Autoencoders and Generative Adversarial Networks

Giacomo Boracchi

Advanced Neural Networks and Deep Learning

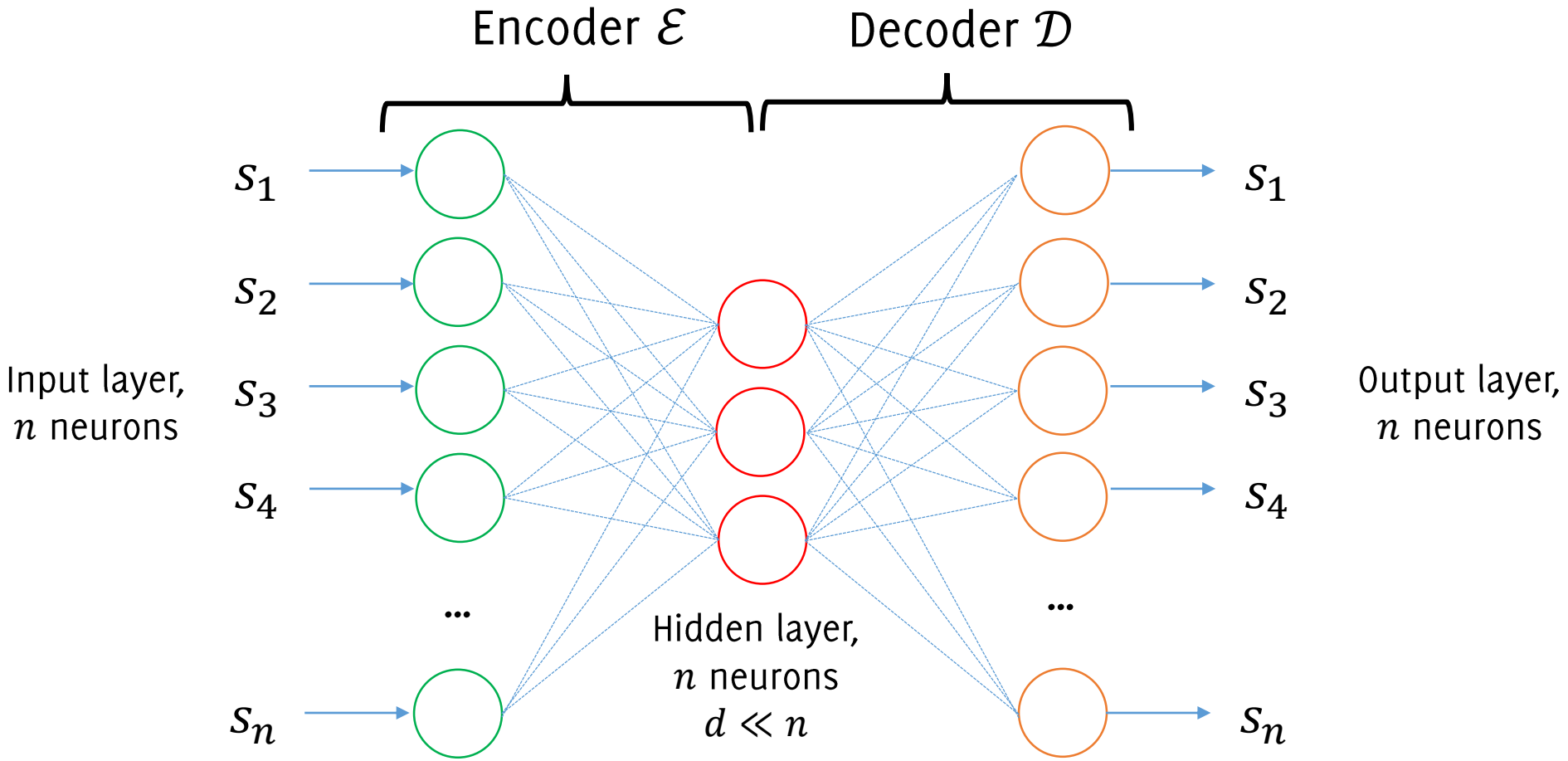
https://boracchi.faculty.polimi.it/

Autoencoders

Autoencoders using MLP

Autoencoders are neural networks used for data reconstruction (unsupervised learning)

The typical structure of an autoencoder is:



Autoencoders using MLP

Autoencoders can be trained to reconstruct all the data in a training set.

The reconstruction loss over a batch S is

$$\ell(S) = \sum_{\boldsymbol{s} \in S} \left\| \boldsymbol{s} - \mathcal{D}\big(\mathcal{E}(\boldsymbol{s})\big) \right\|_2$$

and training of $\mathcal{D}\big(\mathcal{E}(\cdot)\big)$ is performed through standard backpropagation algorithms (e.g. SGD).

The autoencoder thus learns the identity mapping.

Rmk there are **no external labels** involved in training the autoencoder, as it performs reconstruction of the input

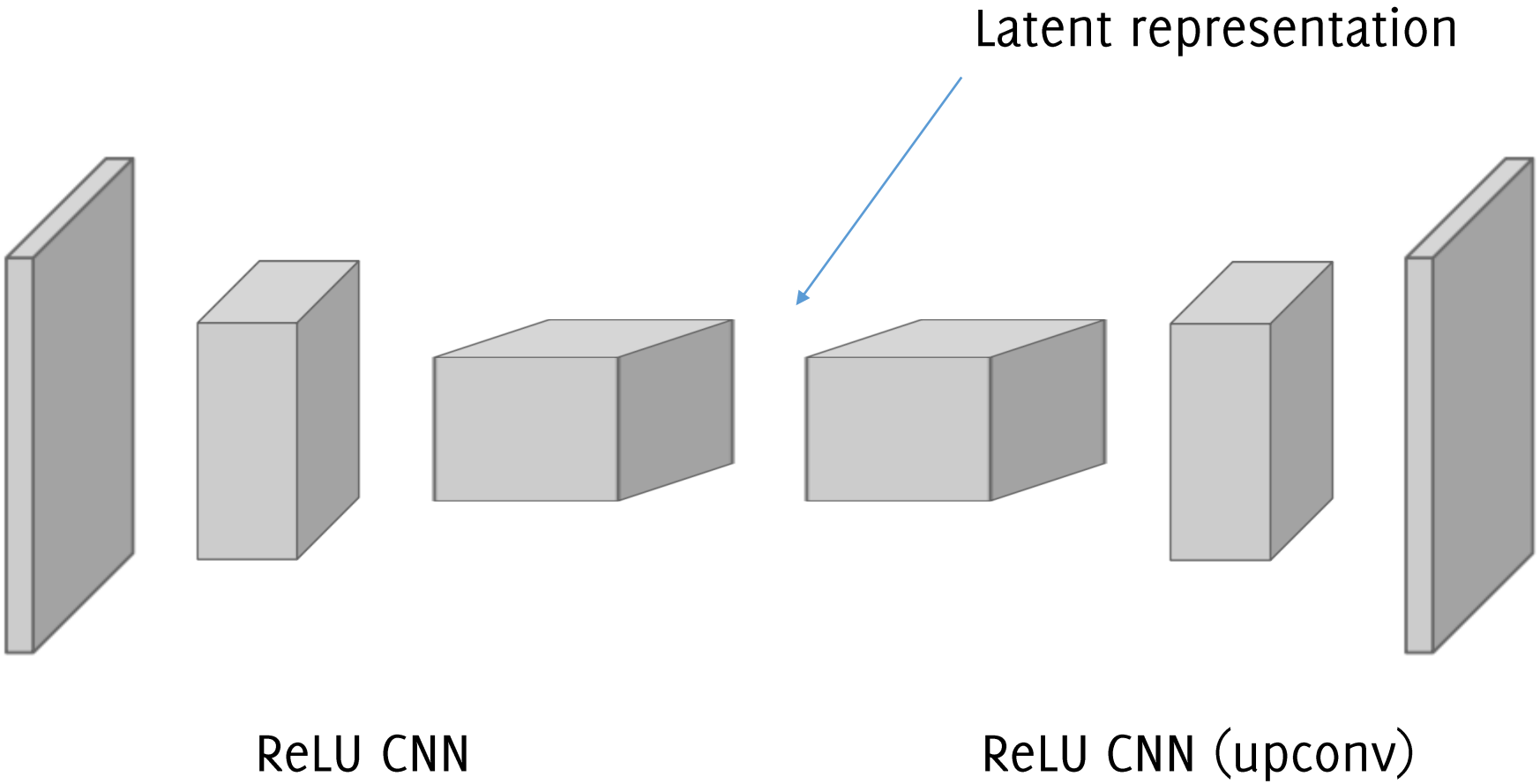
Autoencoders

Remark:

- Features $z = \mathcal{E}(s)$ are typically referred to as **latent representation**
- AE typically do not provide exact reconstruction since $n \ll d$, by doing so we **expect the latent representation to be a meaningful and compact representation** of the input
- **It is possible to add a regularization term** $+\lambda\mathcal{R}(s)$ to steer latent representation $\mathcal{E}(s)$ to satisfy desired properties (e.g. sparsity) or the reconstruction $\mathcal{D}(\mathcal{E}(s))$ (e.g. smoothness, sharp edges in case of images)
- More powerful and nonlinear representations can be learned by **stacking multiple hidden layers** (deep autoencoders)

Convolutional AutoEncoders

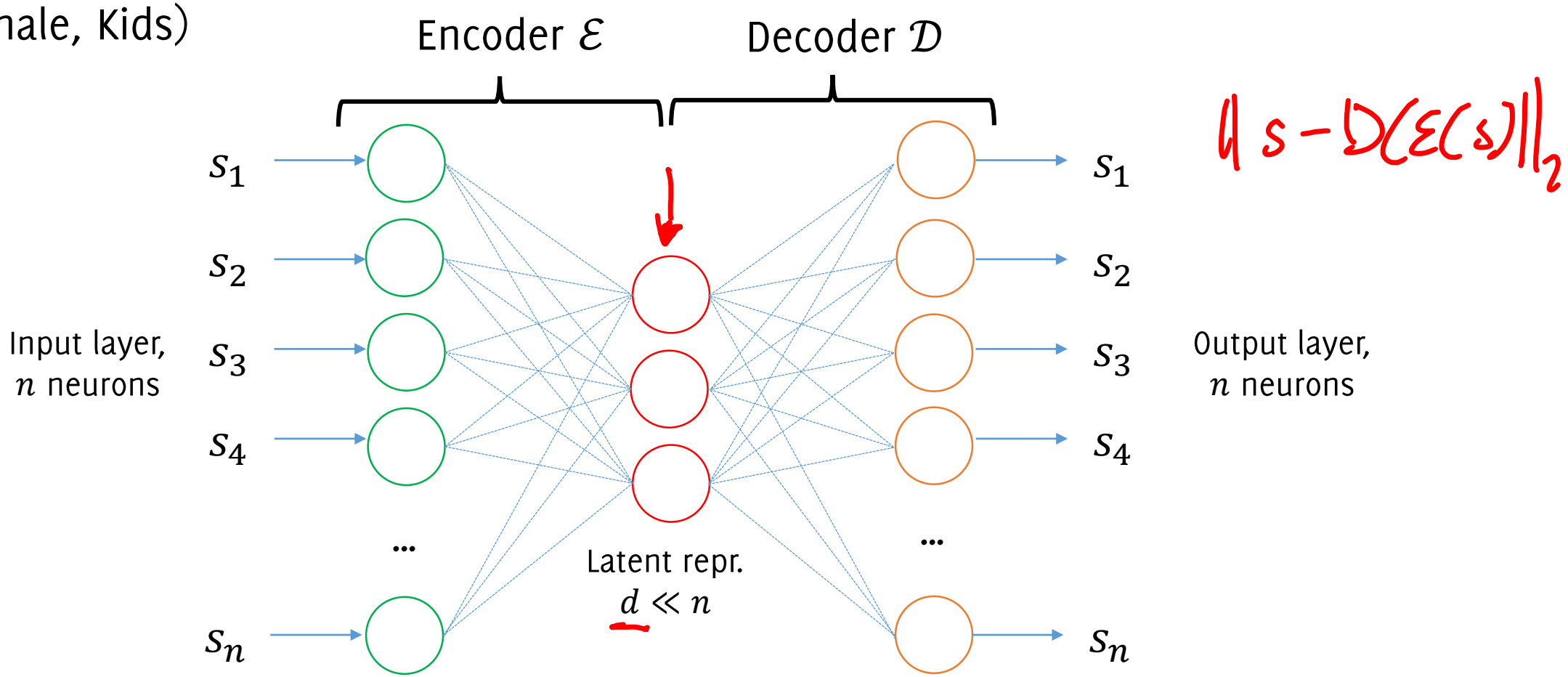
And of course it is possible to use convolutional layers and transpose convolution to implement a deep convolutional autoencoder



Using Autoencoders for classifier initialization

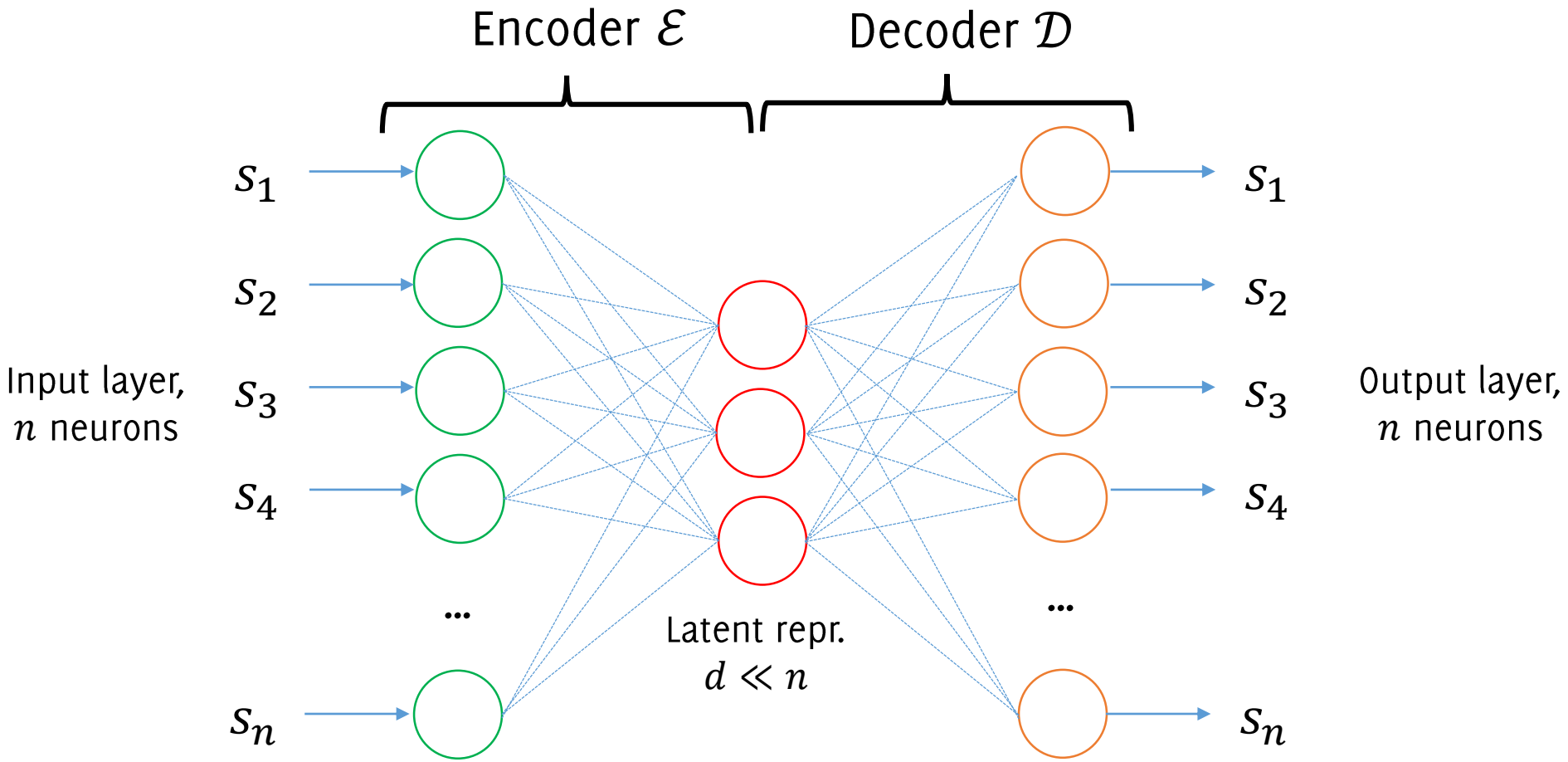
Autoencoders can be used to **inizialize the classifier when** the training set includes

- **few annotated data** (e.g. a large set $S = \{s_i\}$ of unlabeled human faces)
- **many unlabeled ones** (e.g. a small set $L = \{(s_i, y_i)\}$ of faces labelled as Male, Female, Kids)



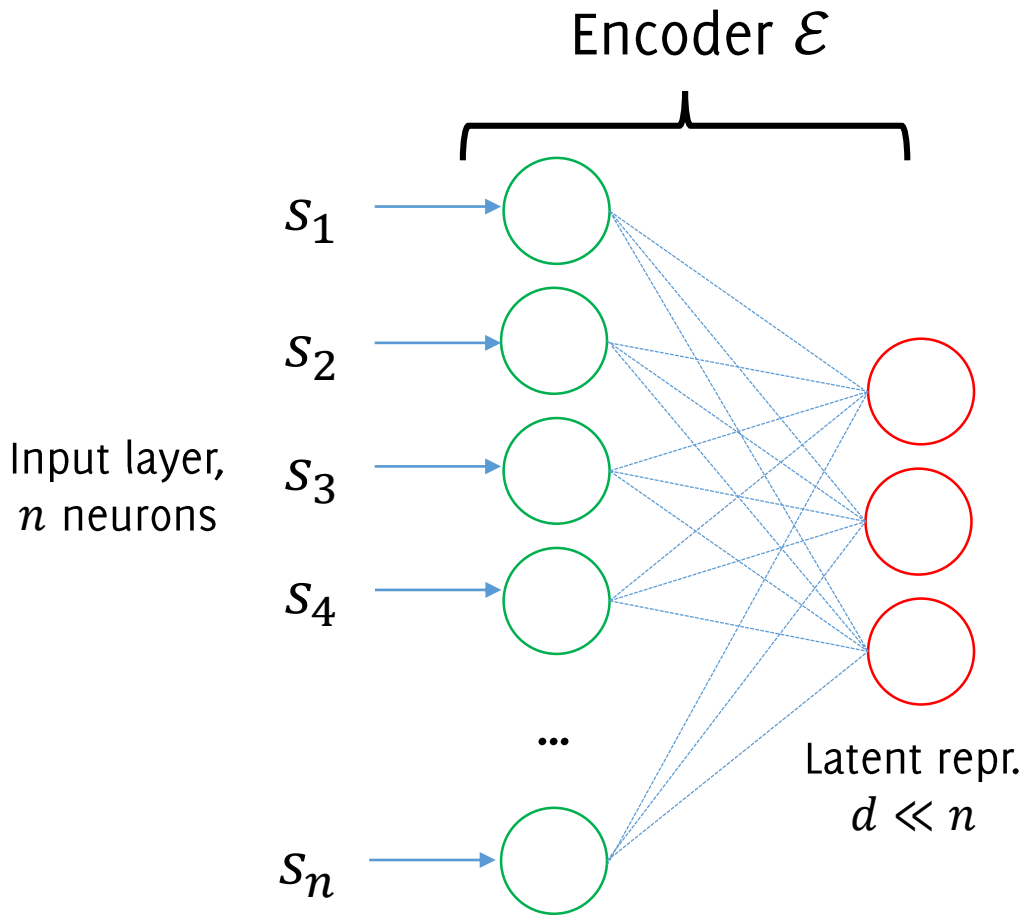
Using Autoencoders for classifier initialization

