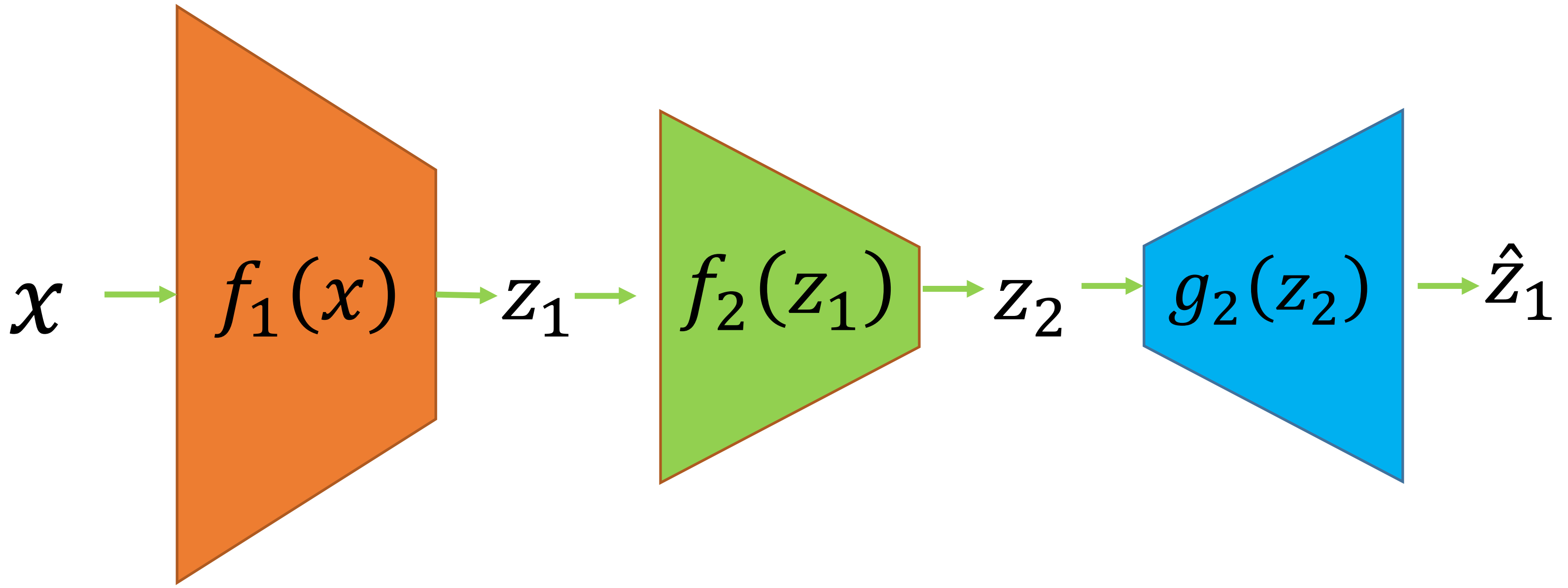
Stacked AE – train process

First Layer Training (AE 1)



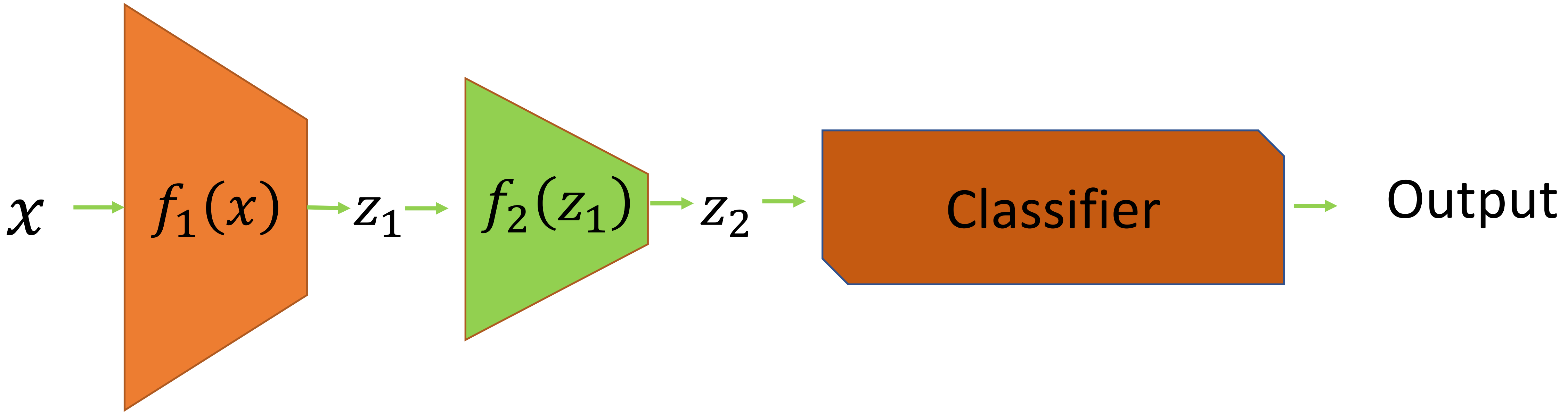
Stacked AE – train process

Second Layer Training (AE 2)



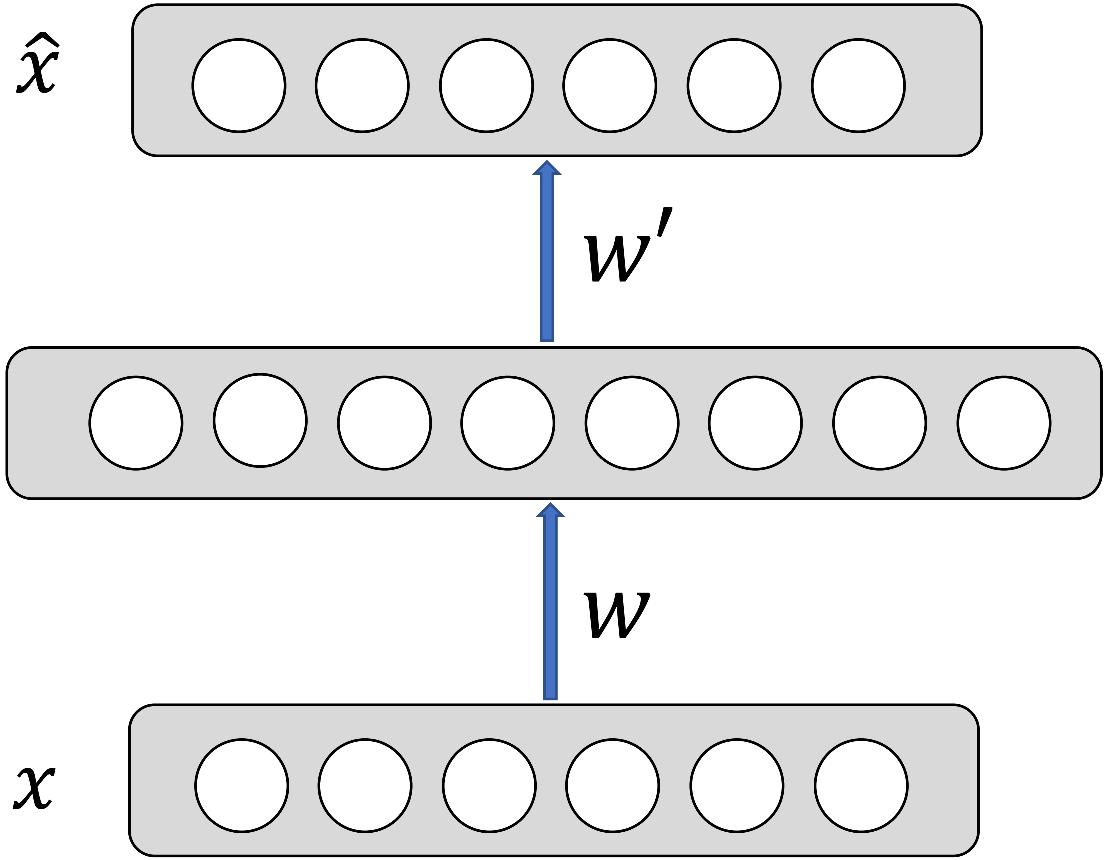
Stacked AE – train process

Add any classifier



Contractive autoencoders

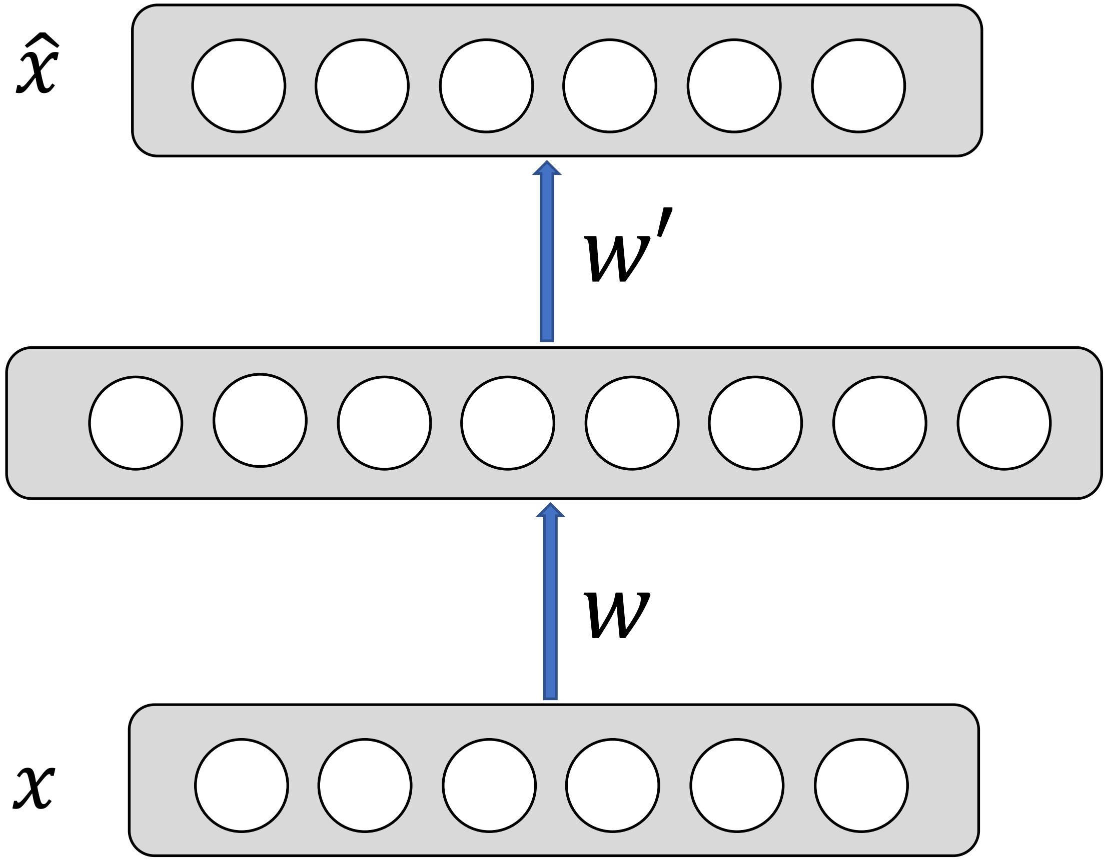
- We are still trying to avoid uninteresting features.
- Here we add a regularization term $\Omega(x)$ to our loss function to limit the hidden layer.



Contractive autoencoders

- Idea: We wish to extract features that **only** reflect variations observed in the training set. We would like to be invariant to the other variations.

Points close to each other in the input space maintain that property in the latent space.



Contractive autoencoders

▸Definitions and reminders:

- Frobenius norm (L2): $\|A\|_F = \sqrt{\Sigma_{i,j} \left| a_{ij} \right|^2}$

- Jacobian Matrix: $J_f(x) = \frac{\partial f(x)}{\partial x} = \begin{bmatrix} \frac{\partial f(x)_1}{\partial x_1} & \cdots & \frac{\partial f(x)_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f(x)_m}{\partial x_1} & \cdots & \frac{\partial f(x)m}{\partial x_n} \end{bmatrix}$

Contractive autoencoders

▶Our new loss function would be:

$$L^{*}(x) = L(x) + \lambda\Omega(x)$$

where $\Omega(x) = \|J_f(x)\|_F^2$ or simply: $\sum_{i,j}\left(\frac{\partial f(x)_j}{\partial x_i}\right)^2$

and where λ controls the balance of our reconstruction objective and the hidden layer "flatness".

Contractive autoencoders

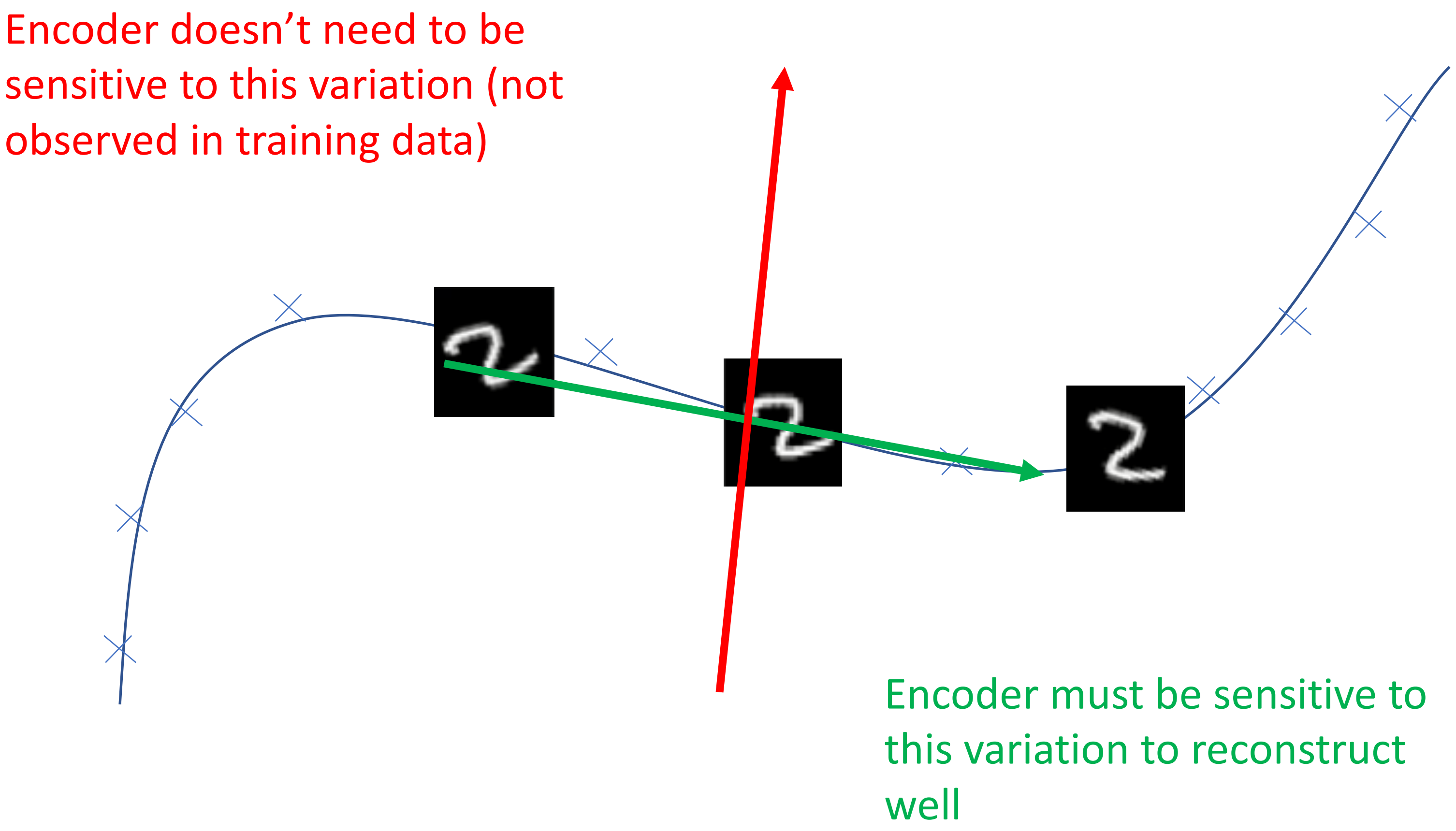
▶Our new loss function would be:

$$L^*(x) = L(x) + \lambda\Omega(x)$$

- $L(x)$ would be an encoder that keeps good information ($\lambda \to 0$)
- $\Omega(x)$ would be an encoder that throws away all information ($\lambda \to \infty$)

Combination would be an encoder that keeps **only** good information.

Contractive autoencoders



Which autoencoder?

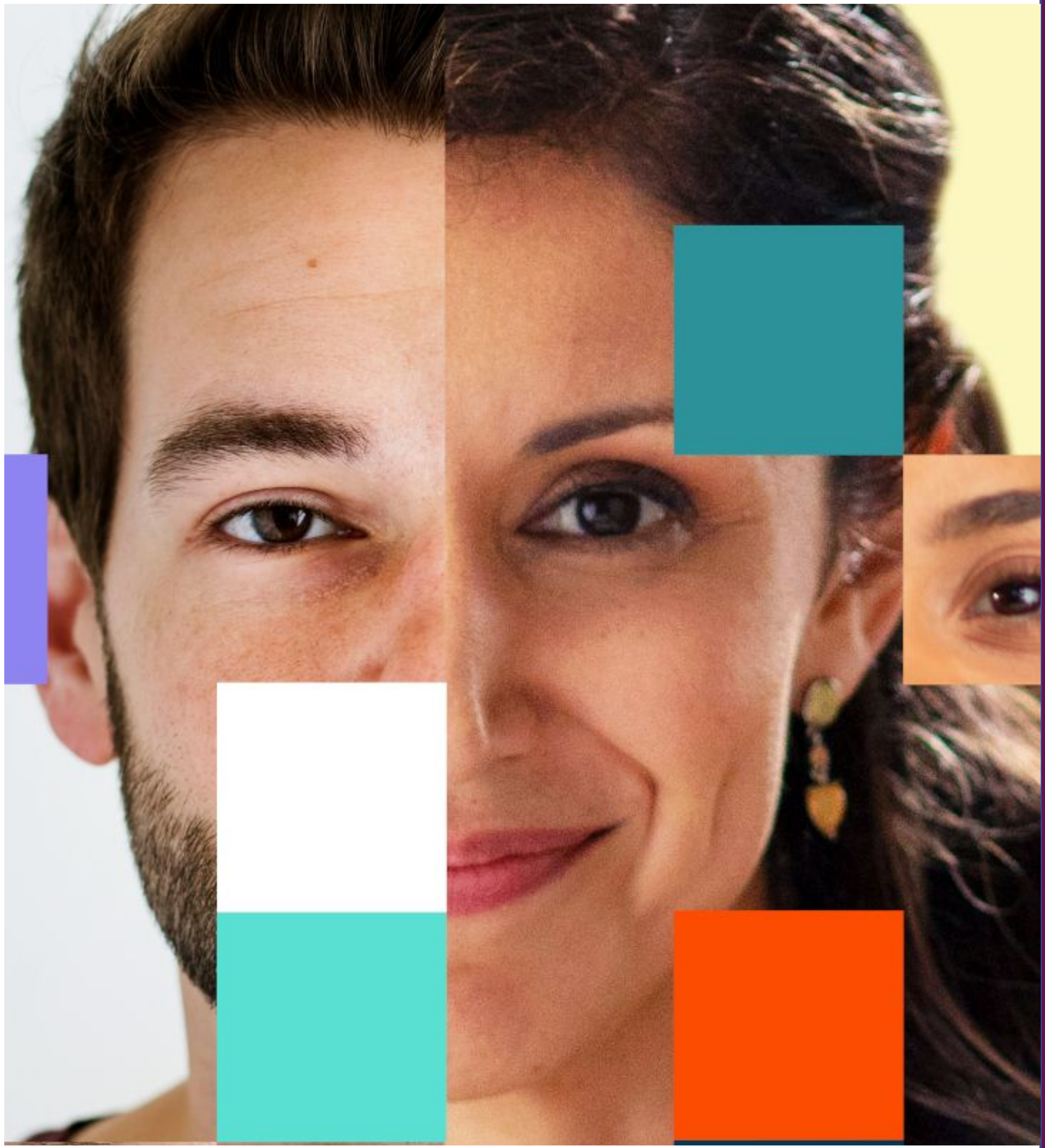
- DAE make the **reconstruction function** resist small, finite sized perturbations in input.
- CAE make the **feature encoding function** resist small, infinitesimal perturbations in input.

- Both denoising AE and contractive AE perform well!

Which autoencoder?

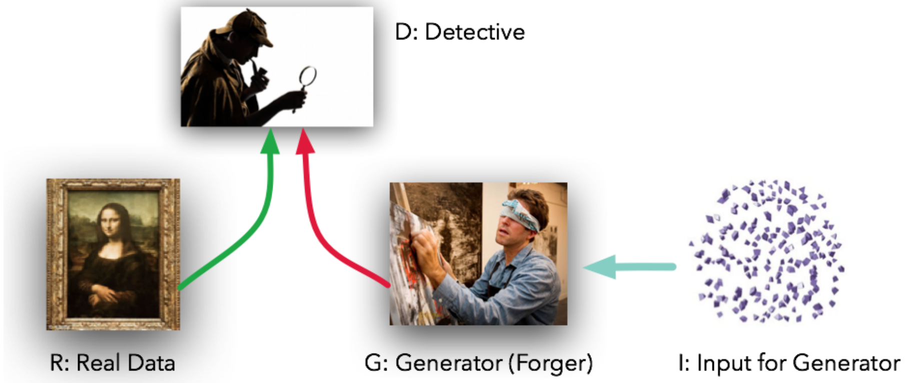
- Advantage of DAE: simpler to implement
  - Requires adding one or two lines of code to regular AE.
  - No need to compute Jacobian of hidden layer.

- Advantage of CAE: gradient is deterministic.
  - might be more stable than DAE, which uses a sampled gradient.
  - one less hyper-parameter to tune (noise-factor)



HOW IS GENERATED A DEEPFAKE?

Strategy 2: GAN



What a GAN is?

▶Generative

- Learn a generative model

▶Adversarial

- Trained in an adversarial setting

▶Networks

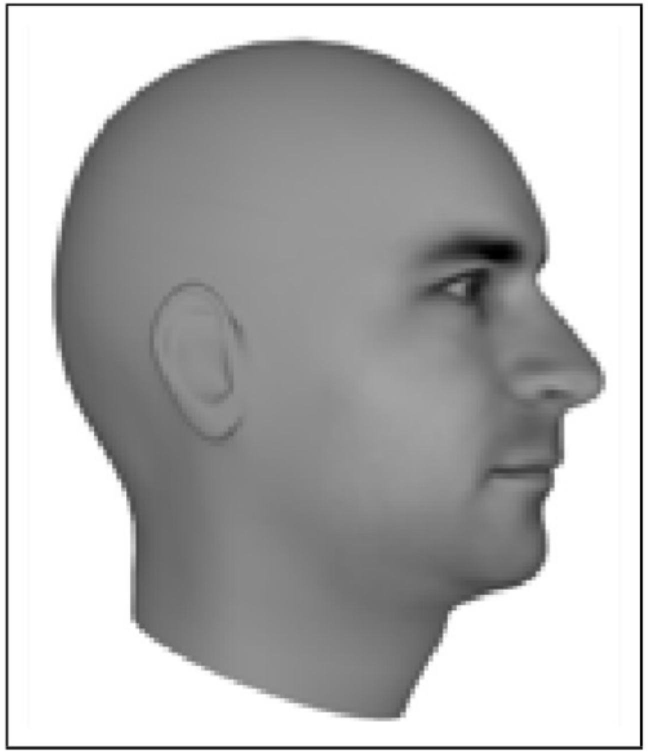
- Use Deep Neural Networks

Why Generative models?

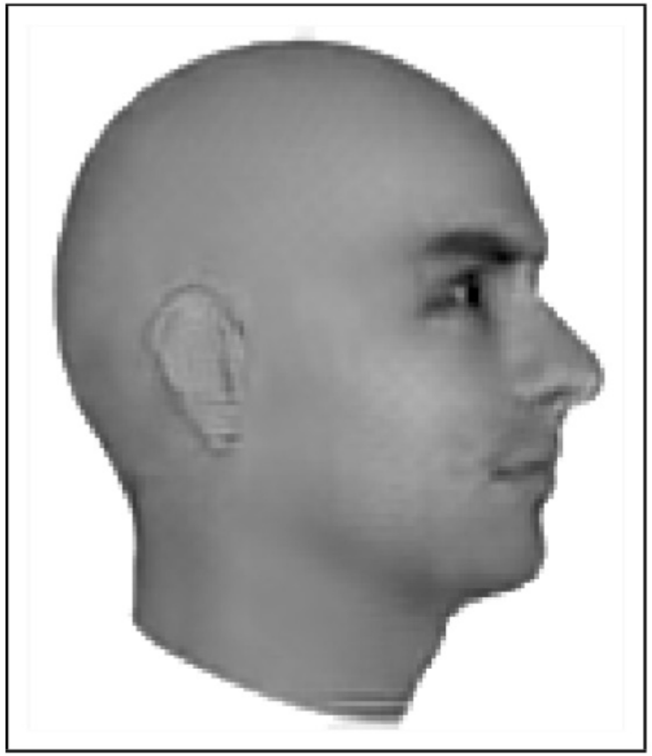
- Discriminative models
  - Given an image **X**, predict a label **Y**
  - Estimates **P(Y|X)**
- Discriminative models have several key limitations
  - Can't model **P(X)**, *i.e.* the probability of seeing a certain image **X**
  - Thus, can't sample from **P(X)**, *i.e.* **can't generate new images**
- Generative models (in general) cope with all of above
  - Can model **P(X)**
  - Can generate new images

Magic of GANs

Ground Truth



Adversarial



Lotter, William, Gabriel Kreiman, and David Cox. "Unsupervised learning of visual structure using predictive generative networks." arXiv preprint arXiv:1511.06380 (2015).

Magic of GANs

- Which one is computer generated?