1) **Train the autoencoder** in a fully unsupervised way, using the unlabeled data S



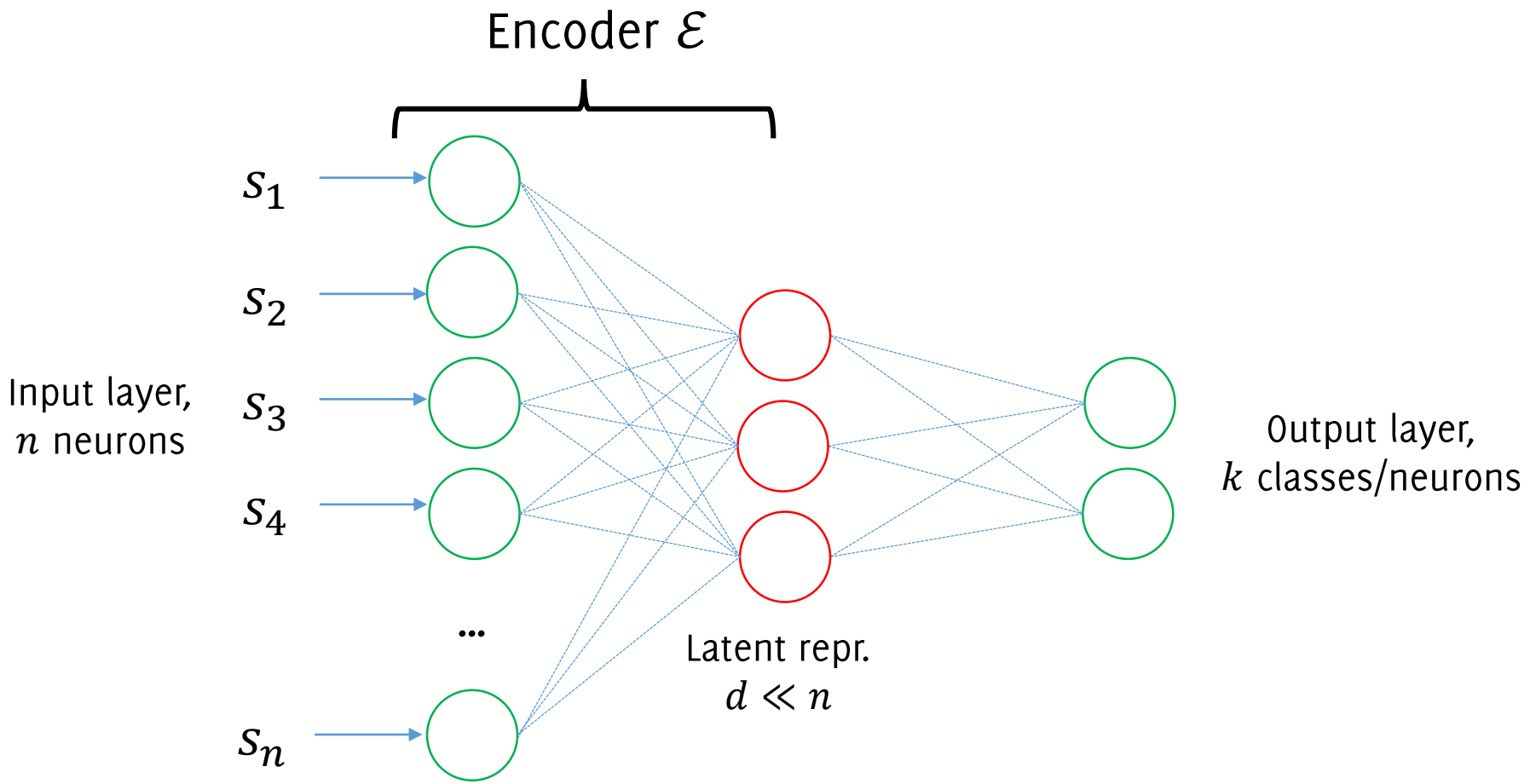
Using Autoencoders for classifier initialization

2) **Get rid of the decoder** and keep the encoder weights



Using Autoencoders for classifier initialization

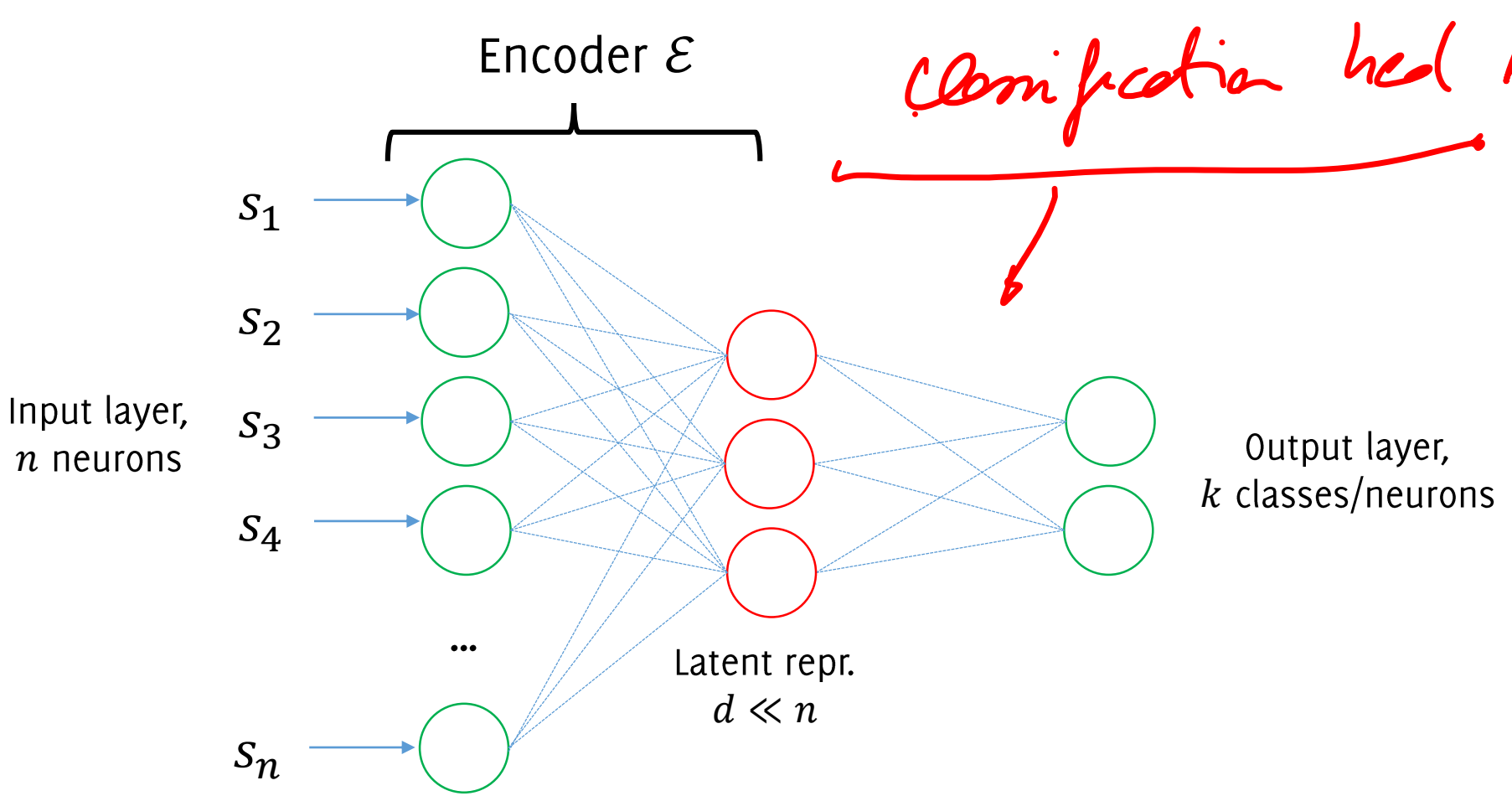
3) **Plug in a FC layer for classifying** samples from the latent representation



Using Autoencoders for classifier initialization

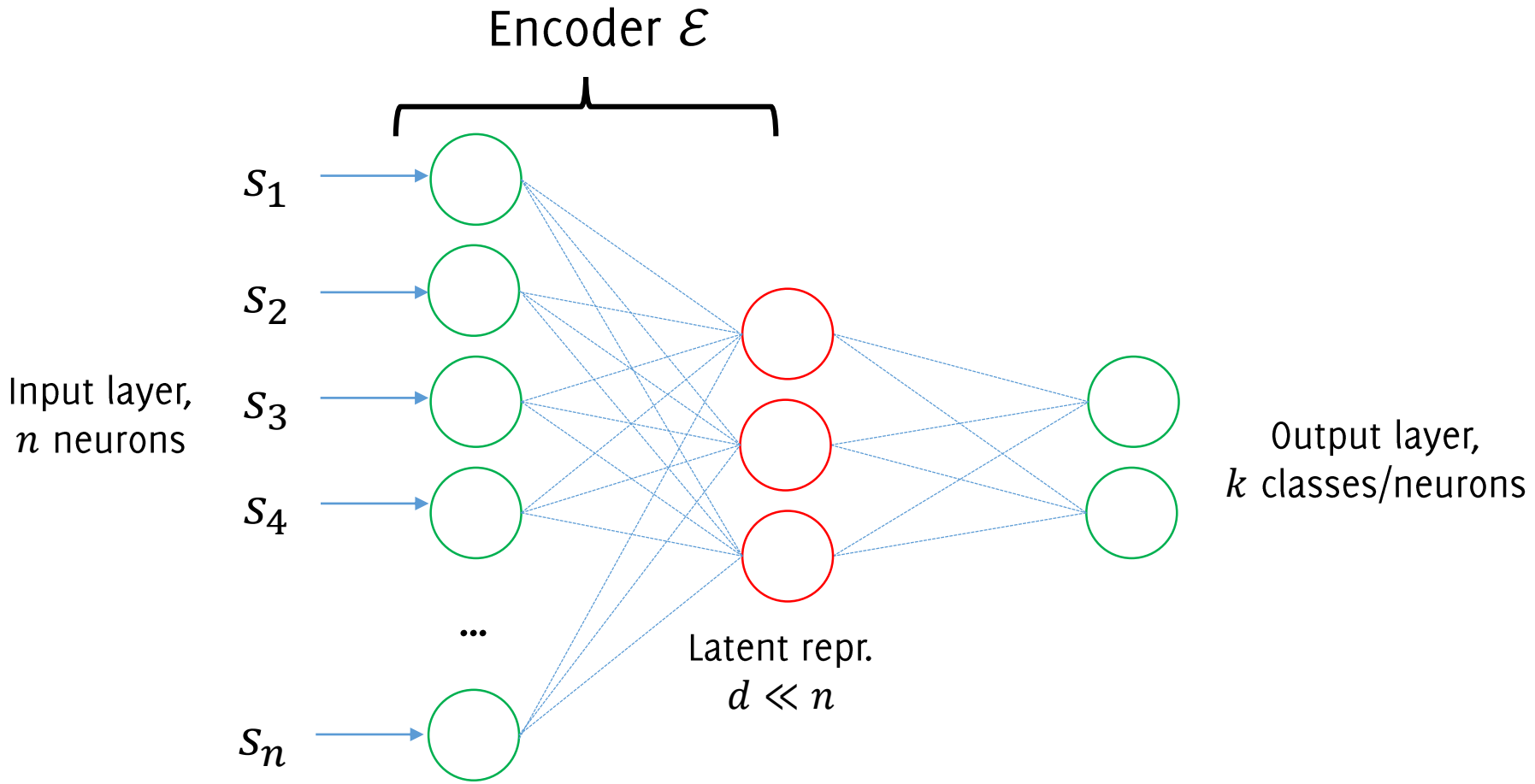
4) **Fine tune** the autoencoder using the few supervised samples provided L. This is perfectly in line with «Transfer Learning» and holds for whatever model.

If L is large enough, the encoder weights $\mathcal{E}$ can also be fine-tuned



Using Autoencoders for classifier initialization

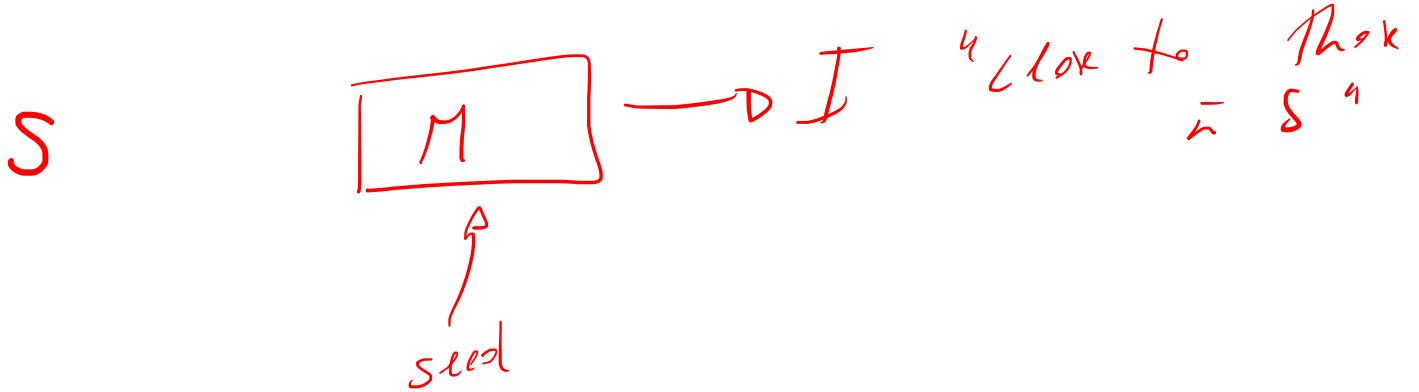
Autoencoders provide a **good initialization** (and reduce the risk of overfitting) because their latent vector is actually **a good (latent) representation** of the inputs used for training.



Generative Models:
Networks able to generate realistic images

Generative Models

Goal: generate, given a training set of images (data) S, other images (data) that are similar to those in S



What for generative models?

- Generative models can be used for **data augmentation**, simulation and planning
- Training generative models can also enable inference of latent representations that can be useful as general features
- Realistic samples for artwork, super-resolution, colorization, etc.
- You are getting close to the “holy grail” of modeling the distribution of natural images
    - This can be a very useful regularization prior in other problems or to perform anomaly detection

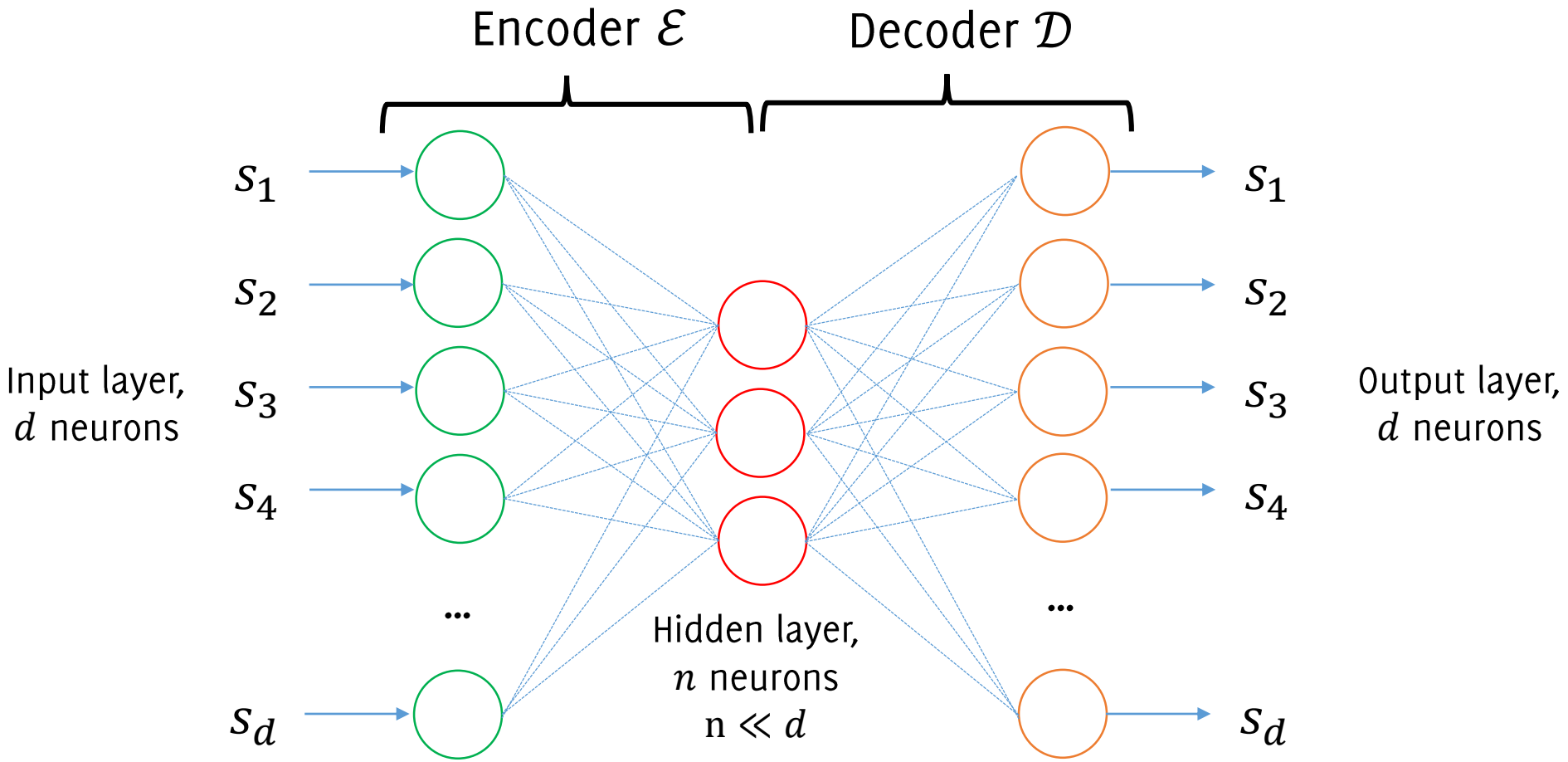
Images randomly generated by a GAN

Radford, et al “Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

What about using Autoencoders as Generative Models?

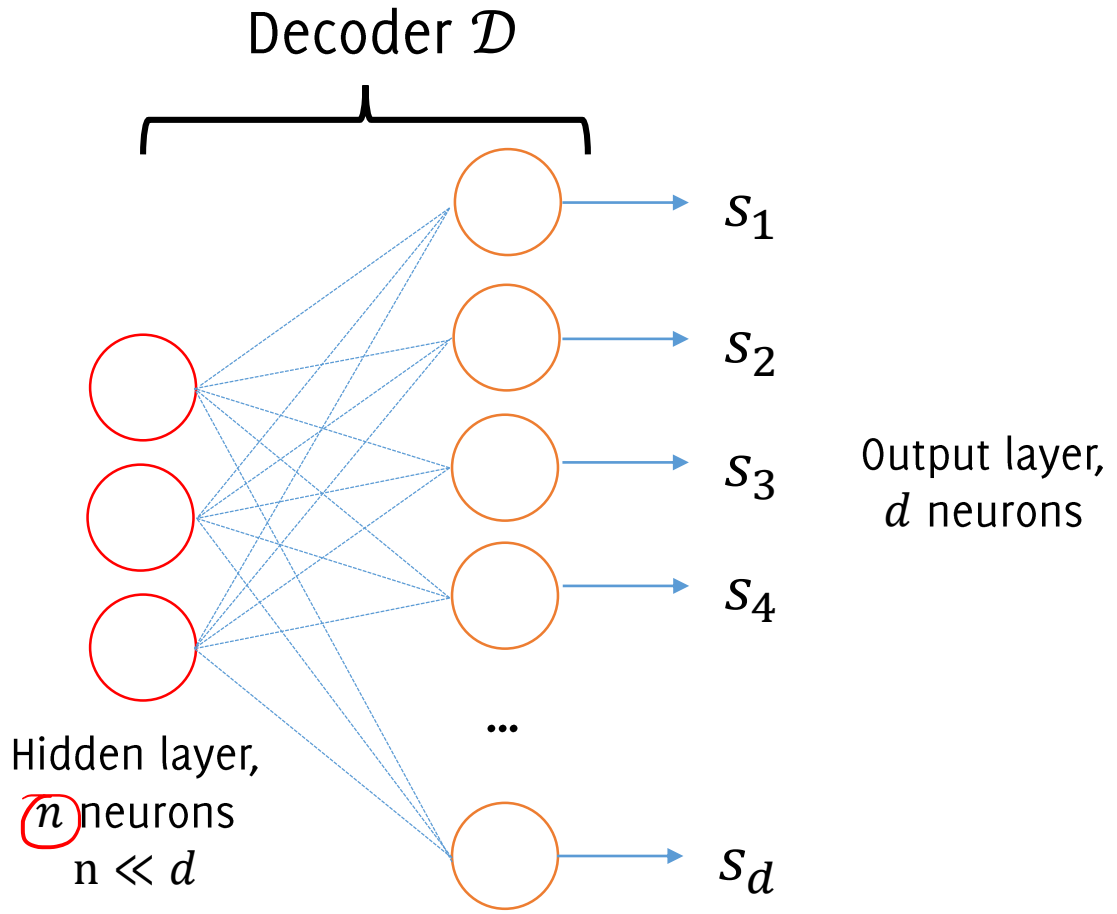
One option would be to

1) **train an autoencoder** on S



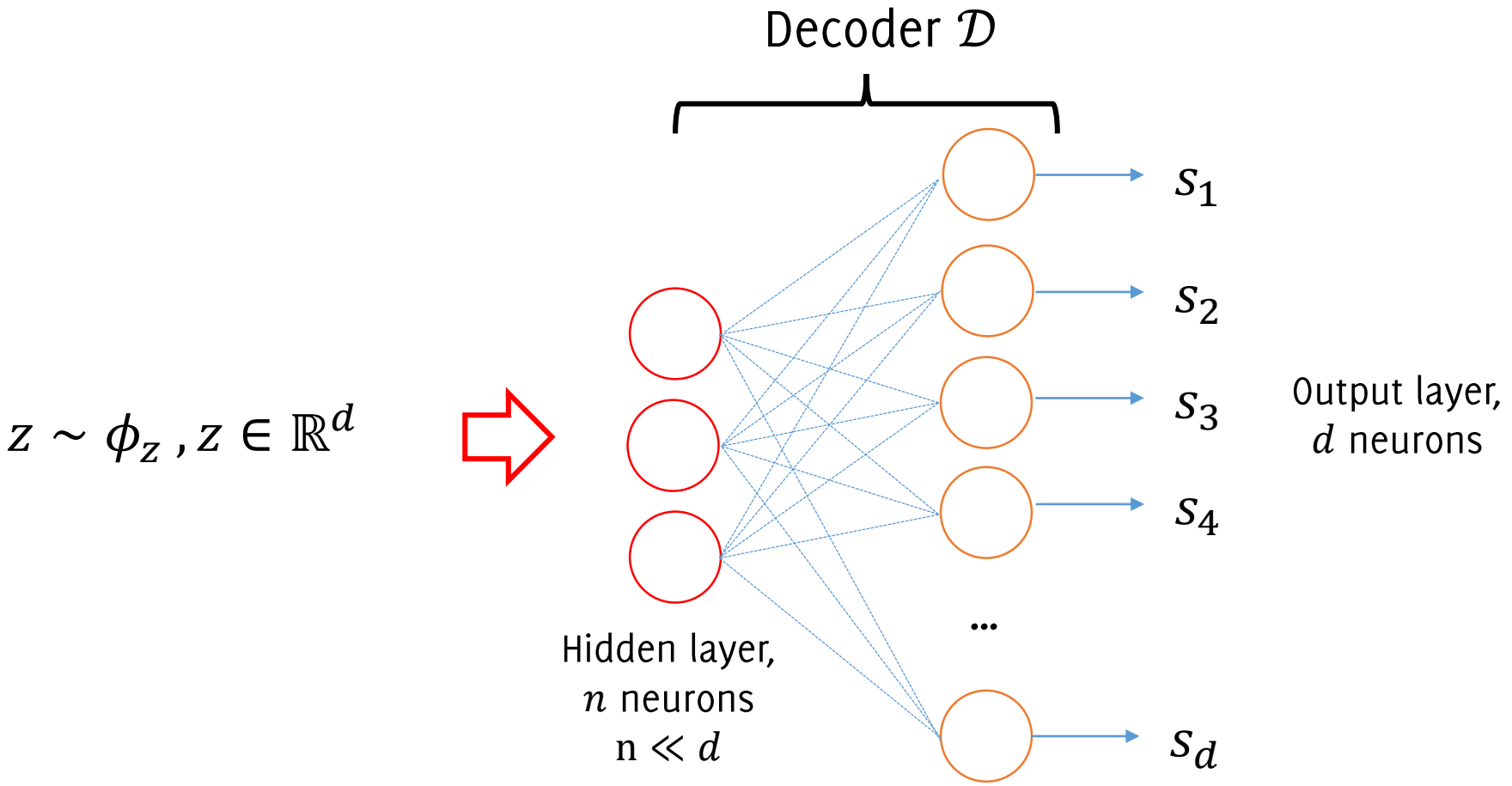
What about using Autoencoders as Generative Models?

2) **Discard the encoder**



What about using Autoencoders as Generative Models?

3) Draw random vectors $z \sim \phi_z$, to mimic «a new latent representation» and feed this to the decoder input

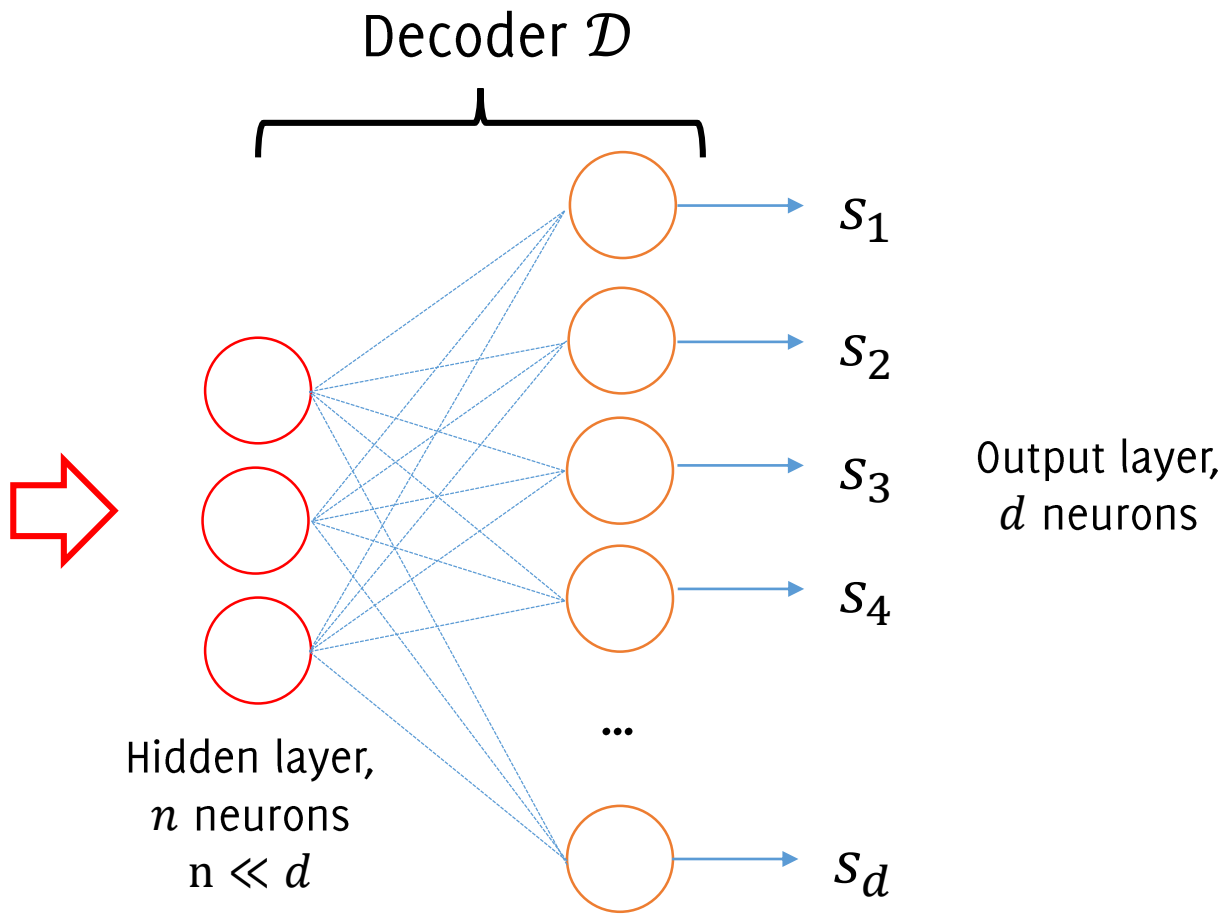


What about using Autoencoders as Generative Models?

This approach does not work since **we do not know the distribution of proper latent representation** (or it is very difficult to estimate).

Variational autoencoders leverages a prior over z to generate images using a similar architecture

$$z \sim \phi_z , z \in \mathbb{R}^d$$



Generative Adversarial Networks

A very effective way to generate images

Generative Adversarial Networks (GAN)

The GAN approach:

- Do not look for an explicit density model ϕ_S describing the manifold of natural images.
- Just find out a model able to generate samples that «looks like» training samples $S \subset \mathbb{R}^n$

Instead of sampling from ϕ_S, just:

- **Sample a seed** from a known distribution ϕ_z . This is defined a priori and also referred to as **noise.**
- Feed this seed to a learned transformation that generates realistic samples, as if they were drawn from ϕ_S

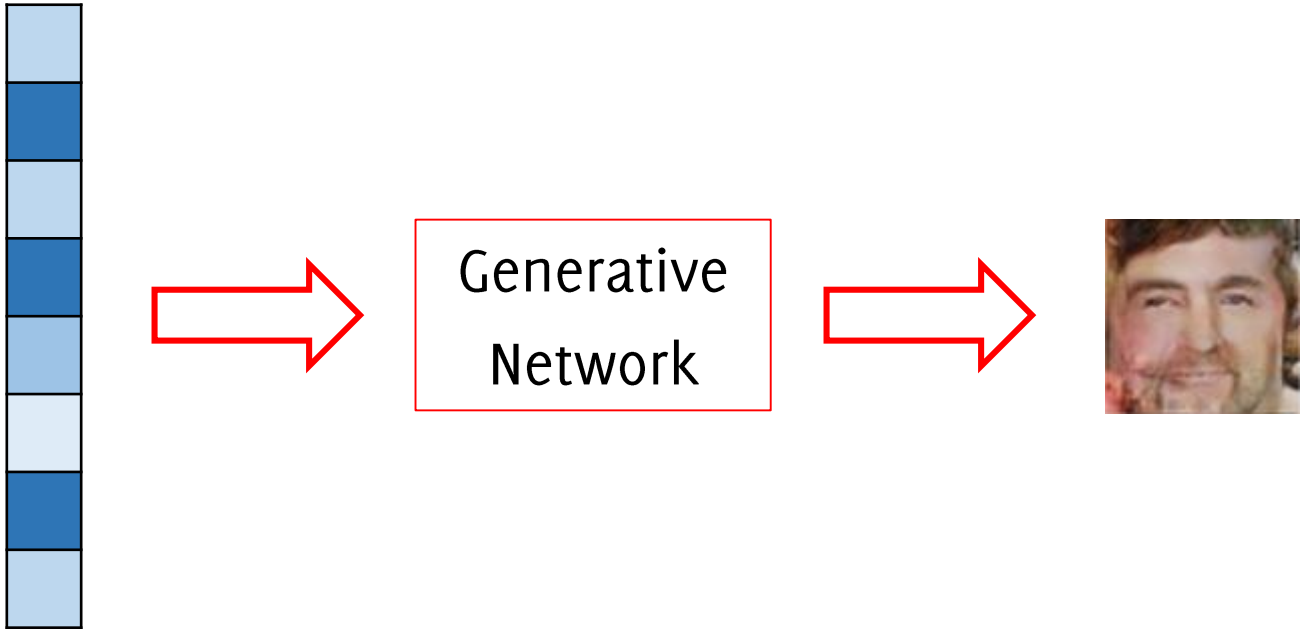
Use a neural network to learn this transformation

The neural network is going to be **trained in an unsupervised manner, no label needed**

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

Generative Adversarial Networks (GAN)

The GAN approach:



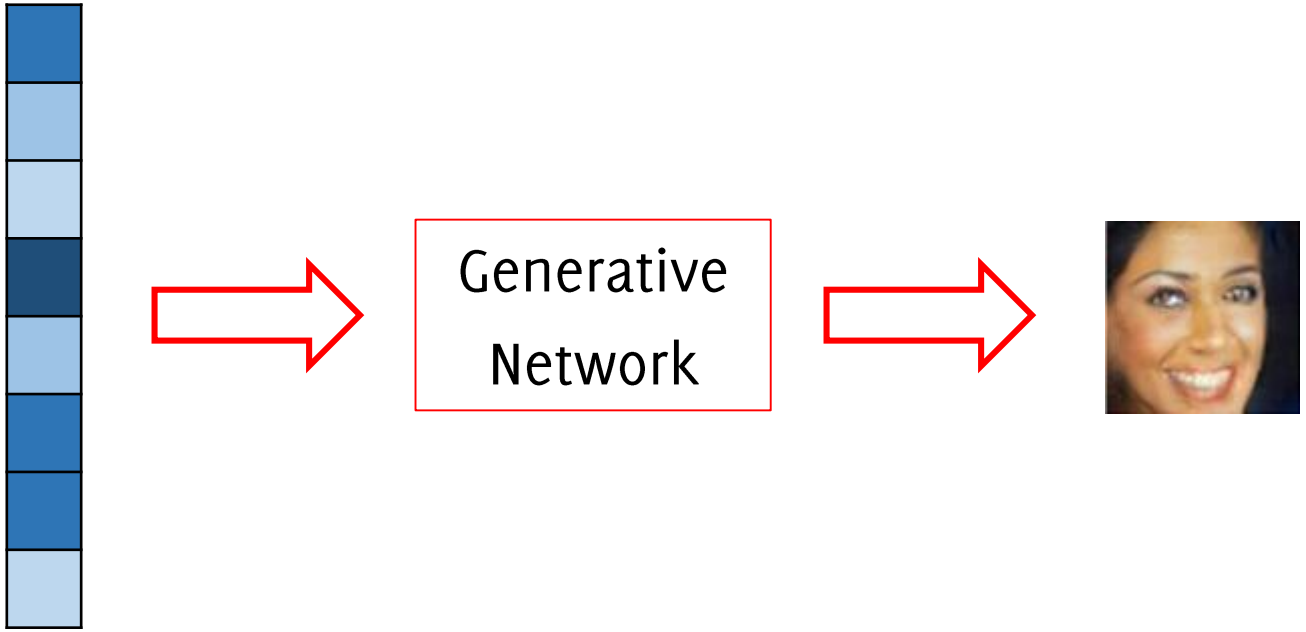
$z \sim \phi_z$

Draw a sample from
the *noise distribution*

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

Generative Adversarial Networks (GAN)

The GAN approach:



$z \sim \phi_z$

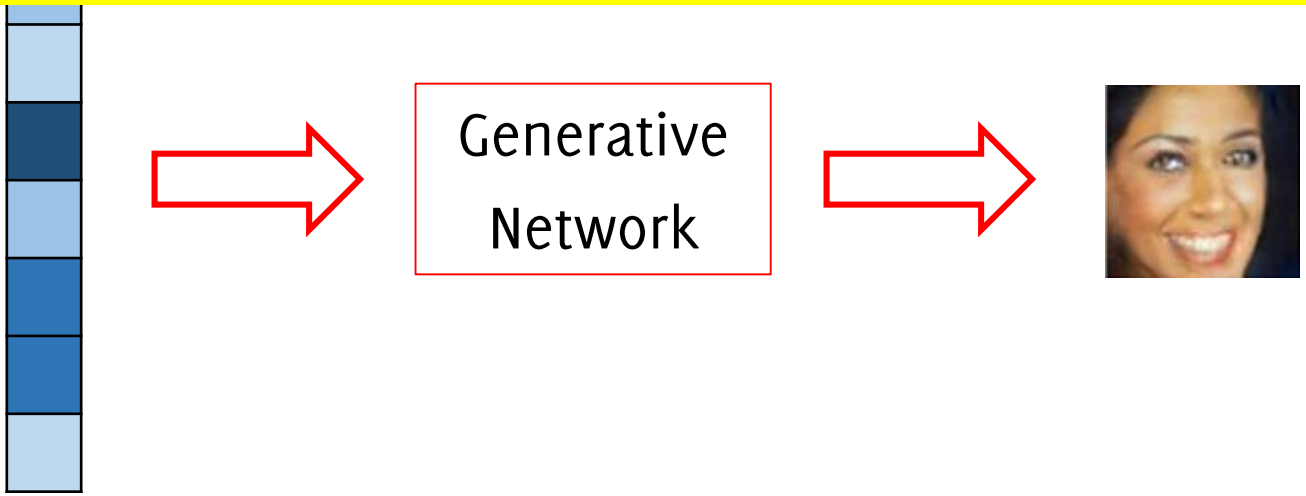
Draw a sample from
the noise distribution

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

Generative Adversarial Networks (GAN)

The GAN approach:

The biggest challenge is to define a suitable loss for assessing whether the output is a realistic image or not



$z \sim \phi_z$

Draw a sample from
the noise distribution

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

Generative Adversarial Networks (GAN)

The GAN solution: Train a pair of neural networks addressing two different tasks that compete in a sort of **two player** (adversarial) **game.**

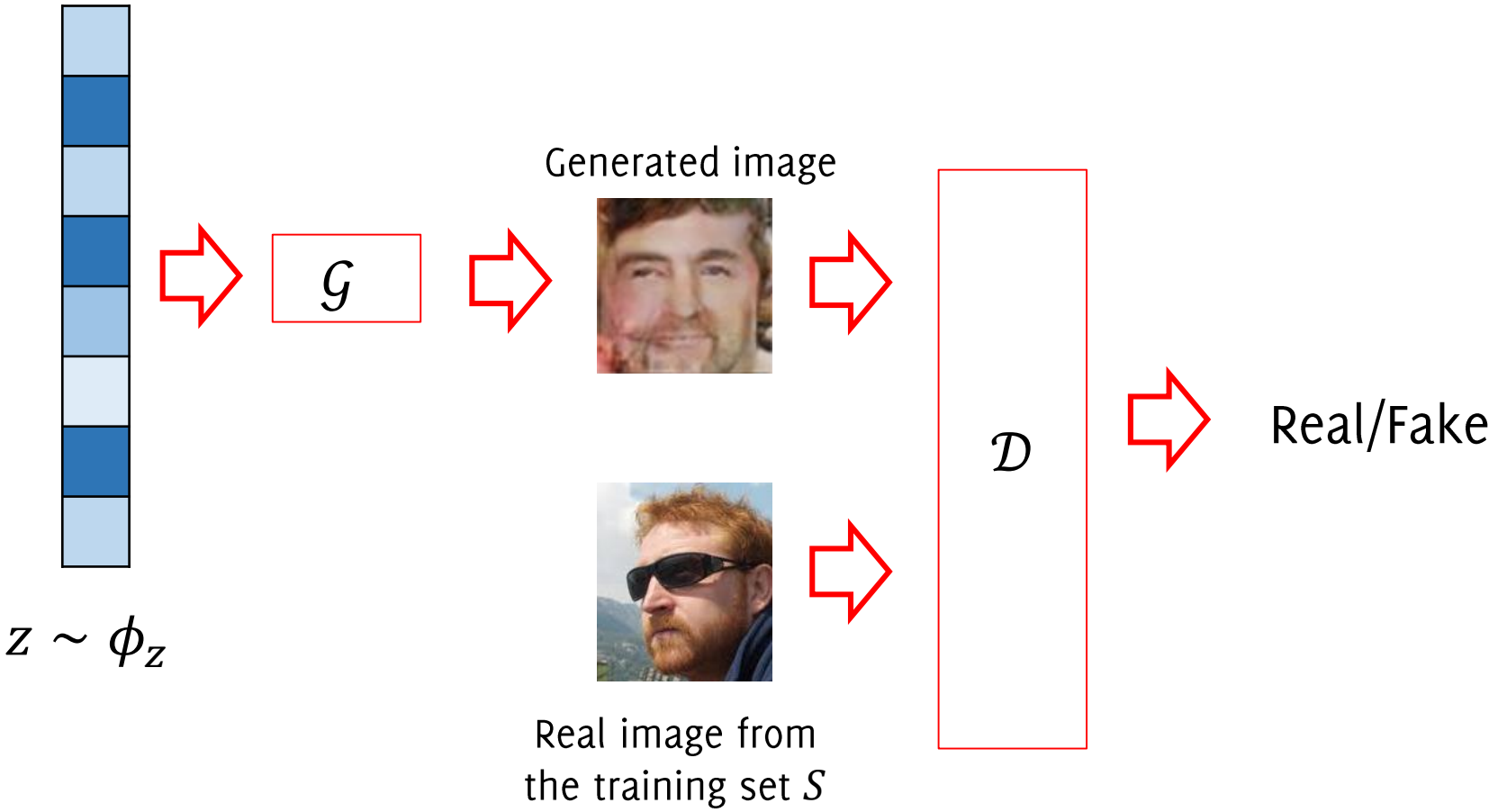
These models are:

- **Generator $\mathcal{G}$ that produces realistic samples** e.g. taking as input some random noise. $\mathcal{G}$ tries to fool the discriminator
- **Discriminator $\mathcal{D}$** that takes as input an image **and assess whether it is real or generated by $\mathcal{G}$**

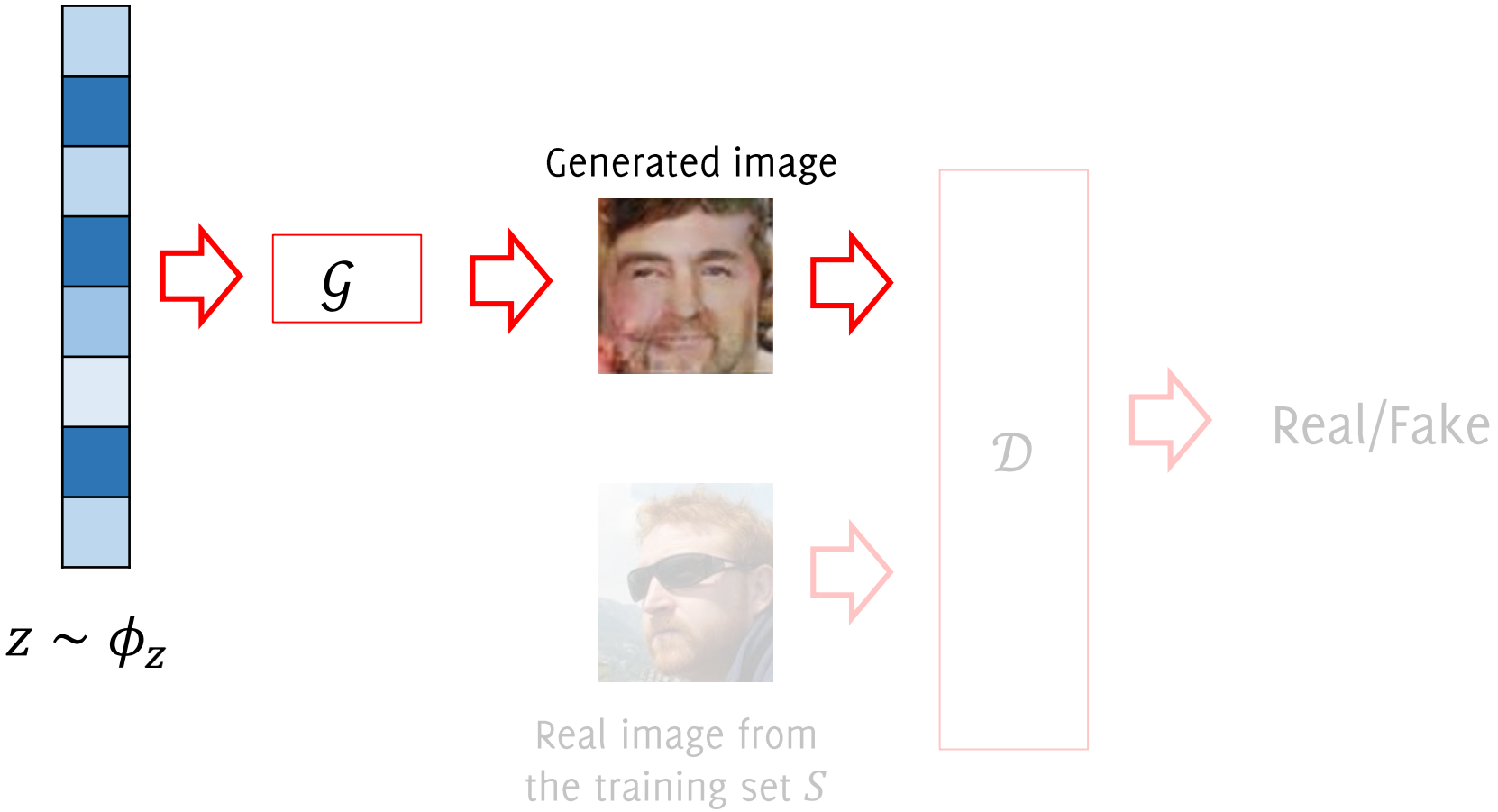
Train the two and **at the end, keep only $\mathcal{G}$**

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

GAN Architecture



Using GAN



Discriminator $\mathcal{D}$ is completely useless and as such dropped. After a successful GAN training, $\mathcal{D}$ is not able to distinguish the real/fake

GAN: Setting up the stage

Both $\mathcal{D}$ and $\mathcal{G}$ are conveniently chosen as MLP or CNN

Our networks take as input:

- $\mathcal{D} = \mathcal{D}(\boldsymbol{s}, \theta_d)$,
- $\mathcal{G} = \mathcal{G}(\boldsymbol{z}, \theta_g)$,

This notation is meant to visualize what are the NN parameters.. The network is taking a single input

θ_g and θ_d are network parameters, $\boldsymbol{s} \in \mathbb{R}^n$ is an input image (either real or generated by $\mathcal{G}$) and $\boldsymbol{z} \in \mathbb{R}^d$ is some random noise to be fed to the generator.

Our networks give as output:

- $\mathcal{D}(\cdot, \theta_d)$: $\mathbb{R}^n \to [0,1]$ gives as output the posterior for the input be a true image
- $\mathcal{G}(\cdot, \theta_d)$: $\mathbb{R}^d \to \mathbb{R}^n$ gives as output the generated image

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

GAN Training

A good discriminator is such:

- $\mathcal{D}(\boldsymbol{s}, \theta_d)$ is maximum when $\mathbf{s} \in S$ (true image from the training set)
- $1 - \mathcal{D}(\boldsymbol{s}, \theta_d)$ is maximum when $\mathbf{s}$ was generated from $\mathcal{G}$
- $1 - \mathcal{D}\big(\mathcal{G}\big(\boldsymbol{z}, \theta_g\big), \theta_d\big)$ is maximum when $\boldsymbol{z} \sim \phi_Z$

Training $\mathcal{D}$ consists in maximizing the **binary cross-entroy**

$$\max_{\theta_d}\big(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}\big[\log(1 - \mathcal{D}\big(\mathcal{G}\big(\boldsymbol{z}, \theta_g\big), \theta_d\big))\big]\big)$$

Written using expectations

GAN Training

A good discriminator is such:

- $\mathcal{D}(\boldsymbol{s}, \theta_d)$ is maximum when $\mathbf{s} \in S$ (true image from the training set)
- $1 - \mathcal{D}(\boldsymbol{s}, \theta_d)$ is maximum when $\mathbf{s}$ was generated from $\mathcal{G}$
- $1 - \mathcal{D}\big(\mathcal{G}\big(\boldsymbol{z}, \theta_g\big), \theta_d\big)$ is maximum when $\boldsymbol{z} \sim \phi_Z$

Training $\mathcal{D}$ consists in maximizing the binary cross-entroy

$$\max_{\theta_d}\big(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}\big[\log(1 - \mathcal{D}\big(\mathcal{G}\big(\boldsymbol{z}, \theta_g\big), \theta_d\big))\big]\big)$$

This has to be 1 since $s \sim \phi_S$, thus images are real

This has to be 0 since $\mathcal{G}\big(\boldsymbol{z}, \theta_g\big)$ is a generated (fake) image

GAN Training

A good discriminator is such:

- $\mathcal{D}(\boldsymbol{s}, \theta_d)$ is maximum when $\mathbf{s} \in S$ (true image from the training set)
- $1 - \mathcal{D}(\boldsymbol{s}, \theta_d)$ is maximum when $\mathbf{s}$ was generated from $\mathcal{G}$
- $1 - \mathcal{D}\big(\mathcal{G}\big(\boldsymbol{z}, \theta_g\big), \theta_d\big)$ is maximum when $\boldsymbol{z} \sim \phi_Z$

Training $\mathcal{D}$ consists in maximizing the binary cross-entroy

$$\max_{\theta_d}\big(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}\big[\log(1 - \mathcal{D}\big(\mathcal{G}\big(\boldsymbol{z}, \theta_g\big), \theta_d\big))\big]\big)$$

A good generator $\mathcal{G}$ makes $\mathcal{D}$ to fail, thus minimizes the above

$$\min_{\theta_g}\max_{\theta_d}\big(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}\big[\log(1 - \mathcal{D}\big(\mathcal{G}\big(\boldsymbol{z}, \theta_g\big), \theta_d\big))\big]\big)$$

GAN Training

Solve by an iterative numerical approach

$$\min_{\theta_g} \max_{\theta_d} \left(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}\left[\log(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}, \theta_g\right), \theta_d\right))\right]\right)$$

GAN Training

Solve by an iterative numerical approach

$$\min_{\theta_g} \max_{\theta_d} \left(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}\left[\log(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}, \theta_g\right), \theta_d\right))\right]\right)$$

Alternate:

- k-steps of Stochastic Gradient Ascent w.r.t. θ_d , keep θ_g fixed and solve

$$\max_{\theta_d} \left(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}\left[\log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}, \theta_g\right), \theta_d\right)\right)\right]\right)$$

- 1-step of Stochastic Grandient Descent w.r.t. θ_g being θ_d fixed

$$\min_{\theta_g} \left(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}\left[\log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}, \theta_g\right), \theta_d\right)\right)\right]\right)$$

GAN Training

for $i = 1 \ldots$ #number of epochs

for k −times # gradient ascent steps for θ_d

- Draw a minibatch $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ of noise realization
- Sample a minibatch of images $\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m\}$
- Update θ_d by stochastic gradient ascend:

$$\nabla_{\theta_d} \left[\sum_i \log \mathcal{D}(\boldsymbol{s_i}, \theta_d) + \log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}_i, \theta_g\right), \theta_d\right)\right) \right]$$

Draw a minibatch $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ of noise realizations # gradient descent steps for θ_g

Update $\mathcal{G}$ by stochastic gradient descent:

$$\nabla_{\theta_g} \left[\sum_i \log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}_i, \theta_g\right), \theta_d\right)\right) \right]$$

GAN Training

This was presented as a best practice, latest GANs such as Wasserstein GANs do not use this

for $i = 1 \ldots$ #number of epochs

for k −times # gradient ascent steps for θ_d

- Draw a minibatch $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ of noise realization
- Sample a minibatch of images $\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m\}$
- Update θ_d by stochastic gradient ascend:

$$\nabla_{\theta_d} \left[\sum_i \log \mathcal{D}(\boldsymbol{s_i}, \theta_d) + \log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}_i, \theta_g\right), \theta_d\right)\right) \right]$$

Draw a minibatch $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ of noise realizations # gradient descent steps for θ_g

Update $\mathcal{G}$ by stochastic gradient descent:

$$\nabla_{\theta_g} \left[\sum_i \log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}_i, \theta_g\right), \theta_d\right)\right) \right]$$

One of the many GAN Training «trick»

1-step of Stochastic Grandient Descent w.r.t. θ_g being θ_d fixed

$$\min_{\theta_g} \left(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}\left[\log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}, \theta_g\right), \theta_d\right)\right)\right]\right)$$

and since the first term does not depend on θ_g, this consists in minimizing

$$\min_{\theta_g} \left(\mathrm{E}_{z\sim\phi_Z}\left[\log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}, \theta_g\right), \theta_d\right)\right)\right]\right)$$

or equivalently maximizing

$$\max_{\theta_g} \left(\mathrm{E}_{z\sim\phi_Z}\left[\log\left(\mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}, \theta_g\right), \theta_d\right)\right)\right]\right)$$

GAN Training

Maximizing w.r.t θ_g (instead of minimizing as before) does not modify the global solution of the min-max problem, but provides a stronger gradient during the early learning stages

Solve by an iterative numerical approach

$$\min_{\theta_g} \max_{\theta_d} \left(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z}[\log(1 - \mathcal{D}(\mathcal{G}(\boldsymbol{z}, \theta_g), \theta_d))] \right)$$

Alternate:

- k-steps of Stochastic Gradient Ascent w.r.t. θ_d to solve

$$\max_{\theta_d} \left(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z} \left[\log \left(1 - \mathcal{D}(\mathcal{G}(\boldsymbol{z}, \theta_g), \theta_d) \right) \right] \right)$$

- 1-step of Stochastic Grandient Ascent w.r.t. θ_g being θ_d fixed

$$\min_{\theta_g} \left(\mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(\boldsymbol{s}, \theta_d)] + \mathrm{E}_{z\sim\phi_Z} \left[\log \left(1 - \mathcal{D}(\mathcal{G}(\boldsymbol{z}, \theta_g), \theta_d) \right) \right] \right)$$