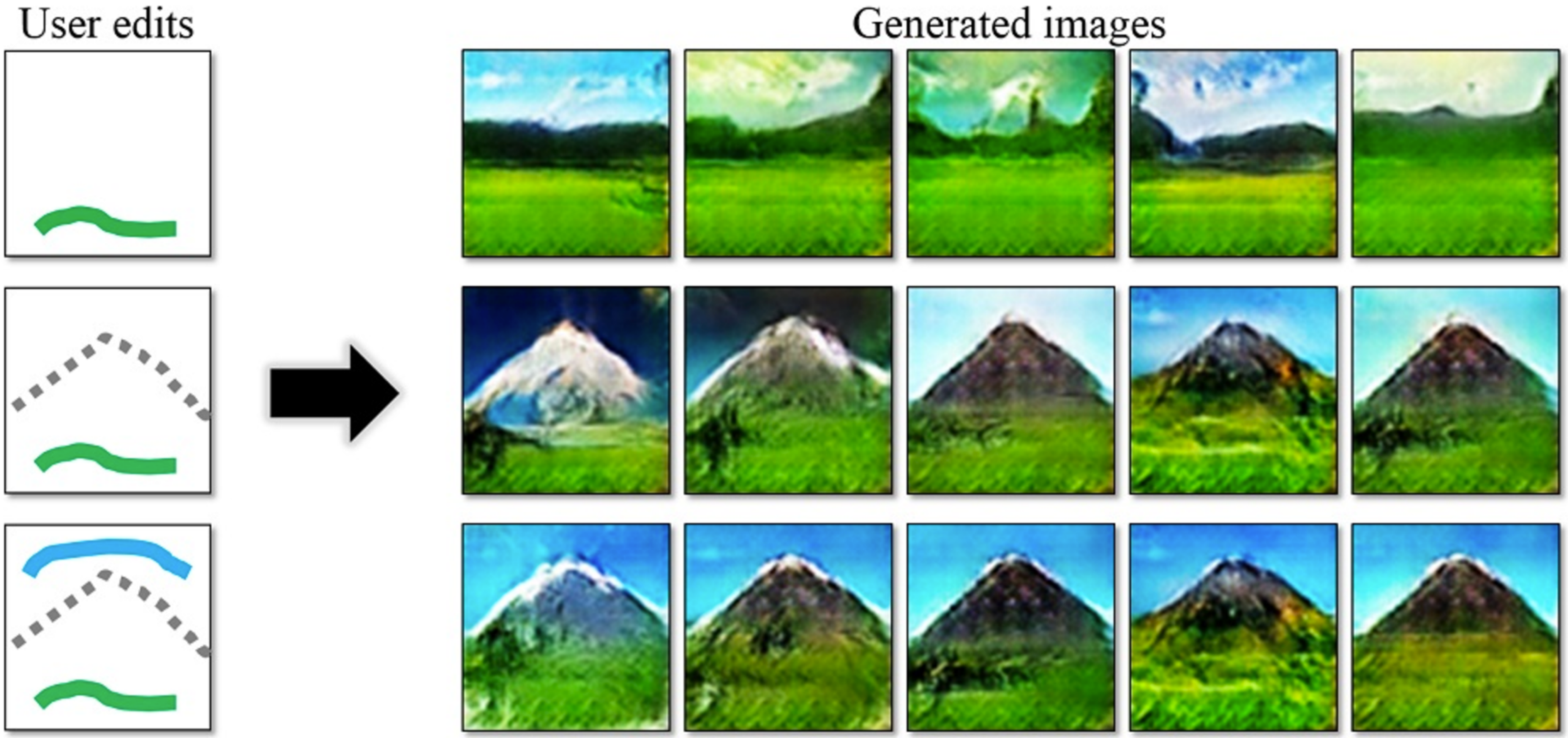
url : www.whichfaceisreal.com (2019)

Magic of GANs

- Which one is computer generated?

url : www.whichfaceisreal.com (2019)

Magic of GANs



http://people.eecs.berkeley.edu/~junyanz/projects/gvm/

Adversarial training

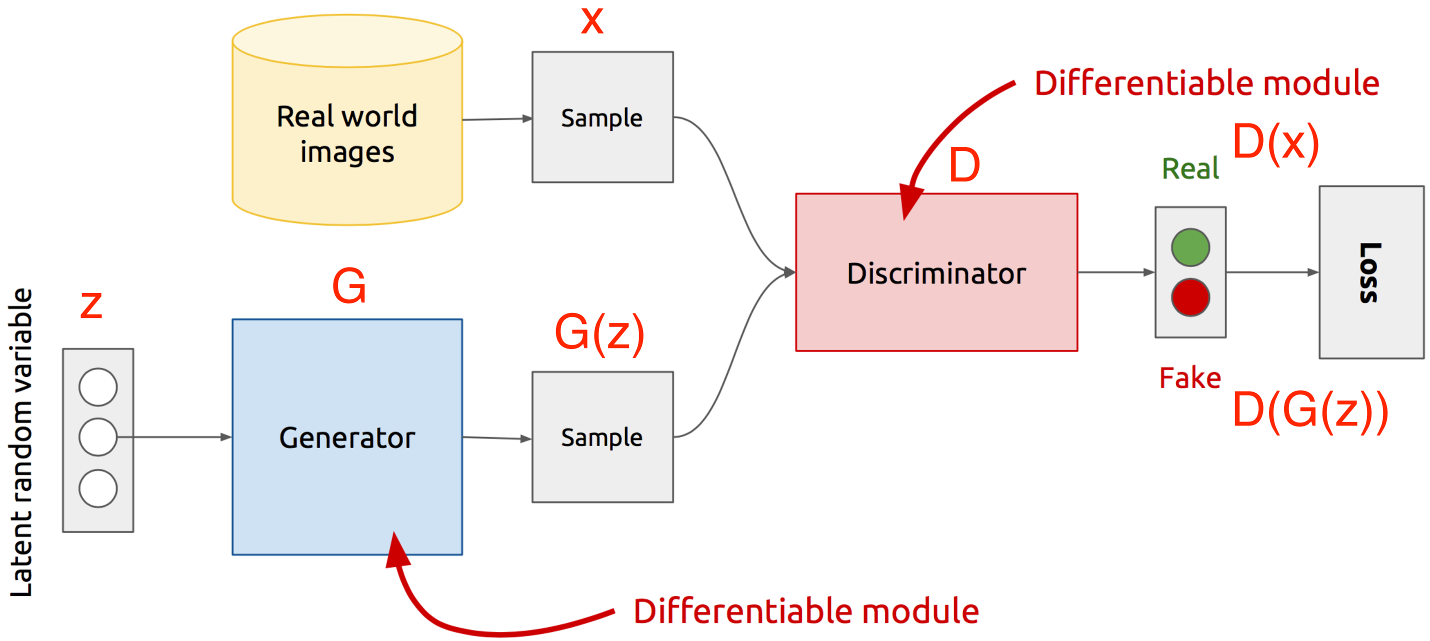
▶ Remarks

- We can generate adversarial samples to fool a discriminative model
- We can use those adversarial samples to make models robust
- We then require more effort to generate adversarial samples
- Repeat this and we get better discriminative model

▶ GANs extend that idea to generative models

- Generator: generate fake samples, tries to fool the Discriminator
- Discriminator: tries to distinguish between real and fake samples
- Train them against each other
- Repeat this and we get better Generator and Discriminator

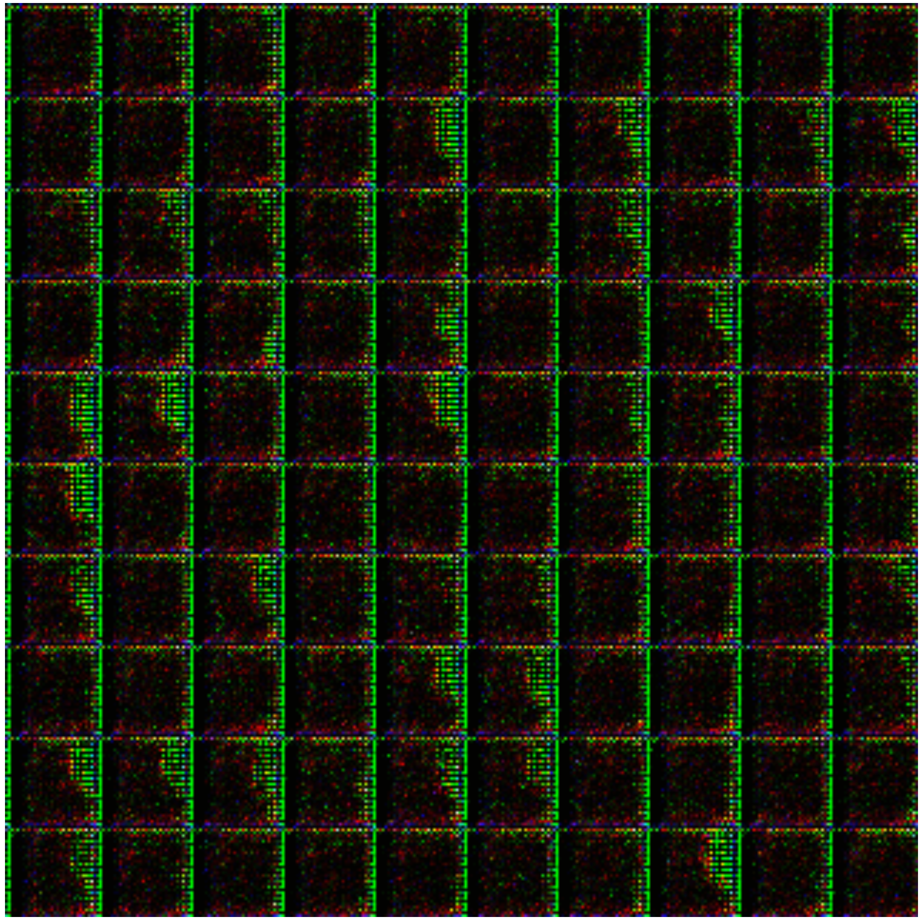
GAN's Architecture



- Z is some random noise (Gaussian/Uniform).
- Z can be thought as the latent representation of the image.

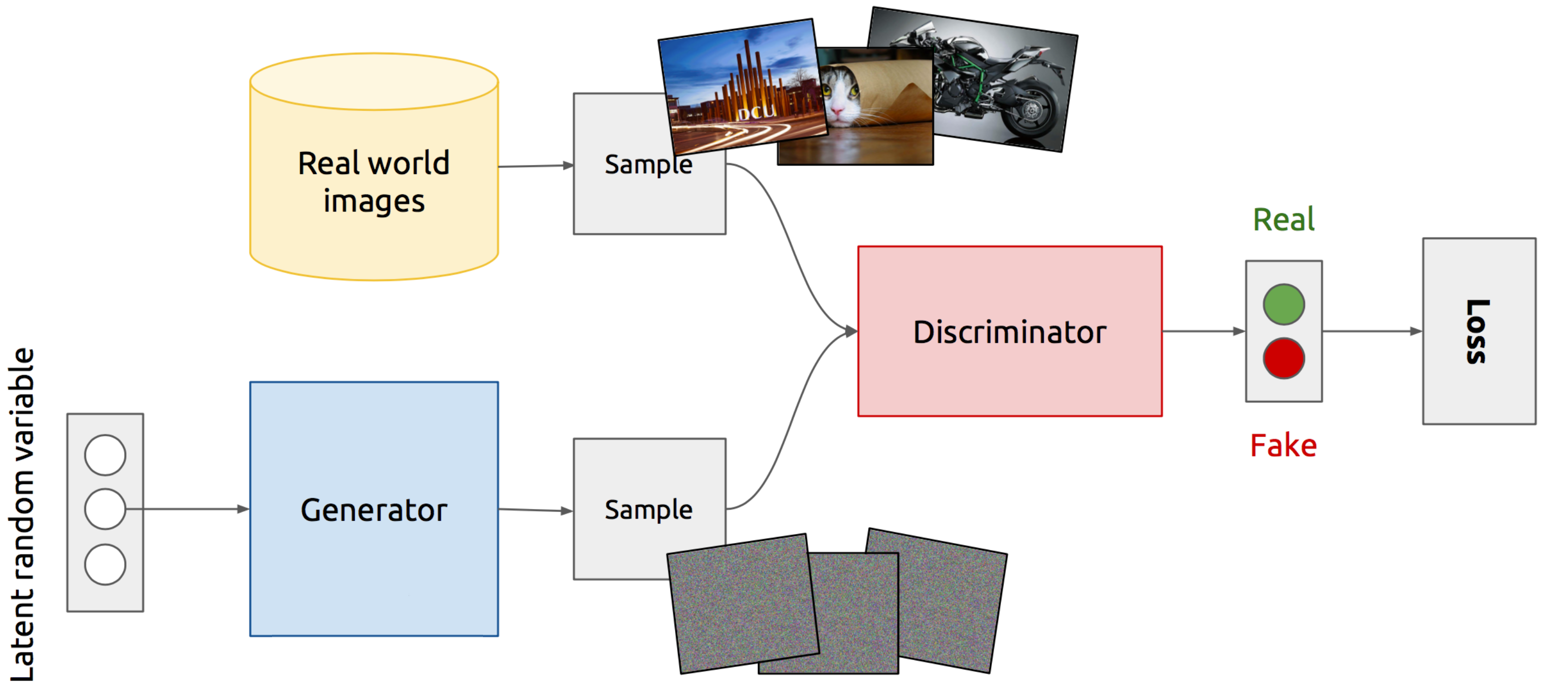
Generator in action

- GAN learning to generate images (linear time)



Training GANs: Two-player game

- **Generator network**: try to fool the discriminator by generating real-looking images
- **Discriminator network**: try to distinguish between real and fake images



Training GANs: Two-player game

- **Generator network**: try to fool the discriminator by generating real-looking images
- **Discriminator network**: try to distinguish between real and fake images

- Train jointly in **minimax game**

- Minimax objective function

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Training GANs: Two-player game

- **Generator network**: try to fool the discriminator by generating real-looking images
- **Discriminator network**: try to distinguish between real and fake images

- Train jointly in **minimax game**

- Minimax objective function

Discriminator outputs likelihood in (0,1) of real image

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \underbrace{\log D_{\theta_d}(x)}_{\text{Discriminator output for real data x}} + \mathbb{E}_{z \sim p(z)} \log(1 - \underbrace{D_{\theta_d}(G_{\theta_g}(z))}_{\text{Discriminator output for generated fake data G(z)}}) \right]$$

- Discriminator (θ_d) wants to **maximize objective** such that D(x) is close to 1 (real) and D(G(z)) is close to 0 (fake)
- Generator (θ_g) wants to **minimize objective** such that D(G(z)) is close to 1 (discriminator is fooled into thinking generated G(z) is real)

Training GANs: Two-player game

- Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

- Alternate between

1. **Gradient ascent** on discriminator

$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

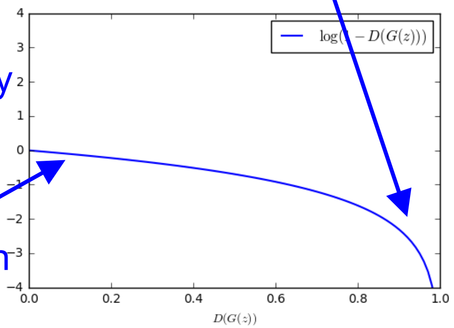
2. **Gradient descent** on generator

$$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

In practice, optimizing this generator objective does not work well!