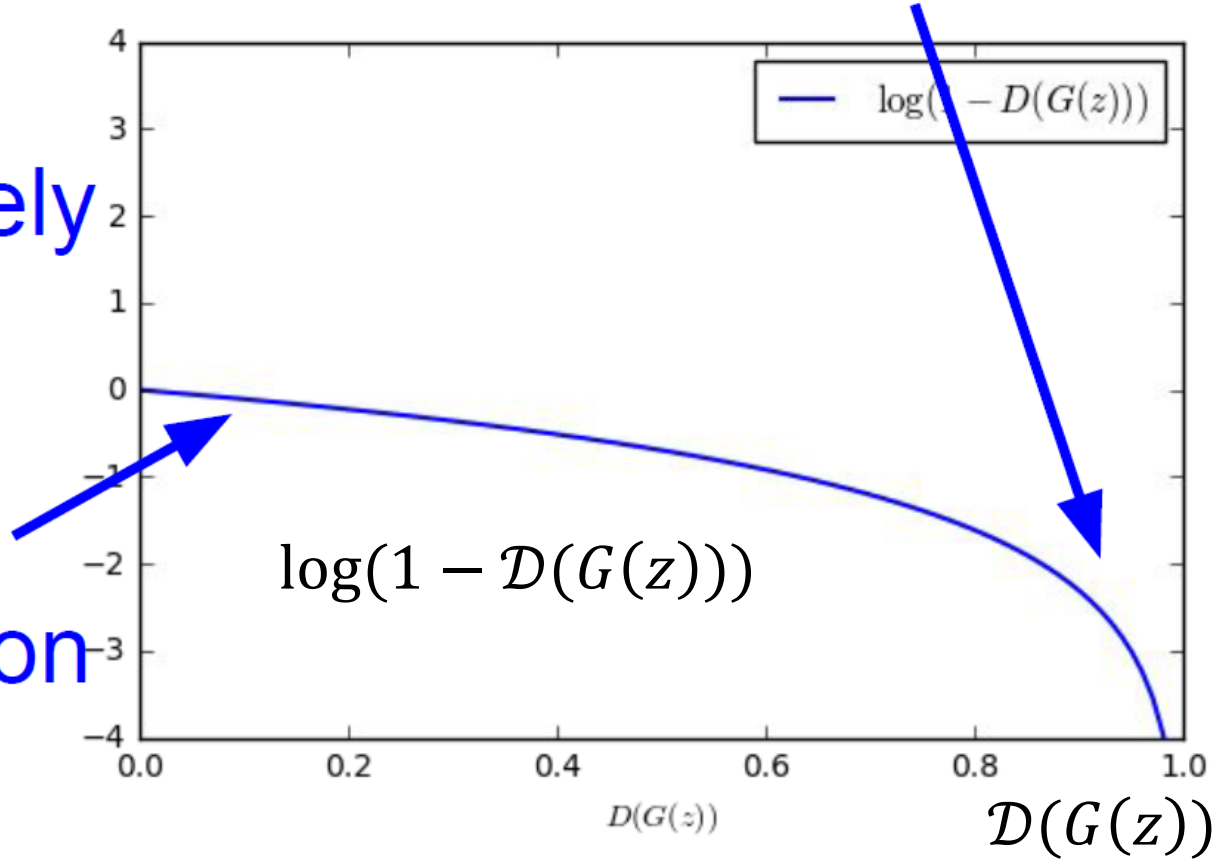
and since the first term does not depend on θ_g, this consists in maximizing

$$\max_{\theta_g} \left(\mathrm{E}_{z\sim\phi_Z} \left[\log \left(\mathcal{D}(\mathcal{G}(\boldsymbol{z}, \theta_g), \theta_d) \right) \right] \right)$$

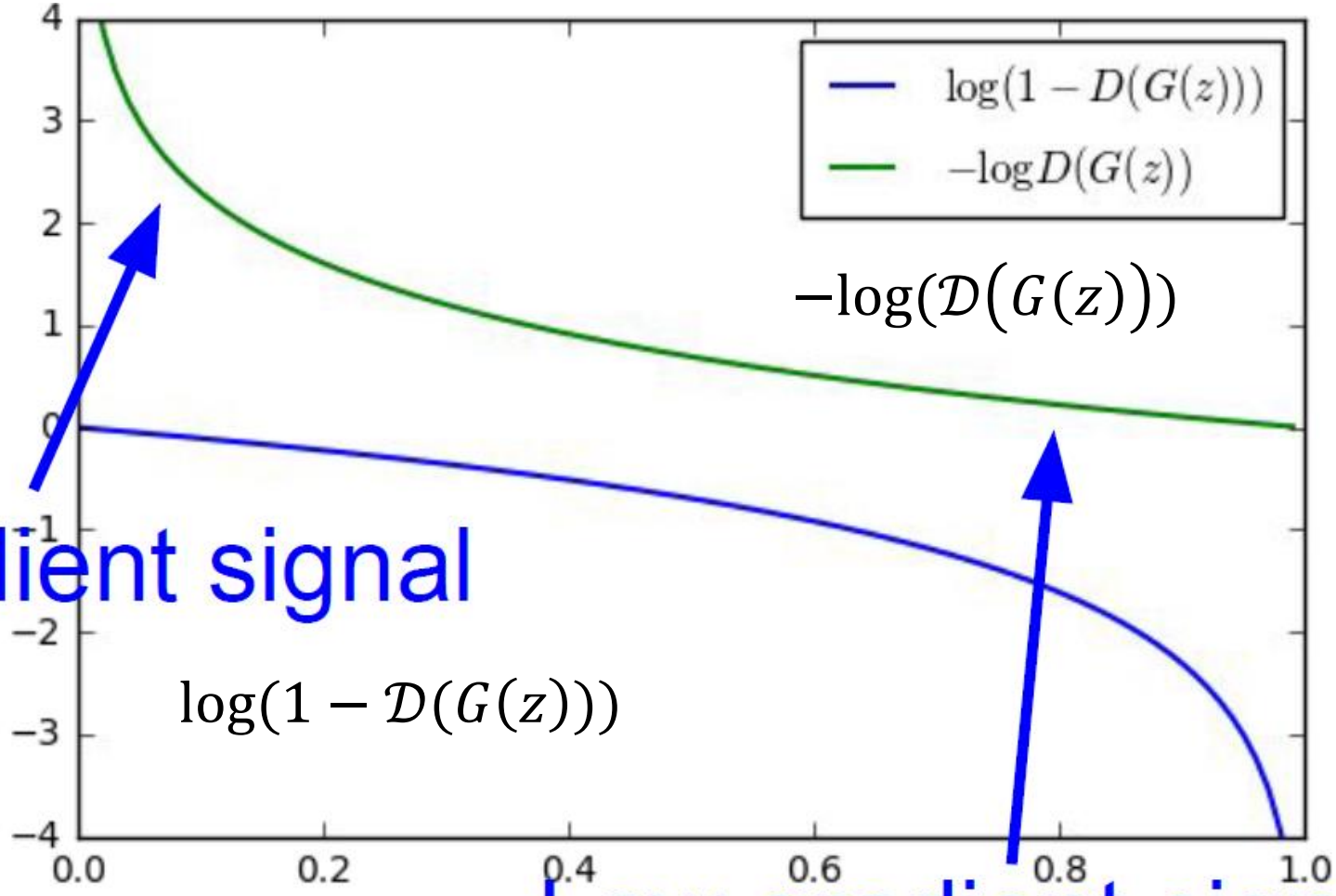
During early learning stages, when G is poor, D can reject samples with high confidence because they are clearly different from the training data (thus $D(G(z)) \approx 0$). In this case, $\log(1 - D(G(z)))$ is flat, thus has very low gradient.

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!



Rather than training G to minimize $log(1 - D(G(z)))$ *we can train G to maximize* $\log(\mathcal{D}(G(z)))$ *[or as in this figure, minimize* $-\log(\mathcal{D}(G(z)))$*] This objective function results in the same fixed point of the dynamics of G and D, but provides much stronger gradients early in learning.*



Algorithm outline

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Goodfellow, I. et al "Generative adversarial nets" NIPS 2014

GAN Training

for $i = 1 \ldots$ #number of epochs

for k −times # gradient ascent steps for θ_d

- Draw a minibatch $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ of noise realization
- Sample a minibatch of images $\{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_m\}$
- Update θ_d by stochastic gradient ascend:

$$\nabla_{\theta_d} \left[\sum_i \log \mathcal{D}(\boldsymbol{s_i}, \theta_d) + \log\left(1 - \mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}_i, \theta_g\right), \theta_d\right)\right) \right]$$

Draw a minibatch $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_m\}$ of noise realizations # gradient descent steps for θ_g

Update $\mathcal{G}$ by stochastic gradient **ascent**:

$$\nabla_{\theta_g} \left[\sum_i \log\left(\mathcal{D}\left(\mathcal{G}\left(\boldsymbol{z}_i, \theta_g\right), \theta_d\right)\right) \right]$$

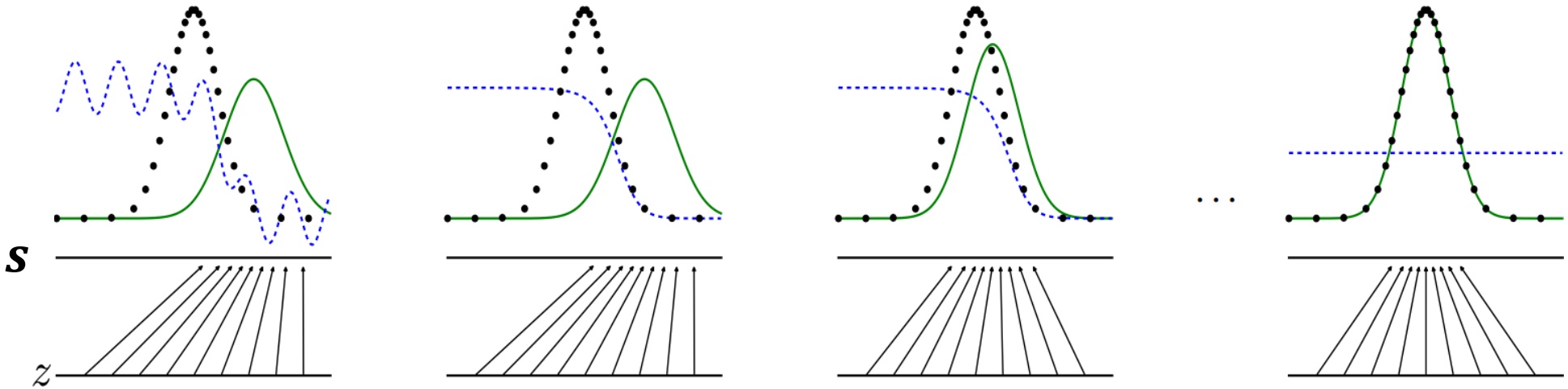
Illustration of the GAN Training Process

In this illustration $\mathbb{R}^d$ and $\mathbb{R}^n$ are collapsed into 1d points this allows also the visualization of their distribution

$\boldsymbol{\phi_s}$, $\boldsymbol{s}$ real

$\boldsymbol{\phi_{\mathcal{G}(z)}}$ $\mathcal{G}(\boldsymbol{z})$ fake

$\mathcal{D}(\cdot)$ $\mathcal{D}$ posterior



Goodfellow, I. et al "Generative adversarial nets" NIPS 2014

At the end of the day…

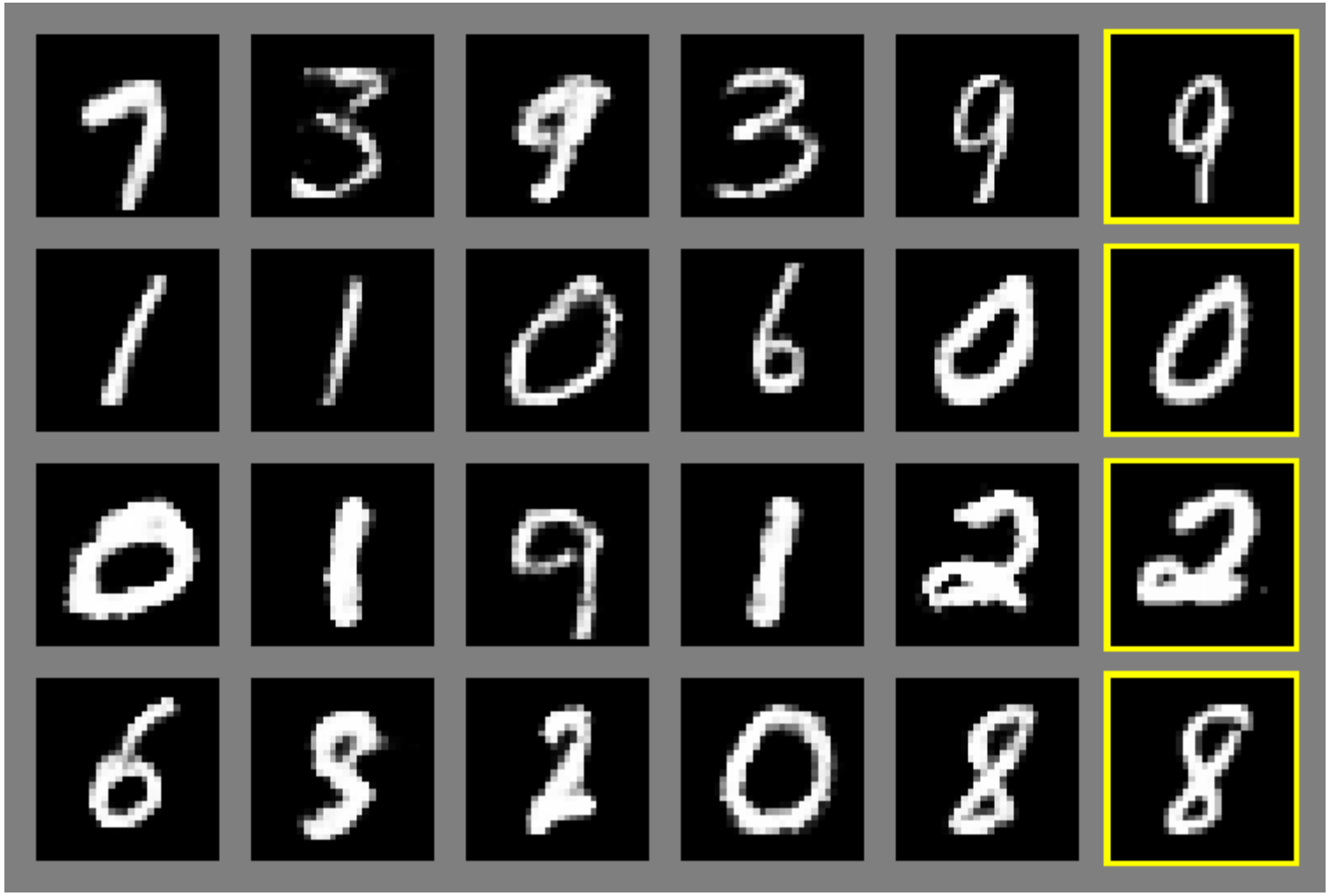
The discriminator $\mathcal{D}$ is discarded

The generator $\mathcal{G}$ and ϕ_Z are preserved as generative model

Remarks:

- **The training is rather unstable**, need to carefully synchronize the two steps (many later works in this direction, e.g. Wasserstein GAN)
- **Training by standard tools**: backpropagation and dropout
- Theoretical results provided
- **Generator does not use S directly during training**
- **Generator performance is difficult to assess quantitatively**
- There is **no explicit expression for the generator**, it is provided in an implicit form -› you cannot compute the likelihood of a sample w.r.t. the learned GAN