Gradient signal dominated by region where sample is already good

When sample is likely fake, want to learn from it to improve generator. But gradient in this region is relatively flat!



Training GANs: Two-player game

- Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

- Alternate between

1. **Gradient ascent** on discriminator

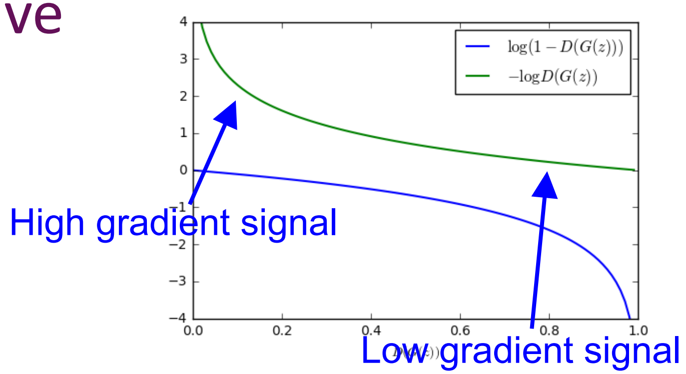
$$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. **Instead**: Gradient **ascent** on generator, different objective

$$\boxed{\max_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(D_{\theta_d}(G_{\theta_g}(z)))}$$

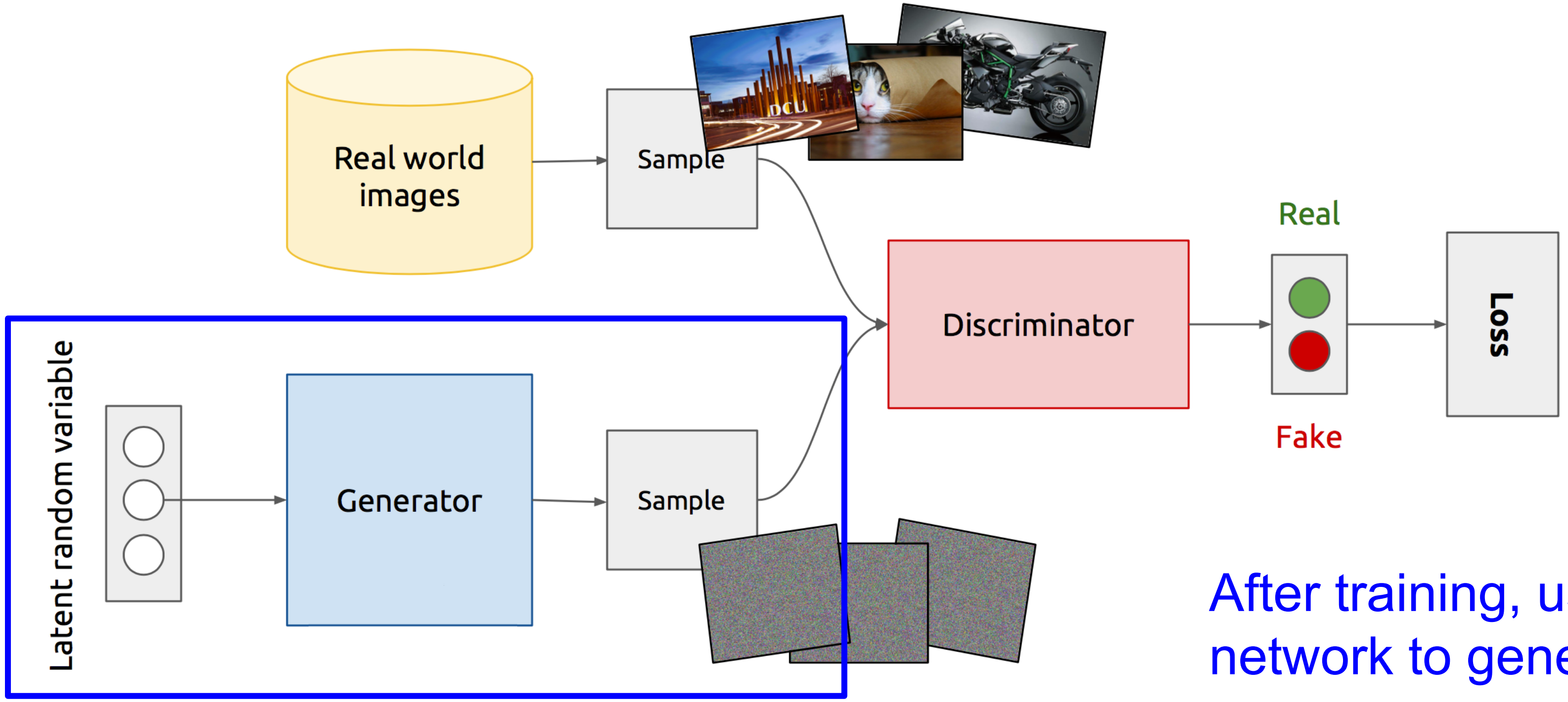
Instead of minimizing likelihood of discriminator being correct, now maximize likelihood of discriminator being wrong.
Same objective of fooling discriminator, but now higher gradient signal for bad samples => works much better! Standard in practice.

Aside: Jointly training two networks is challenging, can be unstable. Choosing objectives with better loss landscapes helps training, is an active area of research.



Training GANs: Two-player game

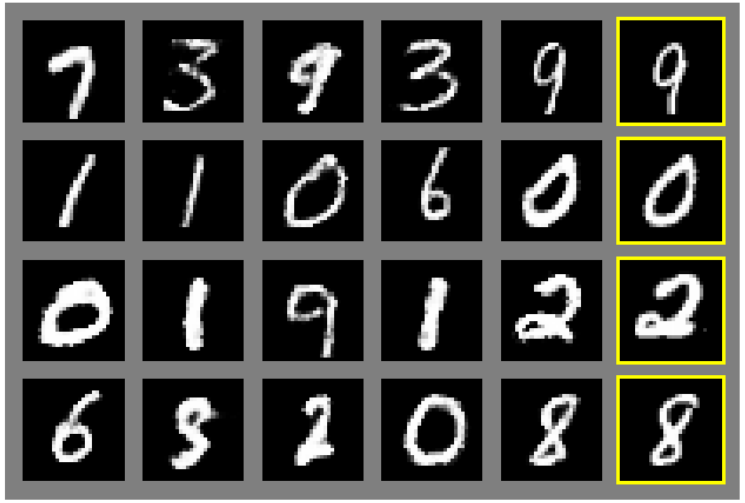
- **Generator network**: try to fool the discriminator by generating real-looking images
- **Discriminator network**: try to distinguish between real and fake images

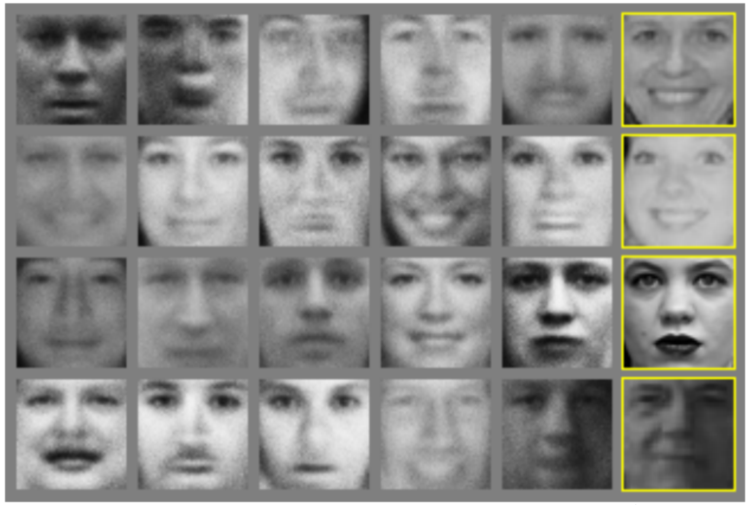


After training, use generator network to generate new images

Generative Adversarial Nets

Generated samples





Nearest neighbor from training set

Generative Adversarial Nets: Convolutional Architectures

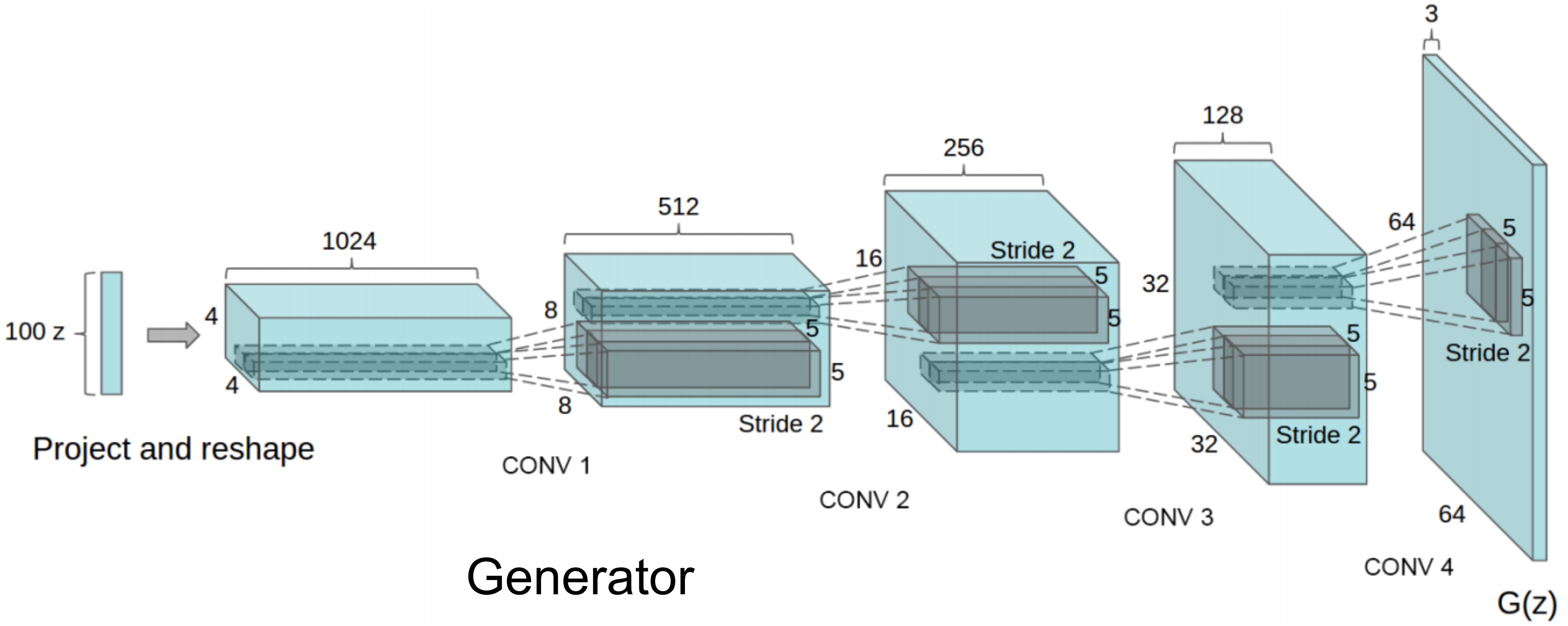
- Generator is an upsampling network with fractionally-strided convolutions
- Discriminator is a convolutional network

Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

Generative Adversarial Nets: Convolutional Architectures



Radford et al, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", ICLR 2016

Generative Adversarial Nets: Convolutional Architectures
Some results

- Trained on LSUN bedroom dataset, 3 millions of images

- Samples from the model look much better!

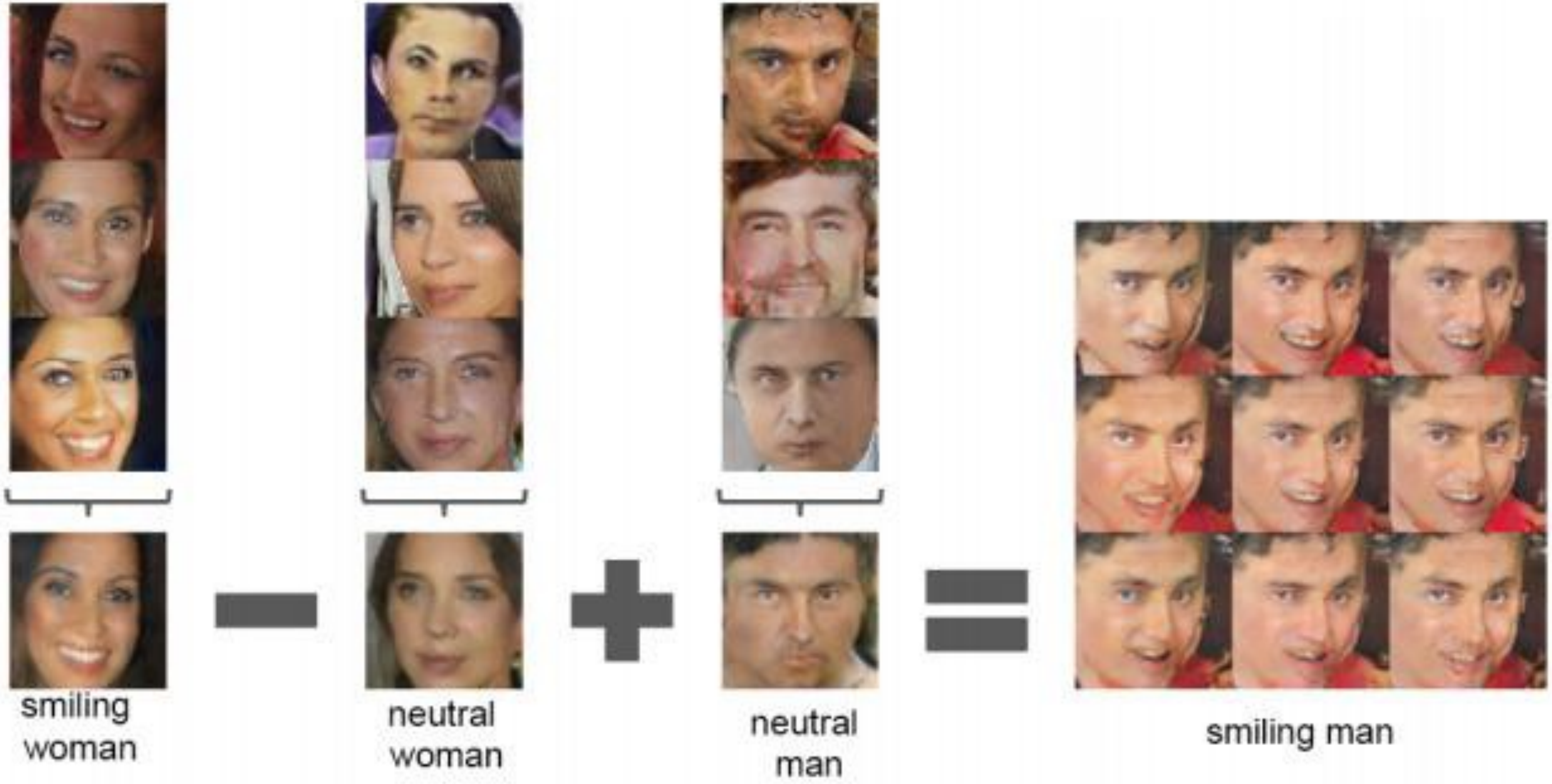
Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

Generative Adversarial Nets: Convolutional Architectures
Some results

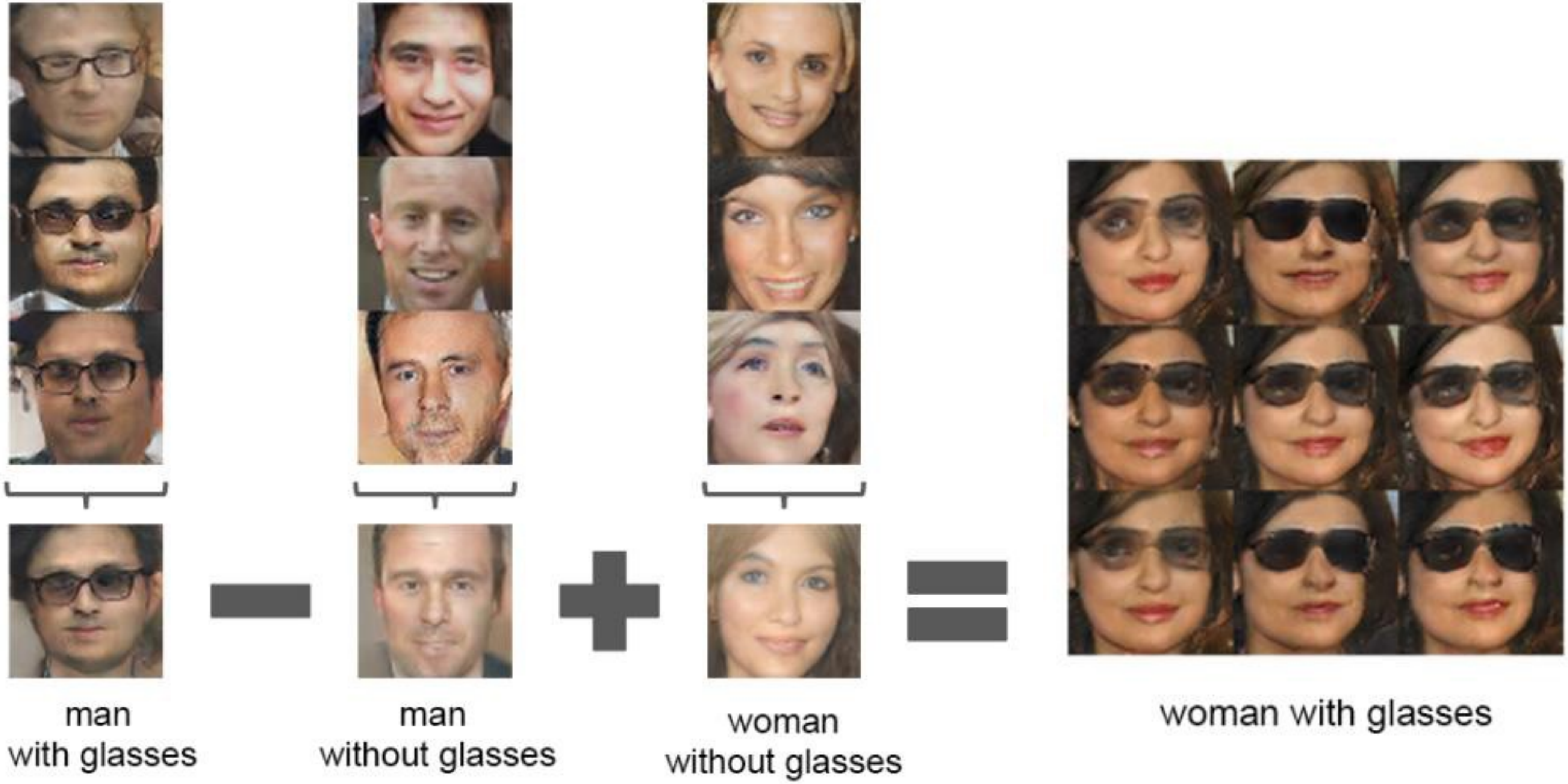
- Allows to evaluate if the network has overlearned

- Interpolating between random points in latent (Z)

Algebra on the latent space Z



Algebra on the latent space Z

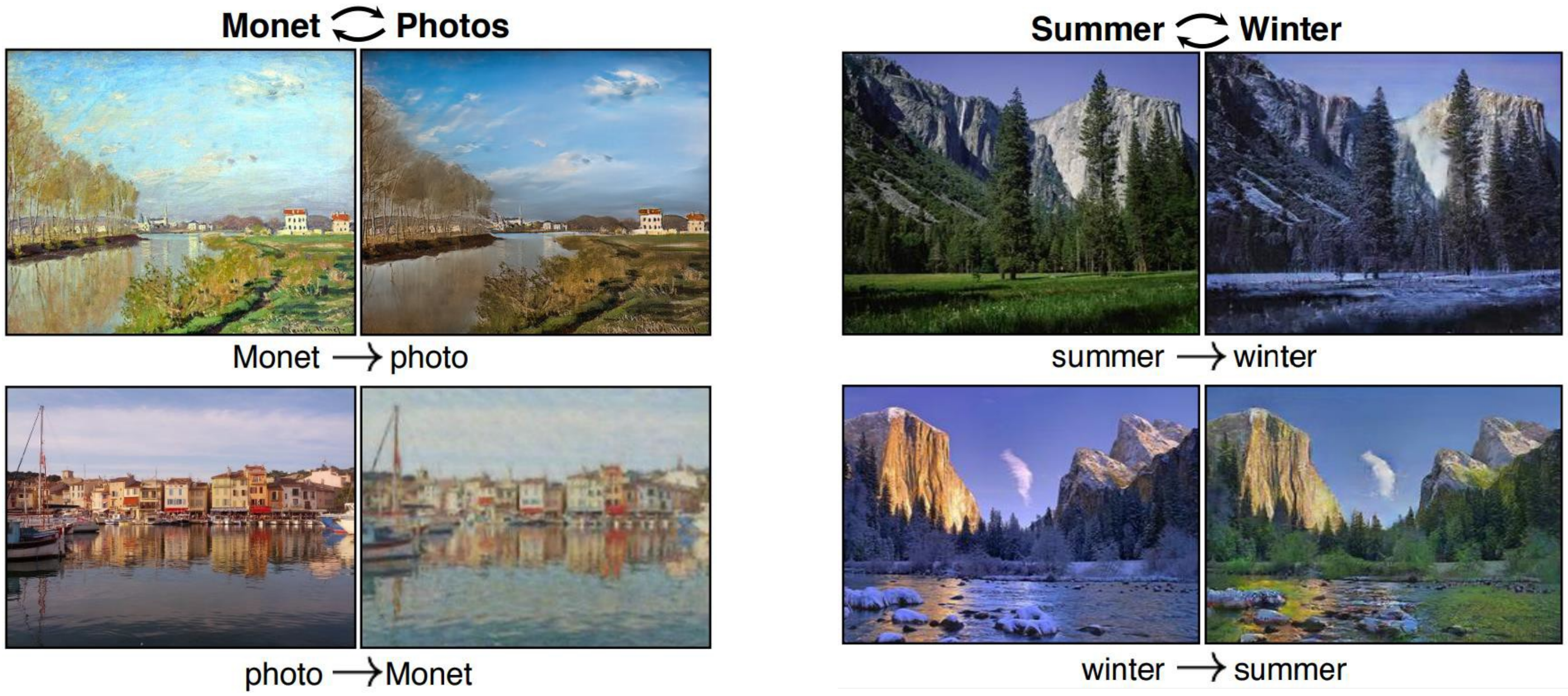


Results over the years

The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation, 2018.

CycleGAN

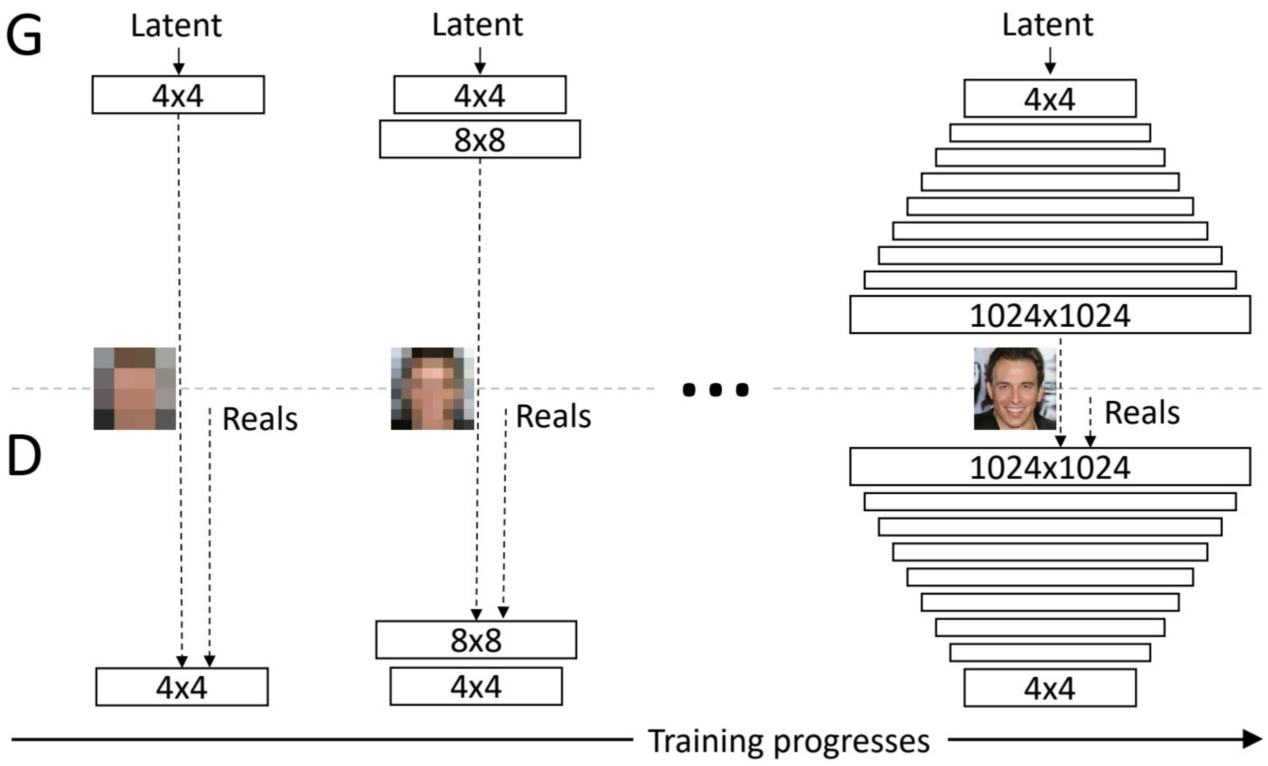
- Allows to define style transfer



Zhu et al., Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, ICCV 2017.