MNIST

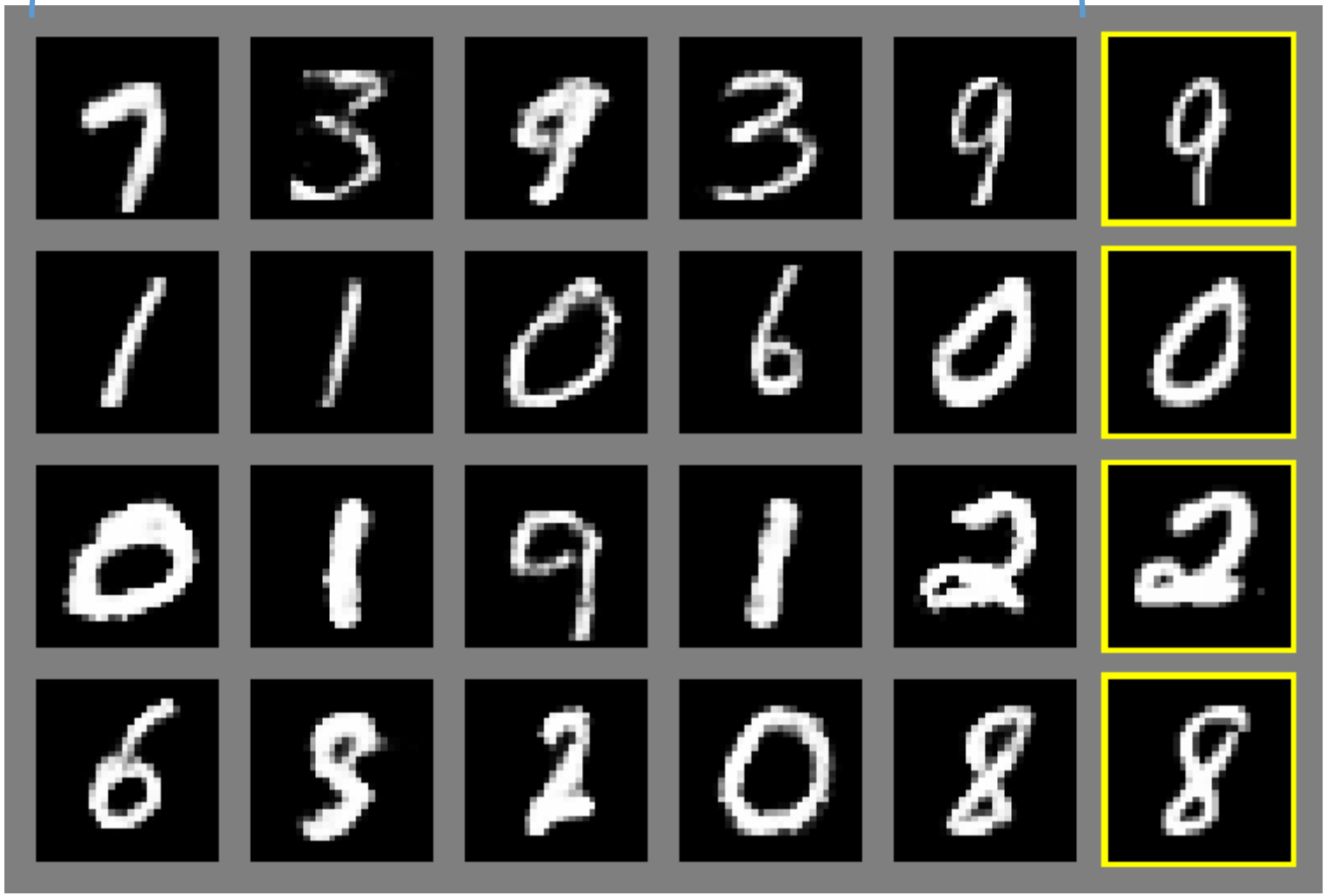


Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

MNIST

Generated samples

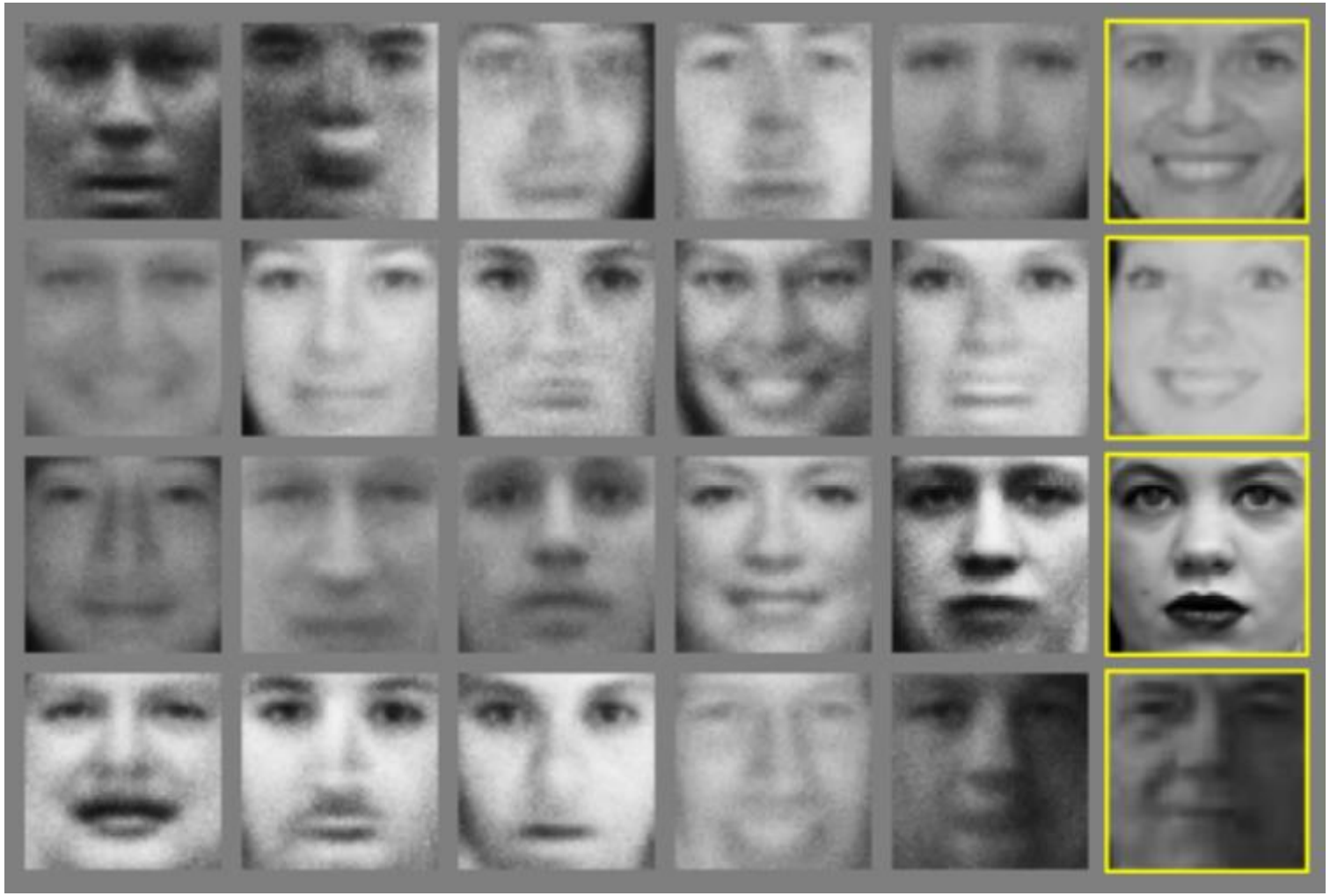
nearest training sample to the second-last column



This GAN generates realistic training samples without memorizing the training set

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

Toronto Face Database (TFD)



Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

CIFAR-10

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014

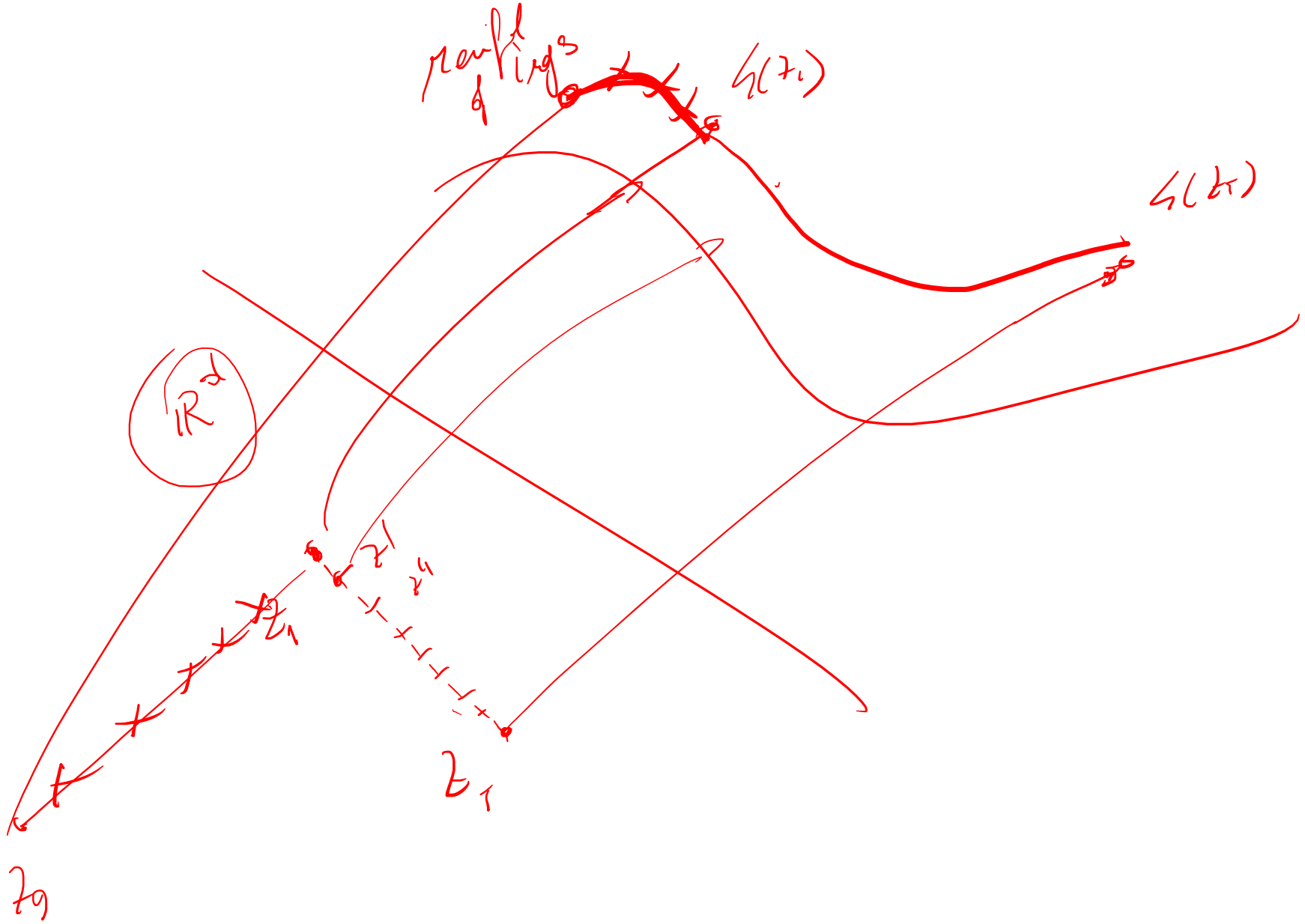
Outputs of interpolated trajectories

Select two noise realization z_1 and z_2 yielding reasonable outputs, and interpolate among the two. Generate the images of intermediate values

$\mathcal{G}(z_1)$ $\mathcal{G}(z_2)$

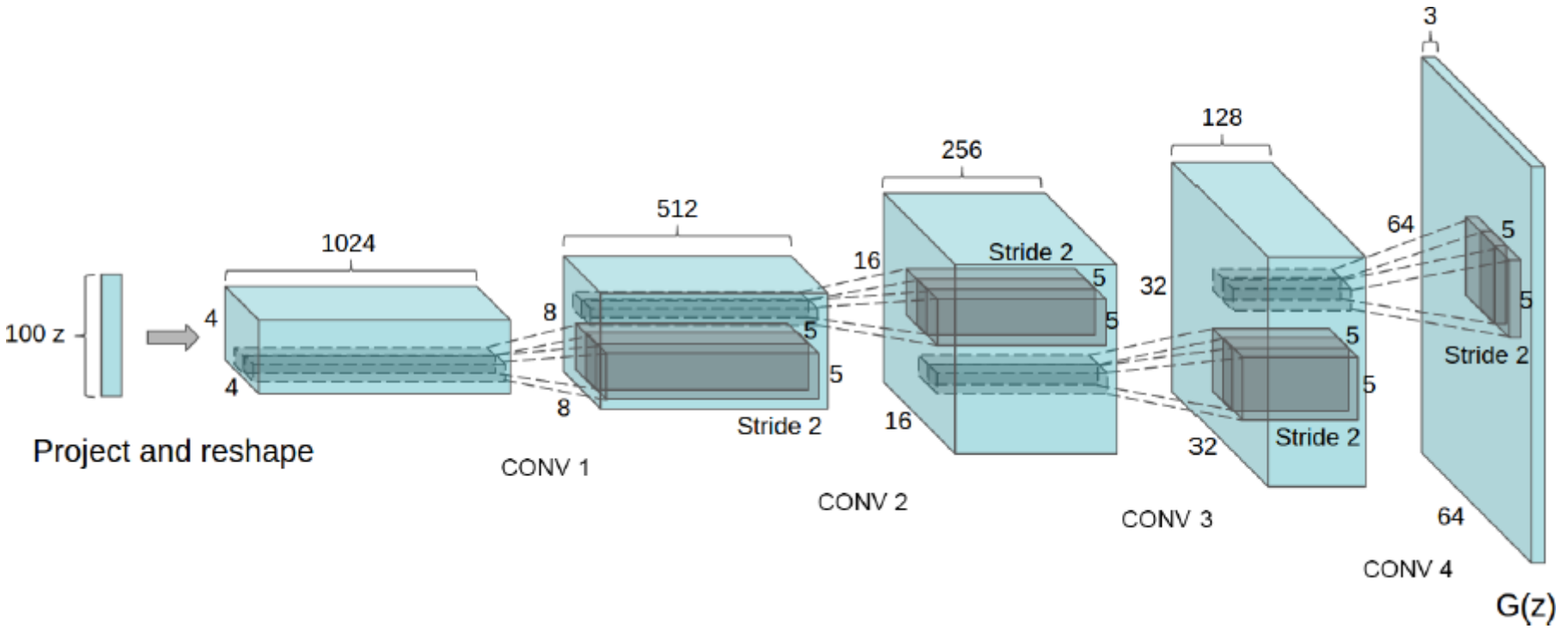
$\mathcal{G}(z_3)$ $\mathcal{G}(z_4)$

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, ... & Bengio, Y. Generative adversarial nets NIPS 2014


mapping
$\gamma(t_1)$
$\gamma(t_2)$
$\mathbb{R}^2$
z_0
z_1
z'
z''
z_1

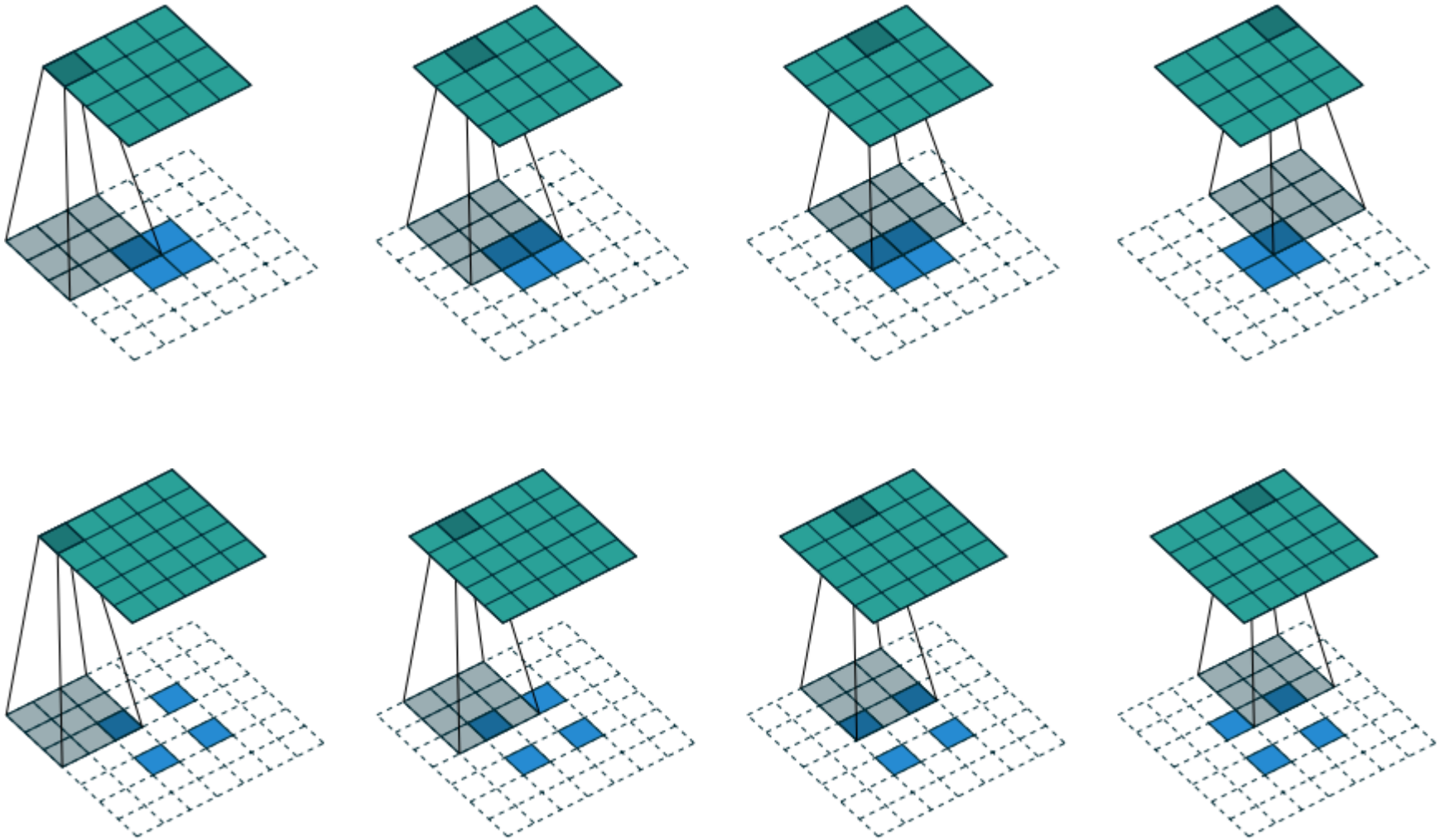
GANs have much improved in the last few years

DC-GAN: Deep Convolutional GANs



Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

How is it possible to "increase the network" size?