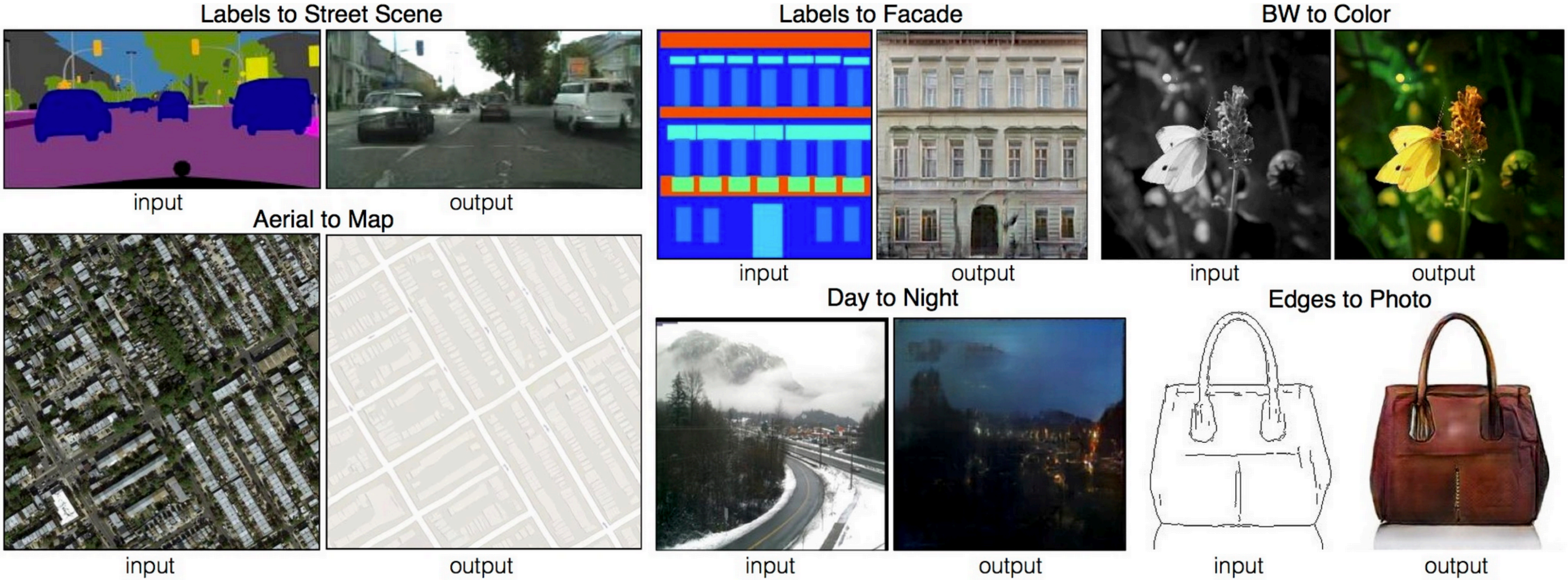
Progressive Growing of GANs

- Allows the network to discover large-scale structures, then refine to detail
- Faster, because it trains mostly on smaller images (2x-6x gain)



Note: fine details are usually problematic for GAN

Pix2Pix examples



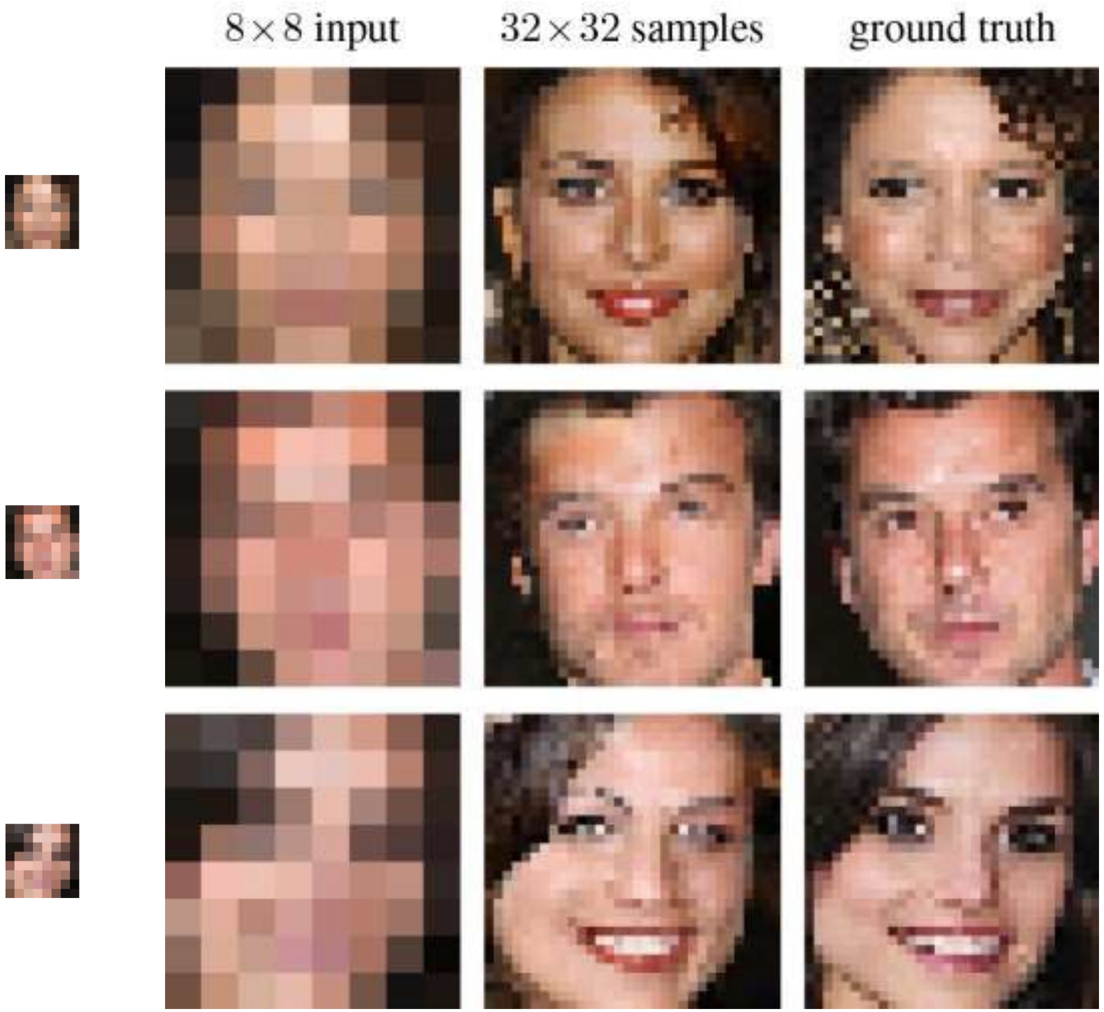
Example results on several image-to-image translation problems. In each case we use the same architecture and objective, simply training on different data.

GANs in action

winter → summer

Photograph → Monet Van Gogh Cezanne Ukiyo-e

Image resolution



Text-to-image Synthesis

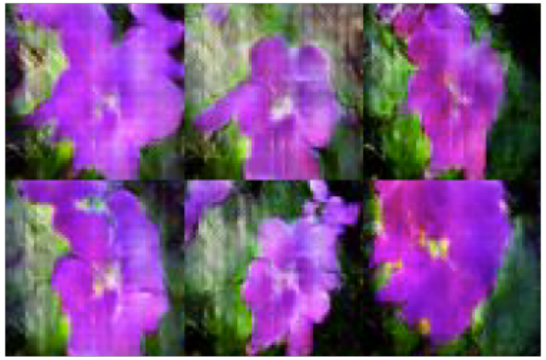
Motivation

- ▶Given a text description, generate images closely associated.
- ▶Uses a conditional GAN with the generator and discriminator being condition on “dense” text embedding.

this small bird has a pink breast and crown, and black primaries and secondaries.

this magnificent fellow is almost all black with a red crest, and white cheek patch.

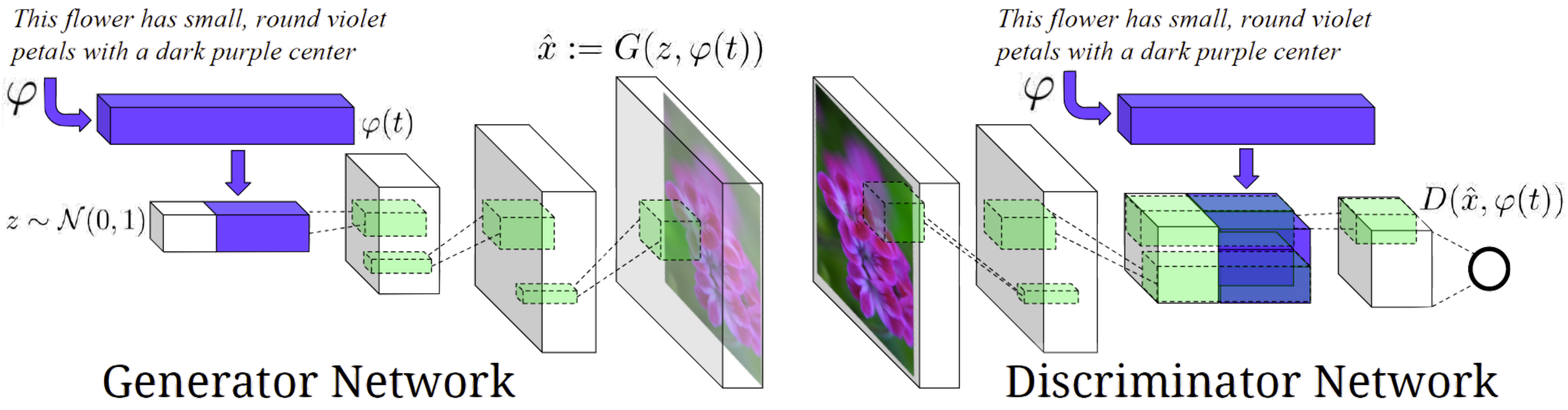
the flower has petals that are bright pinkish purple with white stigma



this white and yellow flower have thin white petals and a round yellow stamen

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., & Lee, H. “Generative adversarial text to image synthesis”. ICML (2016).

Text-to-image Synthesis



Positive Examples

Real Image, Right Text

Negative Examples

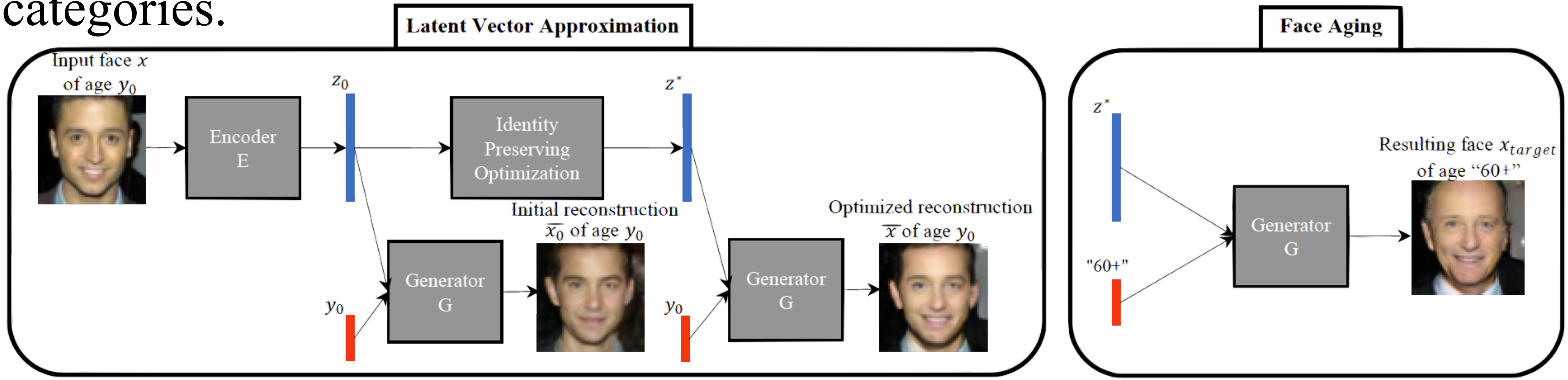
Real Image, wrong Text

Fake Image, right text

Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., & Lee, H. "Generative adversarial text to image synthesis". ICML (2016).

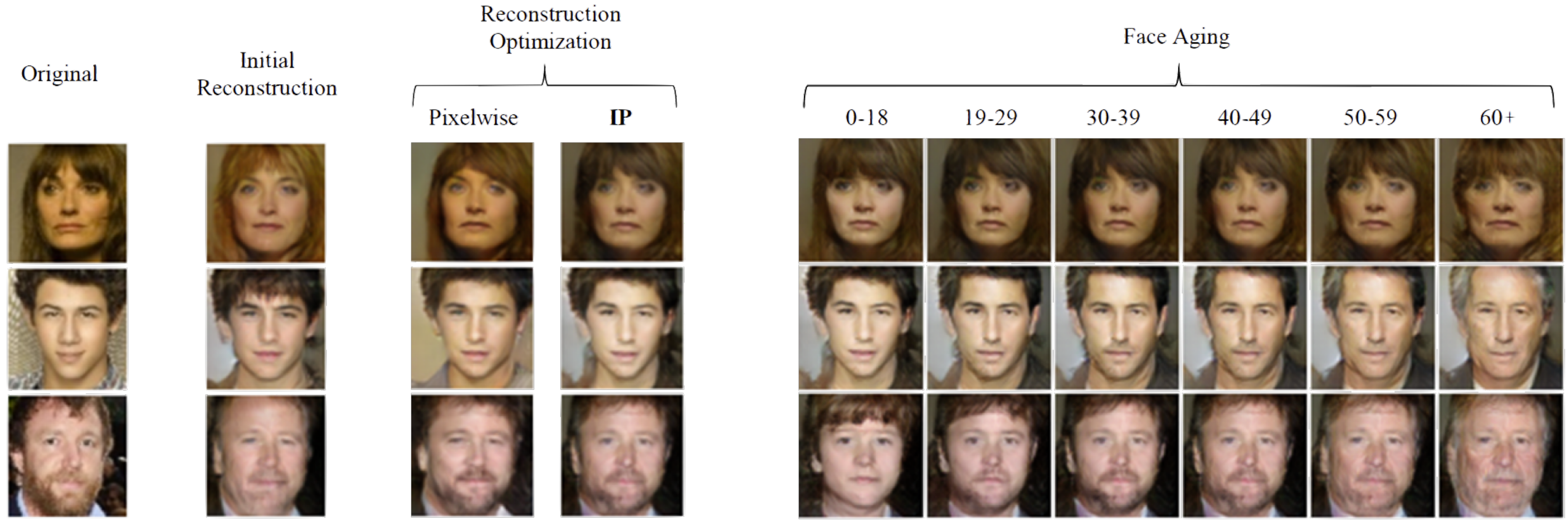
Face Aging with conditional GANs

- Differentiating Feature: Uses an Identity Preservation Optimization using an auxiliary network to get a better approximation of the latent code (z*) for an input image.
- Latent code is then conditioned on a discrete (one-hot) embedding of age categories.



Antipov, G., Baccouche, M., & Dugelay, J. L. (2017). "Face Aging With Conditional Generative Adversarial Networks". arXiv preprint arXiv:1702.01983.

Face Aging with conditional GANs



Antipov, G., Baccouche, M., & Dugelay, J. L. (2017). “Face Aging With Conditional Generative Adversarial Networks”. arXiv preprint arXiv:1702.01983.

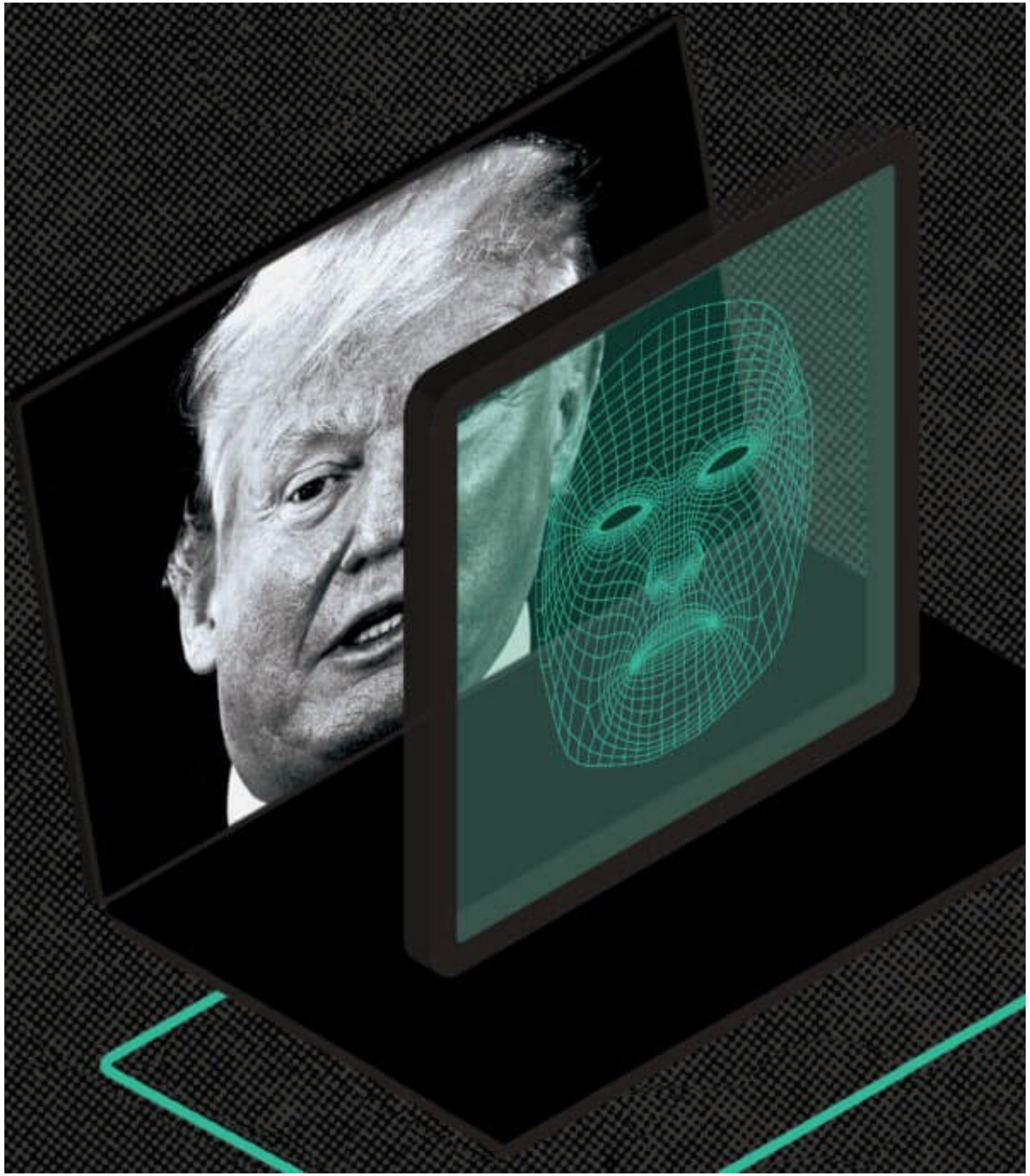
The GAN Zoo

- GAN - Generative Adversarial Networks
- 3D-GAN - Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling
- acGAN - Face Aging With Conditional Generative Adversarial Networks
- AC-GAN - Conditional Image Synthesis With Auxiliary Classifier GANs
- AdaGAN - AdaGAN: Boosting Generative Models
- AEGAN - Learning Inverse Mapping by Autoencoder based Generative Adversarial Nets
- AffGAN - Amortised MAP Inference for Image Super-resolution
- AL-CGAN - Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts
- ALI - Adversarially Learned Inference
- AM-GAN - Generative Adversarial Nets with Labeled Data by Activation Maximization
- AnoGAN - Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery
- ArtGAN - ArtGAN: Artwork Synthesis with Conditional Categorial GANs
- b-GAN - b-GAN: Unified Framework of Generative Adversarial Networks
- Bayesian GAN - Deep and Hierarchical Implicit Models
- BEGAN - BEGAN: Boundary Equilibrium Generative Adversarial Networks
- BiGAN - Adversarial Feature Learning
- BS-GAN - Boundary-Seeking Generative Adversarial Networks
- CGAN - Conditional Generative Adversarial Nets
- CaloGAN - CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks
- CCGAN - Semi-Supervised Learning with Context-Conditional Generative Adversarial Networks
- CatGAN - Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks
- CoGAN - Coupled Generative Adversarial Networks

- Context-RNN-GAN - Contextual RNN-GANs for Abstract Reasoning Diagram Generation
- C-RNN-GAN - C-RNN-GAN: Continuous recurrent neural networks with adversarial training
- CS-GAN - Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets
- CVAE-GAN - CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training
- CycleGAN - Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks
- DTN - Unsupervised Cross-Domain Image Generation
- DCGAN - Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks
- DiscoGAN - Learning to Discover Cross-Domain Relations with Generative Adversarial Networks
- DR-GAN - Disentangled Representation Learning GAN for Pose-Invariant Face Recognition
- DualGAN - DualGAN: Unsupervised Dual Learning for Image-to-Image Translation
- EBGAN - Energy-based Generative Adversarial Network
- f-GAN - f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization
- FF-GAN - Towards Large-Pose Face Frontalization in the Wild
- GAWWN - Learning What and Where to Draw
- GeneGAN - GeneGAN: Learning Object Transfiguration and Attribute Subspace from Unpaired Data
- Geometric GAN - Geometric GAN
- GoGAN - Gang of GANs: Generative Adversarial Networks with Maximum Margin Ranking
- GP-GAN - GP-GAN: Towards Realistic High-Resolution Image Blending
- IAN - Neural Photo Editing with Introspective Adversarial Networks
- iGAN - Generative Visual Manipulation on the Natural Image Manifold
- IcGAN - Invertible Conditional GANs for image editing
- ID-CGAN - Image De-raining Using a Conditional Generative Adversarial Network
- Improved GAN - Improved Techniques for Training GANs
- InfoGAN - InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets
- LAGAN - Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis
- LAPGAN - Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks

https://github.com/hindupuravinash/the-gan-zoo

Also see https://paperswithcode.com/task/image-generation/latest



Normandie Université

GREYC
Electronics and Computer Science Laboratory

DEEPFAKE MISUSE

Deepfake misuse

Cyber crime

Privacy invasion

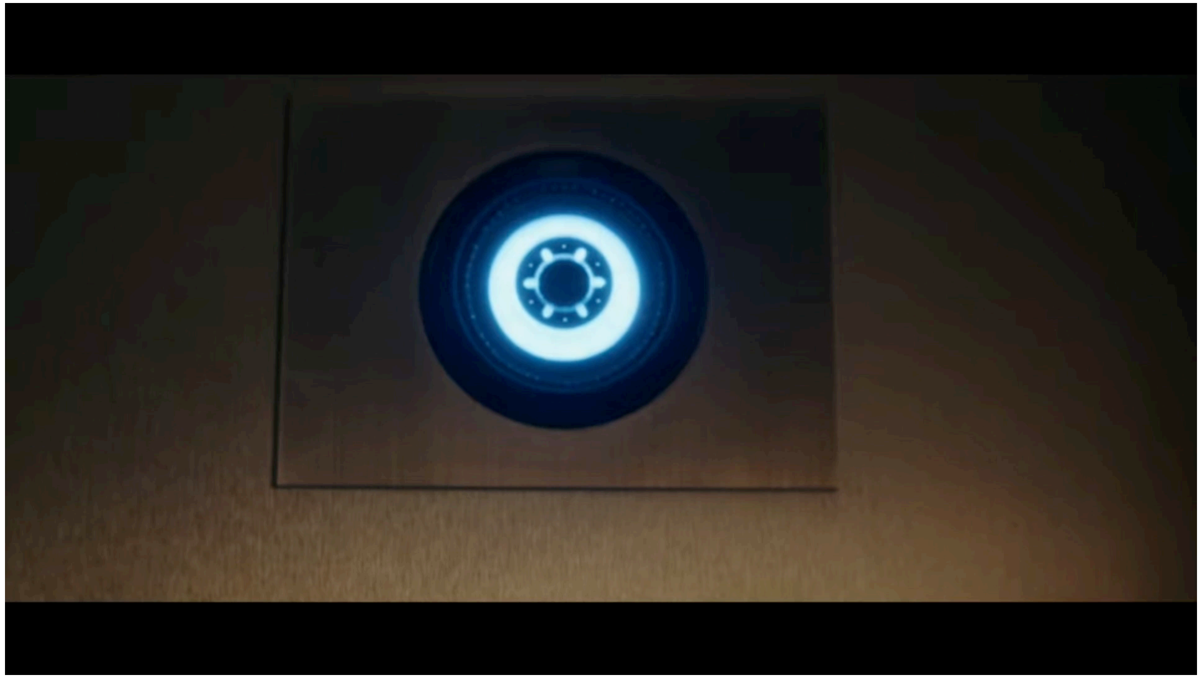
Social manipulation

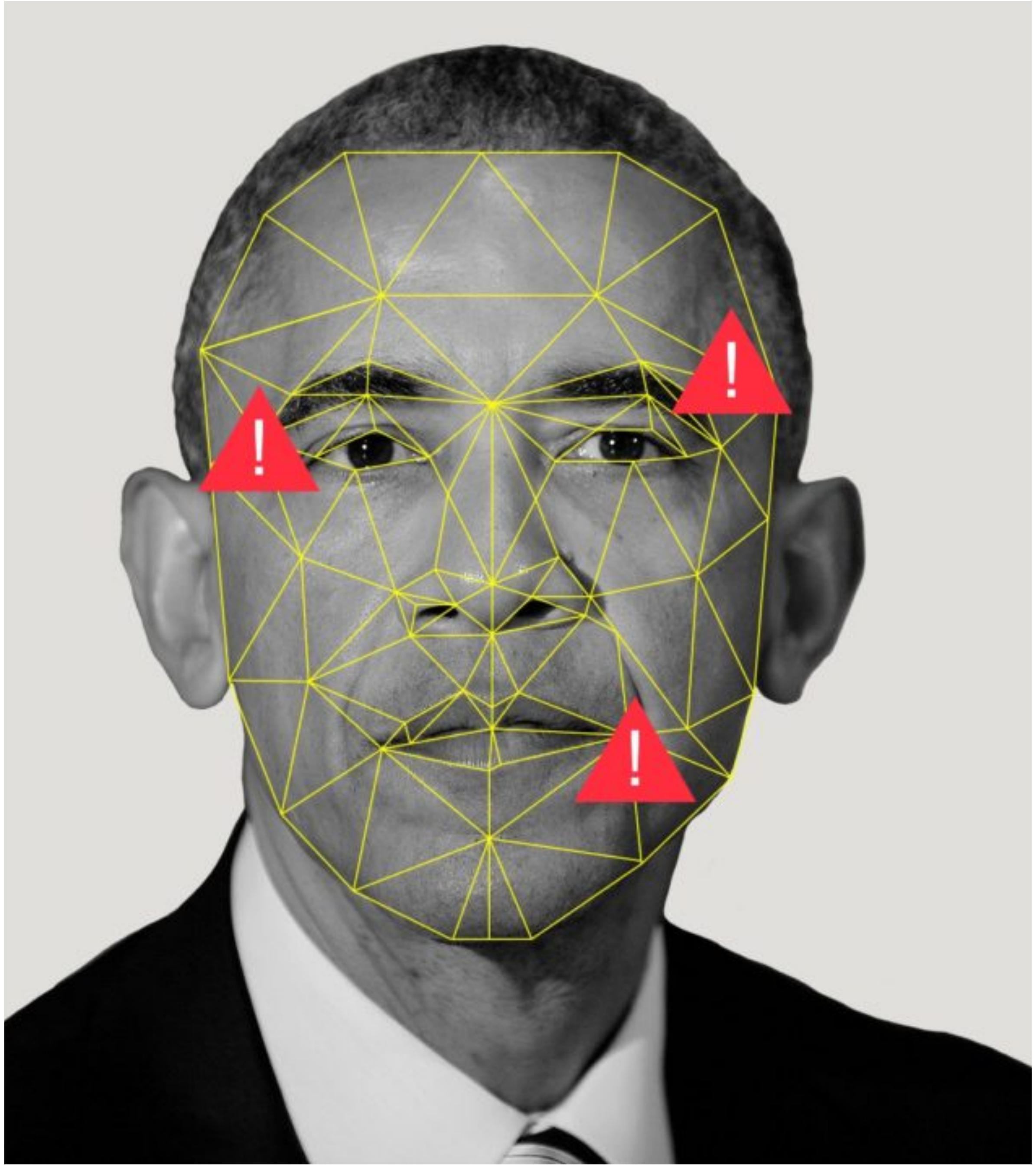
Deepfake misuse

- Attacks

In 2017….

In 2021





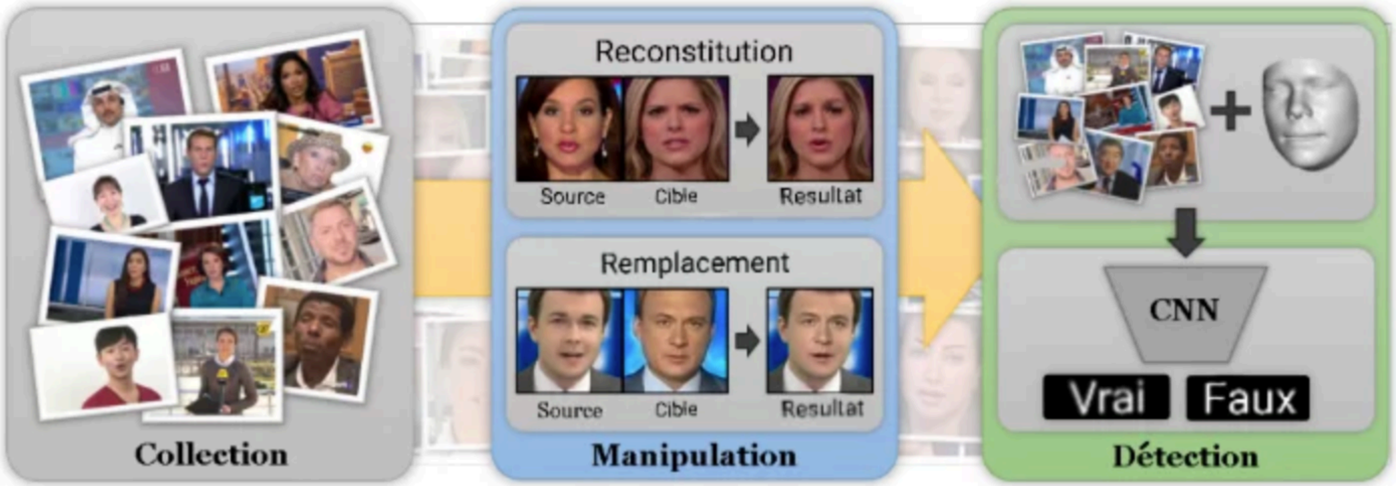
DEEPFAKE DETECTION

Deepfake detection challenge

- In 2021, Facebook has launched a project, with Michigan State University, to create the "most successful deepfake detection software available today". This technique is called reverse engineering. It consists of deconstructing the photo or video to identify imperfections added to the editing.

FaceForensic++

- FaceForensics++ is a database that allows researchers to be aware of the latest advances in deepfake detection software. This dataset is based on different deepfake methods that analyze videos and try to find a clue of faking through a CNN



Video Athenticator

- In Sept. 2020, Microsoft unveiled a software that analyzes a still image or video.
- It will provide a percentage of chance the media is artificially manipulated.
- In the case of a video, it can provide this percentage in real time on all the images while the video is playing.
- It uses the FaceForensics++ database and has been tested during the DeepFake Detection Challenge with a high success rate (around 90%)

Detection challenges

- If deepfake detection software rise up, they tackle at least three major difficulties:

  1. The quality of the video to analyze

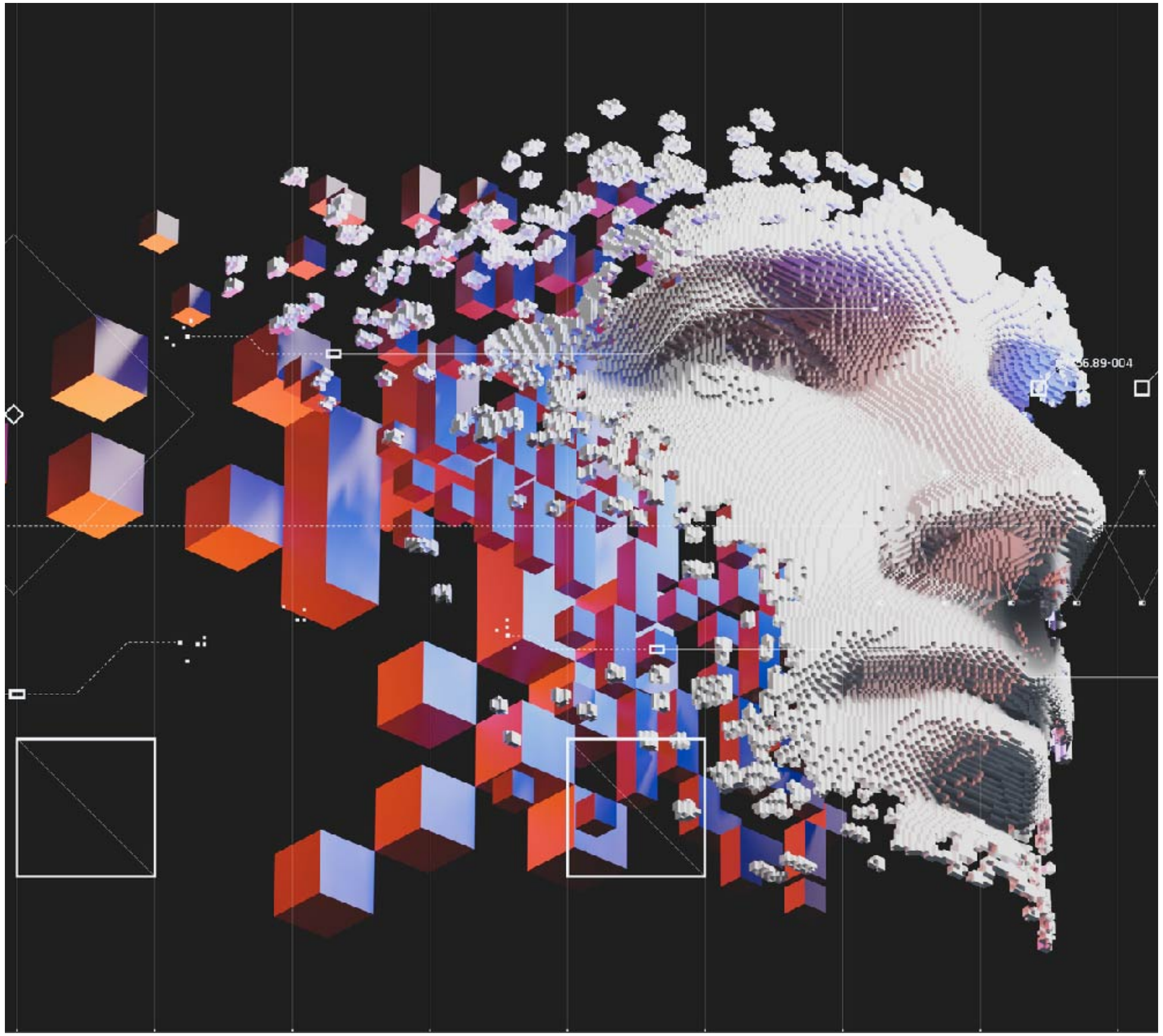
     Video of low quality decreases the performance of detector

  2. The multiplicity of deepfake methods

  3. The ever-increasing evolution of technology

Can we find deepfake in movies?

- The movie industry is starting to be interested in deepfake technology for its movies, both in Hollywood and in France.
- In the french TV soap "Plus belle la vie", an episode uses deepfake to compensate for the absence of an actress.
- A specialist of this technology has been hired by Lucas Film for the next season of "The Mandalorian”….



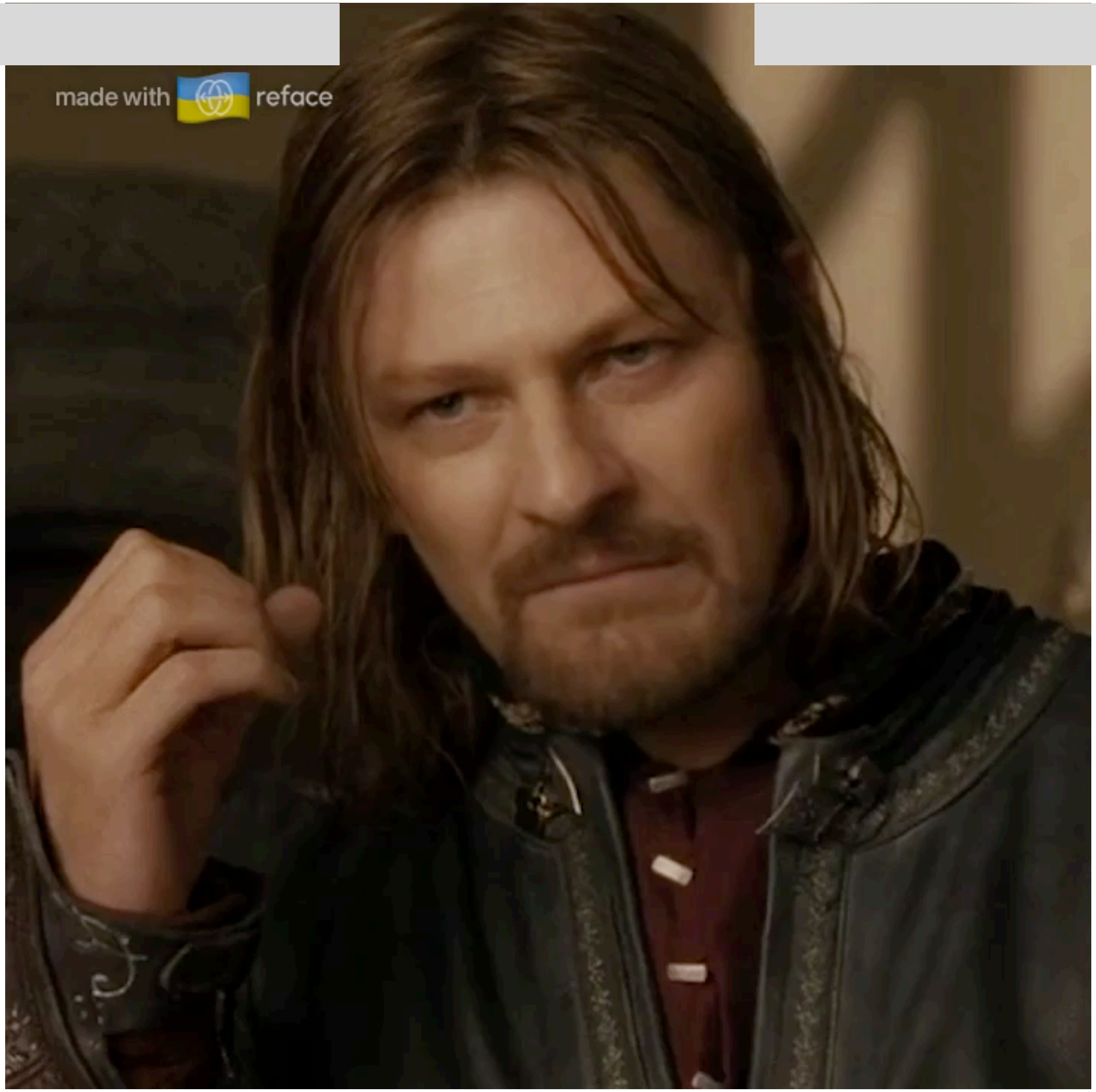
FUTURE TRENDS

And now?

- Pandora's box has opened, and it looks like the competition between the creation and detection and prevention of deepfakes will become increasingly fierce in the future, with deepfake technology not only becoming easier to access, but deepfake content easier to create and progressively harder to distinguish from real.
- According to experts, GANs (generative adversarial networks) will be the main drivers of deepfakes development in the future, and these will be near-impossible to distinguish from authentic content.

And now?

- Deepfakes are not only a technical problem, and as the Pandora's box has been opened, they are not going to disappear in the foreseeable future.
- But with technical improvements in our ability to detect them, and the increased public awareness of the problem, we can learn to co-exist with them and to limit their negative impacts in the future.


made with
reface

Contacts

https://www.greyc.fr/?page_id=455

christophe.charrier@unicaen.fr

ENSICAEN
GREYC
6 Boulevard du Maréchal Juin
Bâtiment F
CS 45053
14050 CAEN cedex 4
TEL : +33 (0)2 31 45 25 04