Dumoulin V., Visin F. - *A guide to convolution arithmetic for deep learning*

GANs have much improved in the last few years

Images generated after 1 training epochs

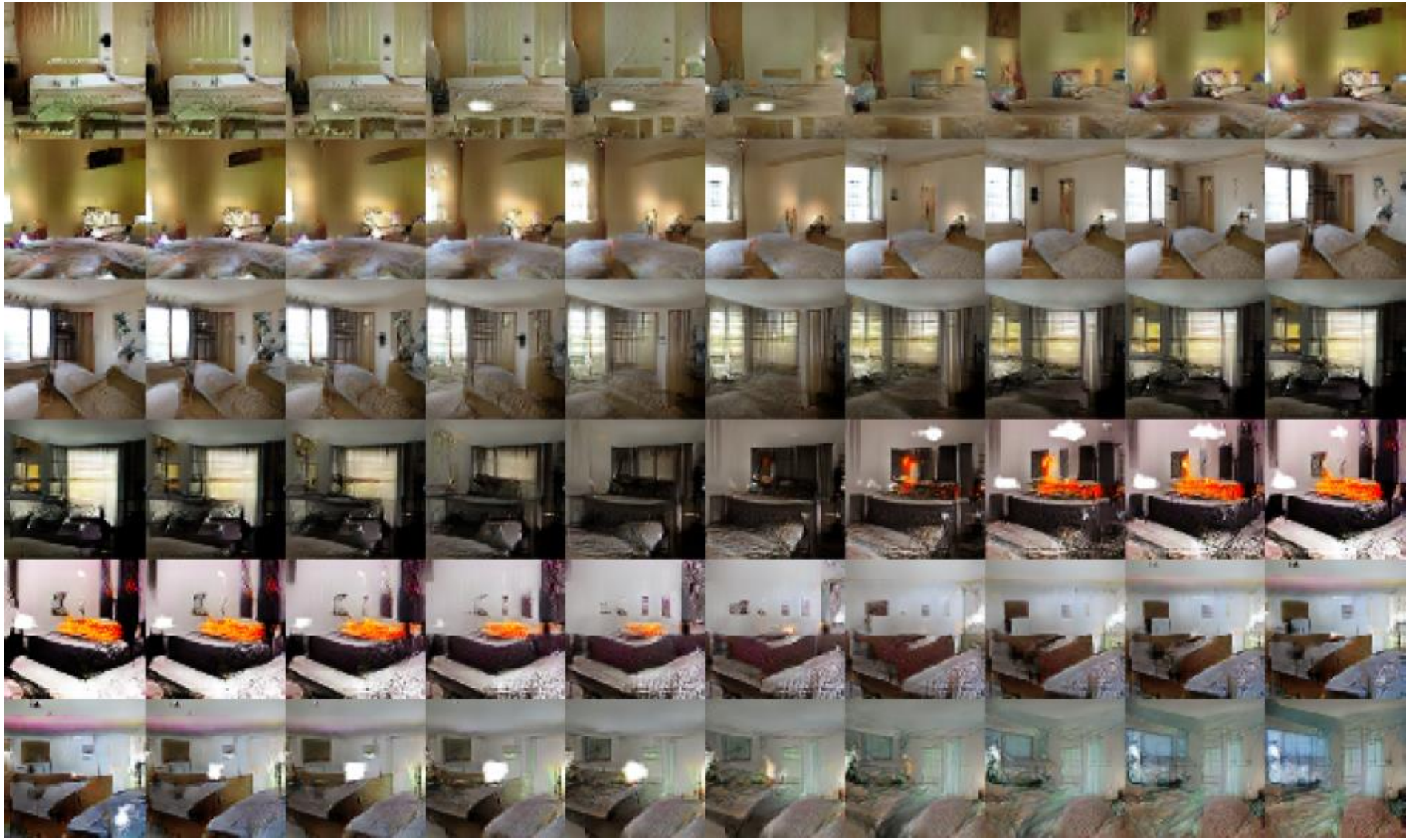
Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

GANs have much improved in the last few years

Images generated after 5 training epochs

Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

Interpolation between a series of 9 random points



Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

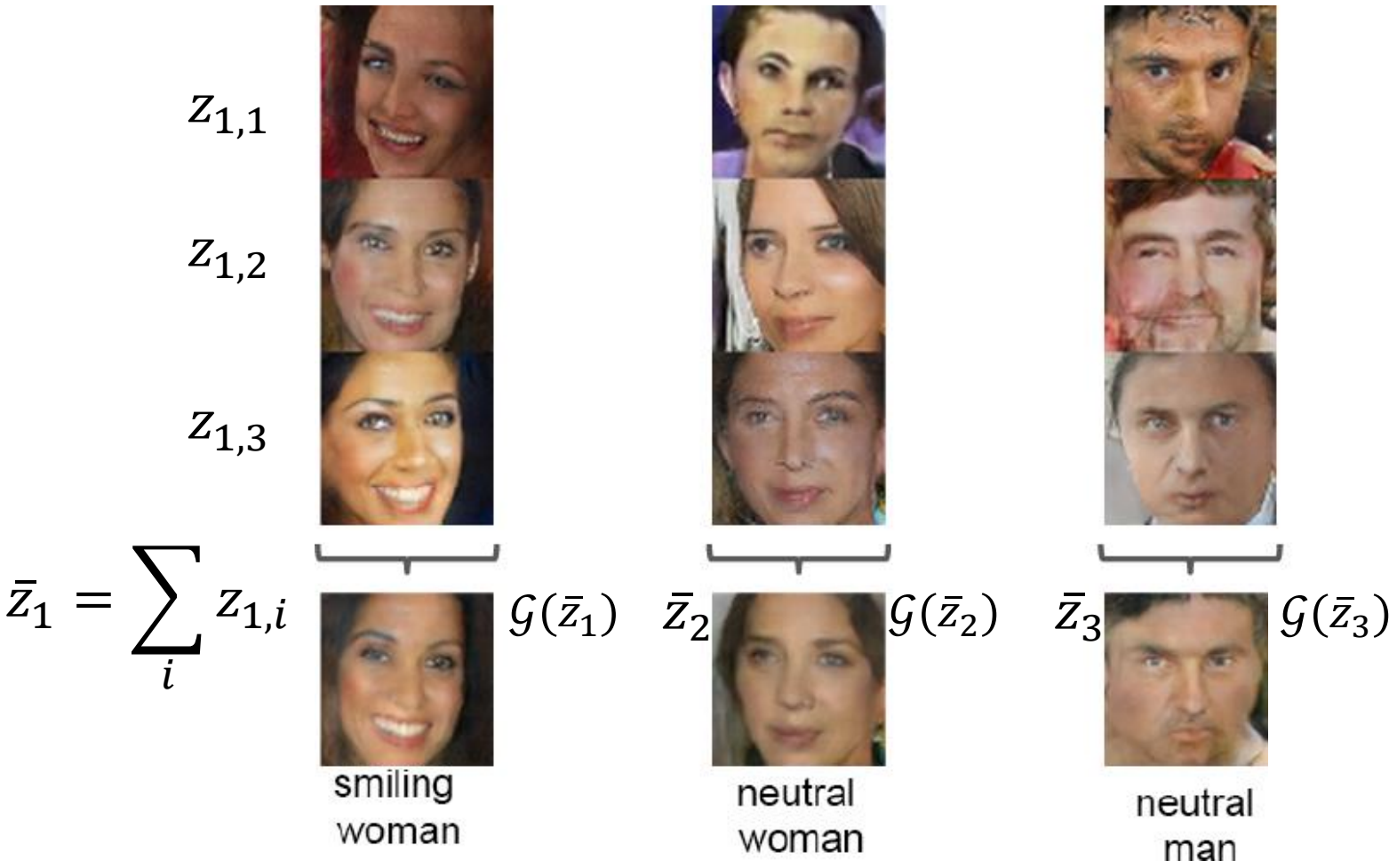
Vector Arithmetic

Take randomly generated samples of smiling women, neutral women, and neutral men

Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

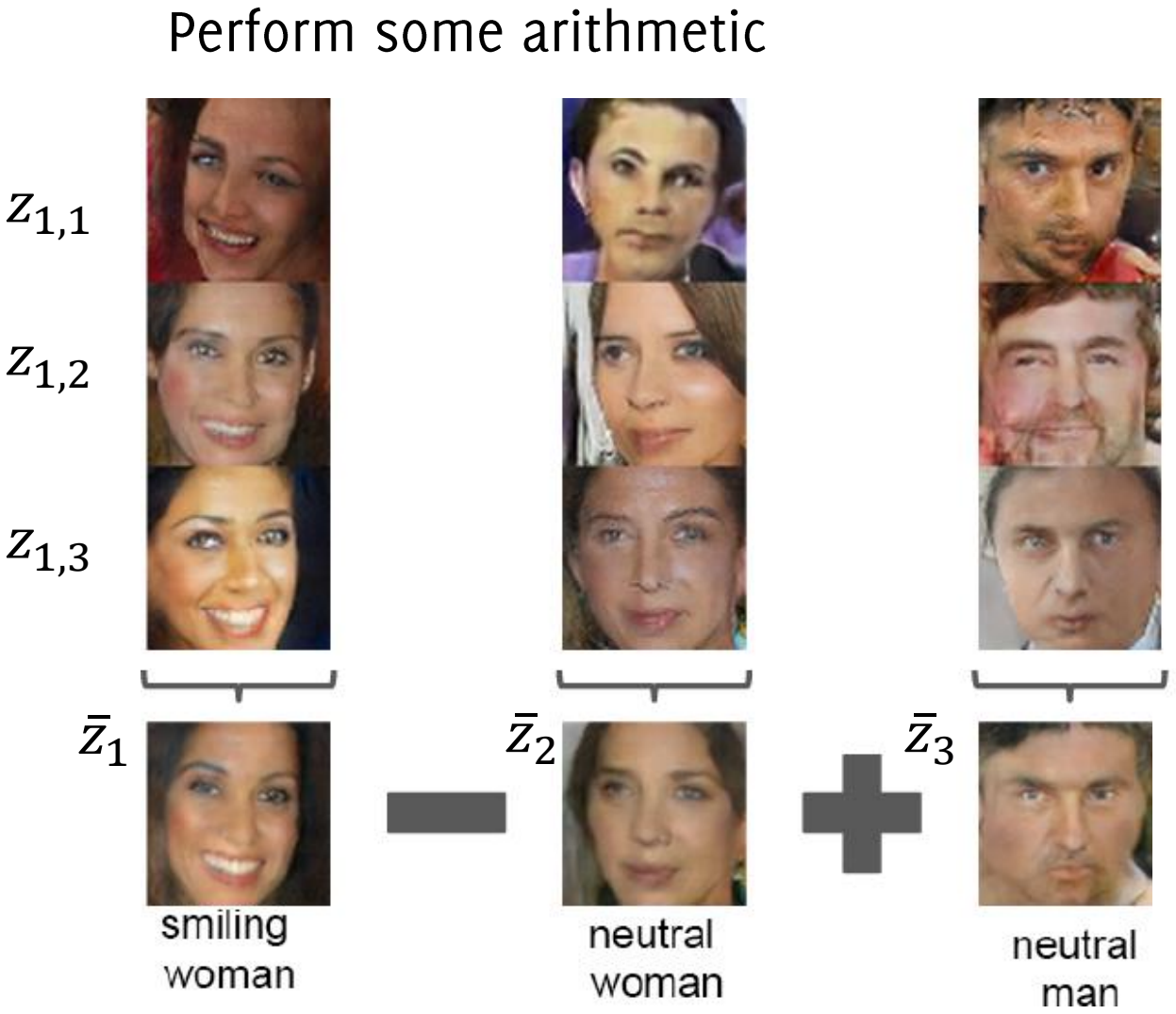
Vector Arithmetic

Average the corresponding noise seeds



Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

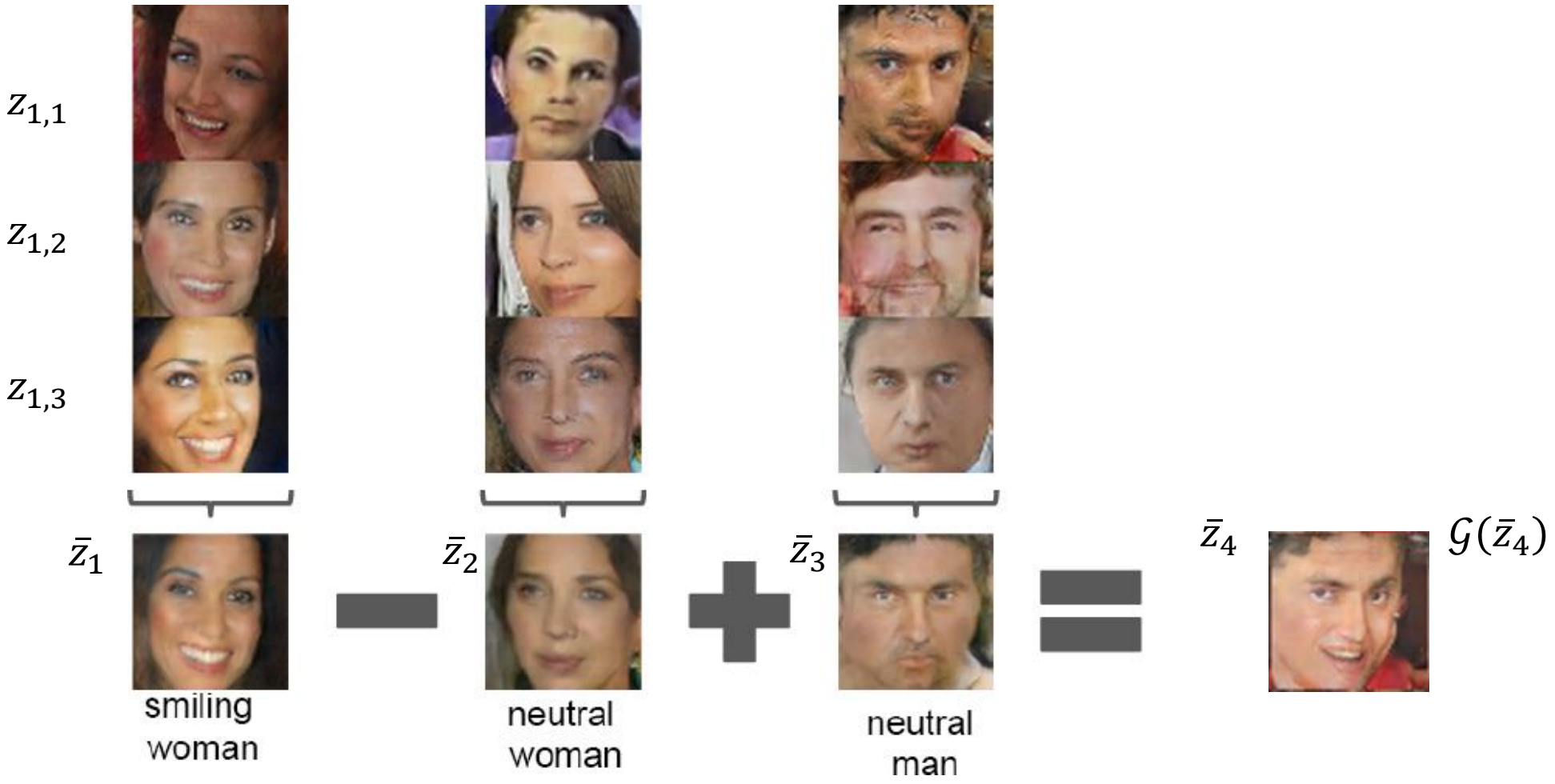
Vector Arithmetic



Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

Vector Arithmetic

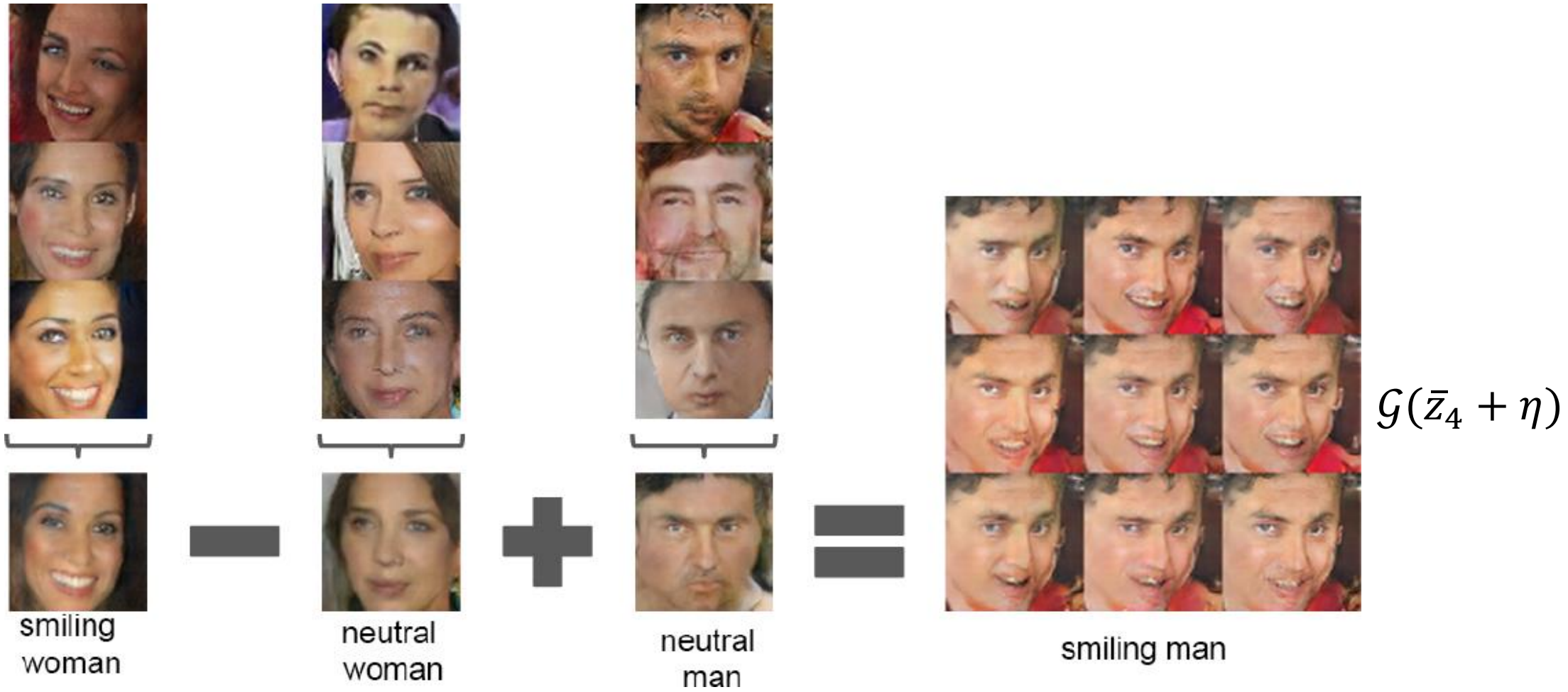
Perform some arithmetic $\bar{z}_1 - \bar{z}_2 + \bar{z}_3 = \bar{z}_4$



Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

Vector Arithmetic

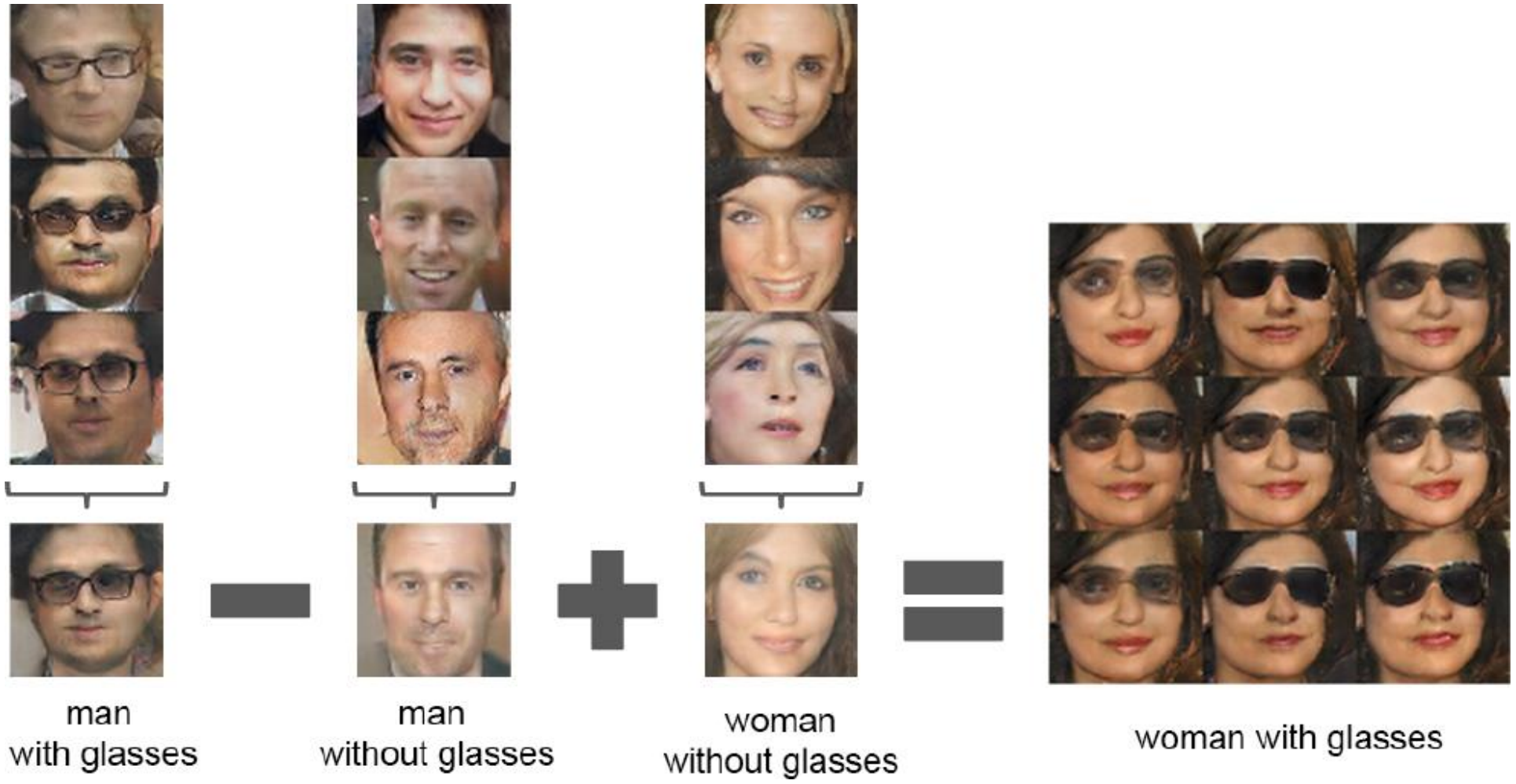
Add some noise to the input vector and that's pretty robust



Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

Vector Arithmetic

Similar example as word embedding



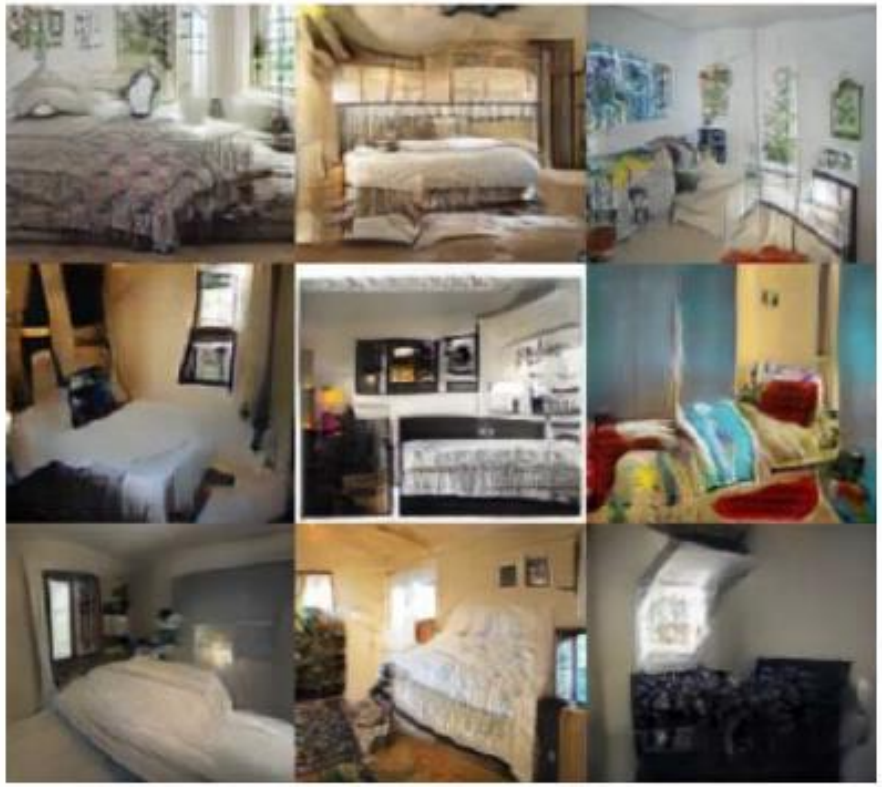
Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

Vector Arithmetic

Interpolation of view changes

Radford, A., Metz, L., & Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. ICLR 2016

GAN has been a very active research field



LSGAN, Zhu 2017.

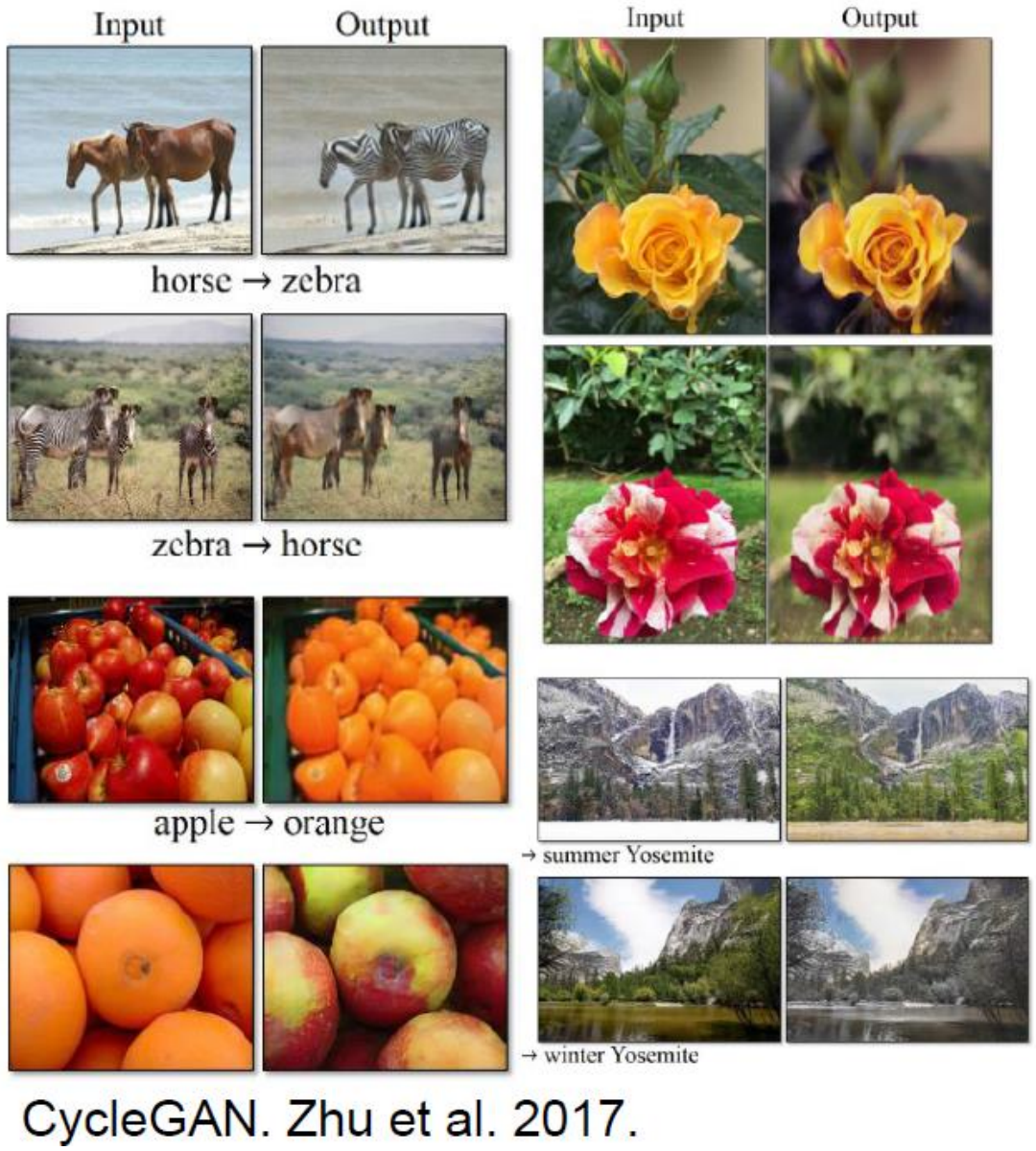
Wasserstein GAN, Arjovsky 2017. Improved Wasserstein GAN, Gulrajani 2017.

GAN has been a very active research field

Karras, Tero, et al. "Progressive growing of gans for improved quality, stability, and variation." *ICLR 2018*

GANs for source-target domain transfer

This can also be used for photo enhancement and data augmentation



CycleGAN. Zhu et al. 2017.

… «The GAN ZOO» and https://github.com/soumith/ganhacks

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

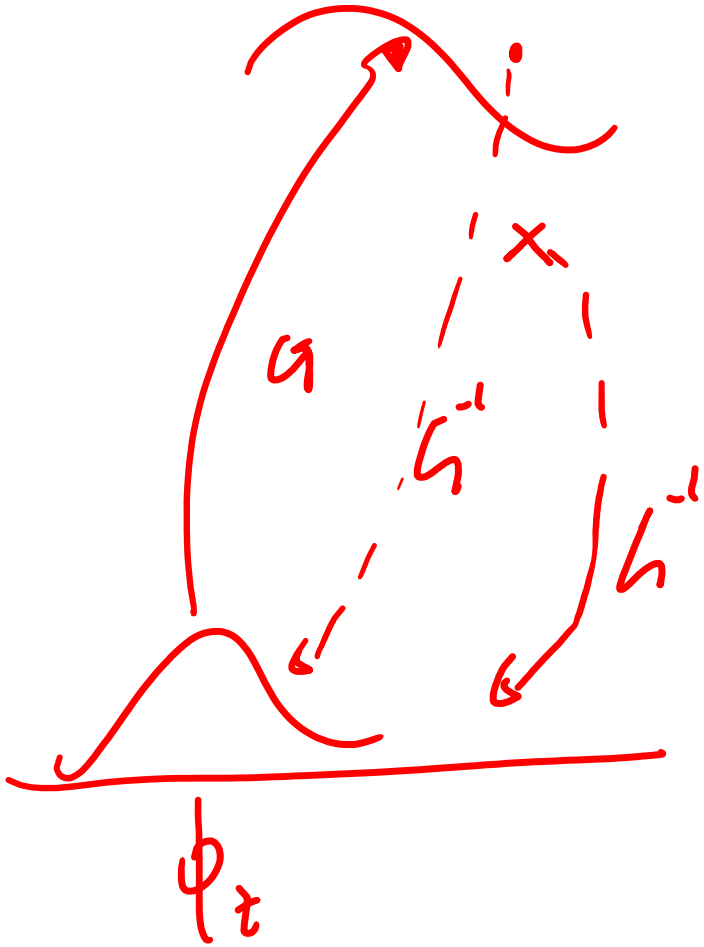
this magnificent fellow is almost all black with a red crest, and white cheek patch.

Reed et al. 2017.

Pix2pix. Isola 2017. Many examples at https://phillipi.github.io/pix2pix/

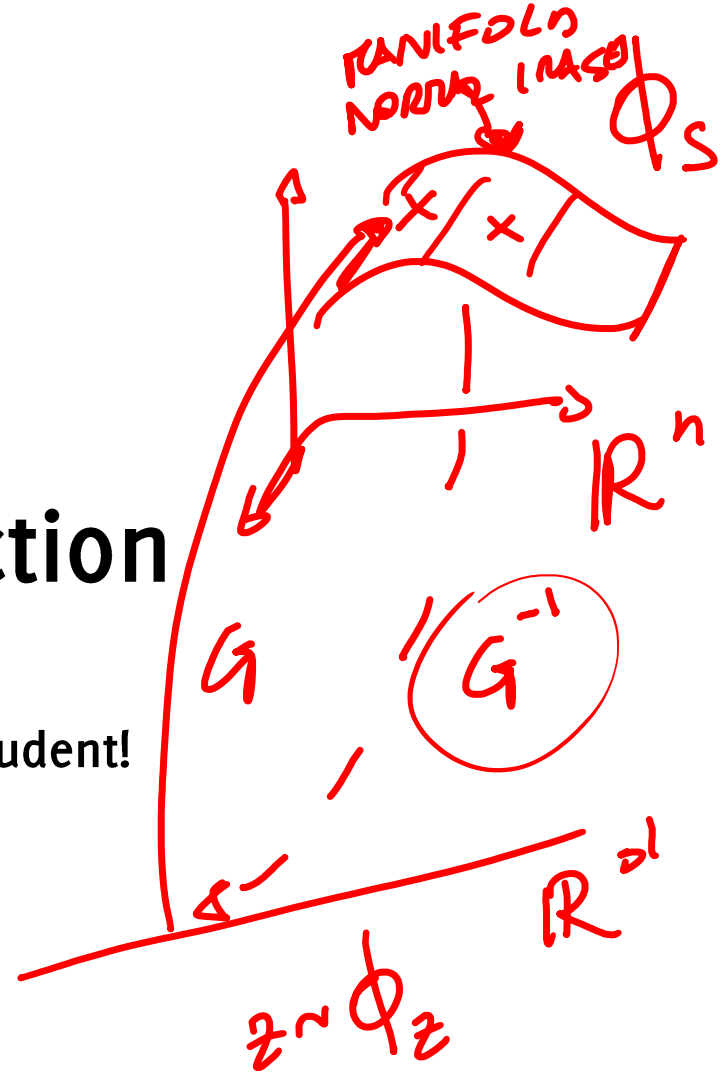
Generative Adversarial Networks (these people do not exist)

Tero Karras, Samuli Laine, Timo Aila «A Style-Based Generator Architecture for Generative Adversarial Networks” **CVPR 2019**



GAN for Anomaly Detection

Thanks to Stefano Pecchia, former AN2DL student!



Anomaly Detection in images

Let s be an image defined over the pixel domain $\mathcal{X} \subset \mathbb{Z}^2$,

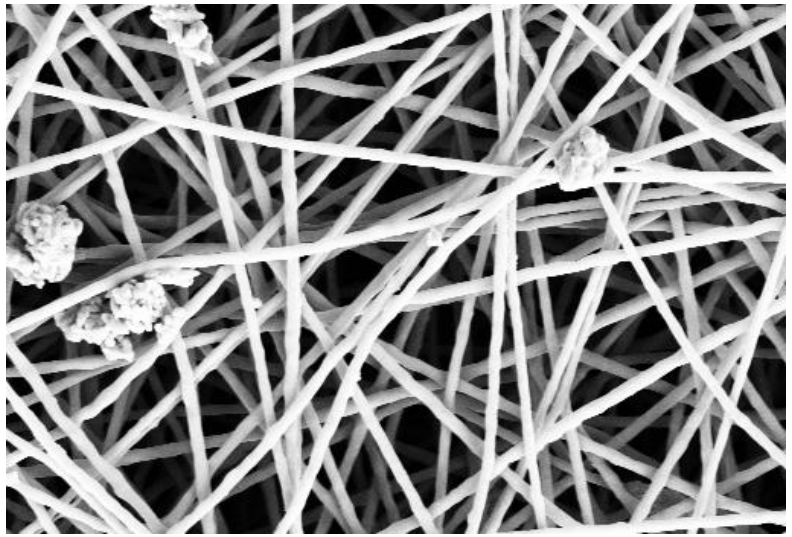
let $c \in \mathcal{X}$ be a pixel and $s(c)$ the corresponding intensity.

Our goal is to **locate any anomalous region** in s, i.e. **estimating the unknown anomaly mask Ω** defined as

TR = { normal images }

$$\Omega(c) = \begin{cases} 0 & \text{if } c \text{ falls in a normal region} \\ 1 & \text{if } c \text{ falls in an anomalous region} \end{cases}$$

s



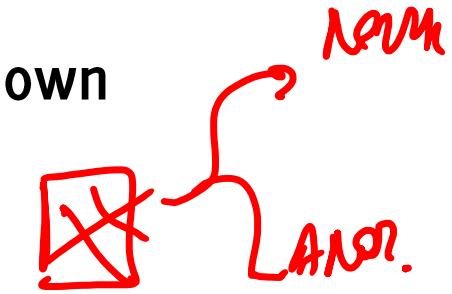
Anomaly Detection in images

Let s be an image defined over the pixel domain $\mathcal{X} \subset \mathbb{Z}^2$,

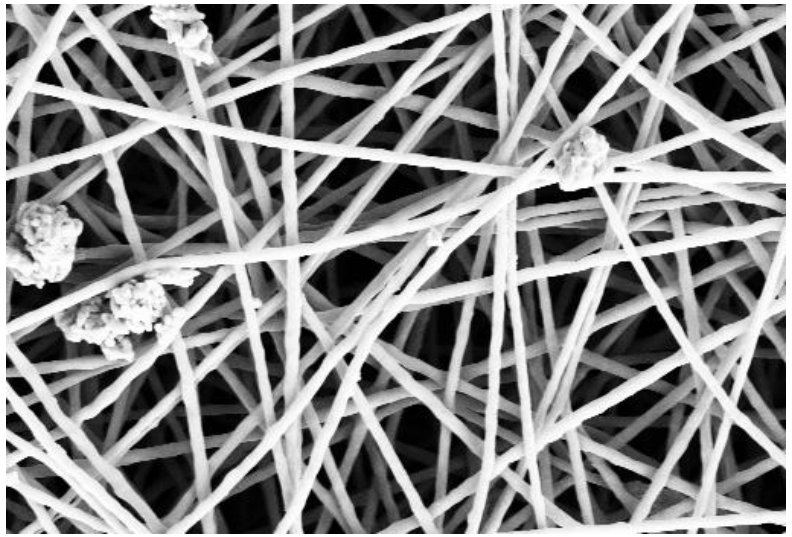
let $c \in \mathcal{X}$ be a pixel and $s(c)$ the corresponding intensity.

Our goal is to **locate any anomalous region** in s, i.e. **estimating the unknown anomaly mask Ω** defined as

$$\Omega(c) = \begin{cases} 0 & \text{if } c \text{ falls in a normal region} \\ 1 & \text{if } c \text{ falls in an anomalous region} \end{cases}$$

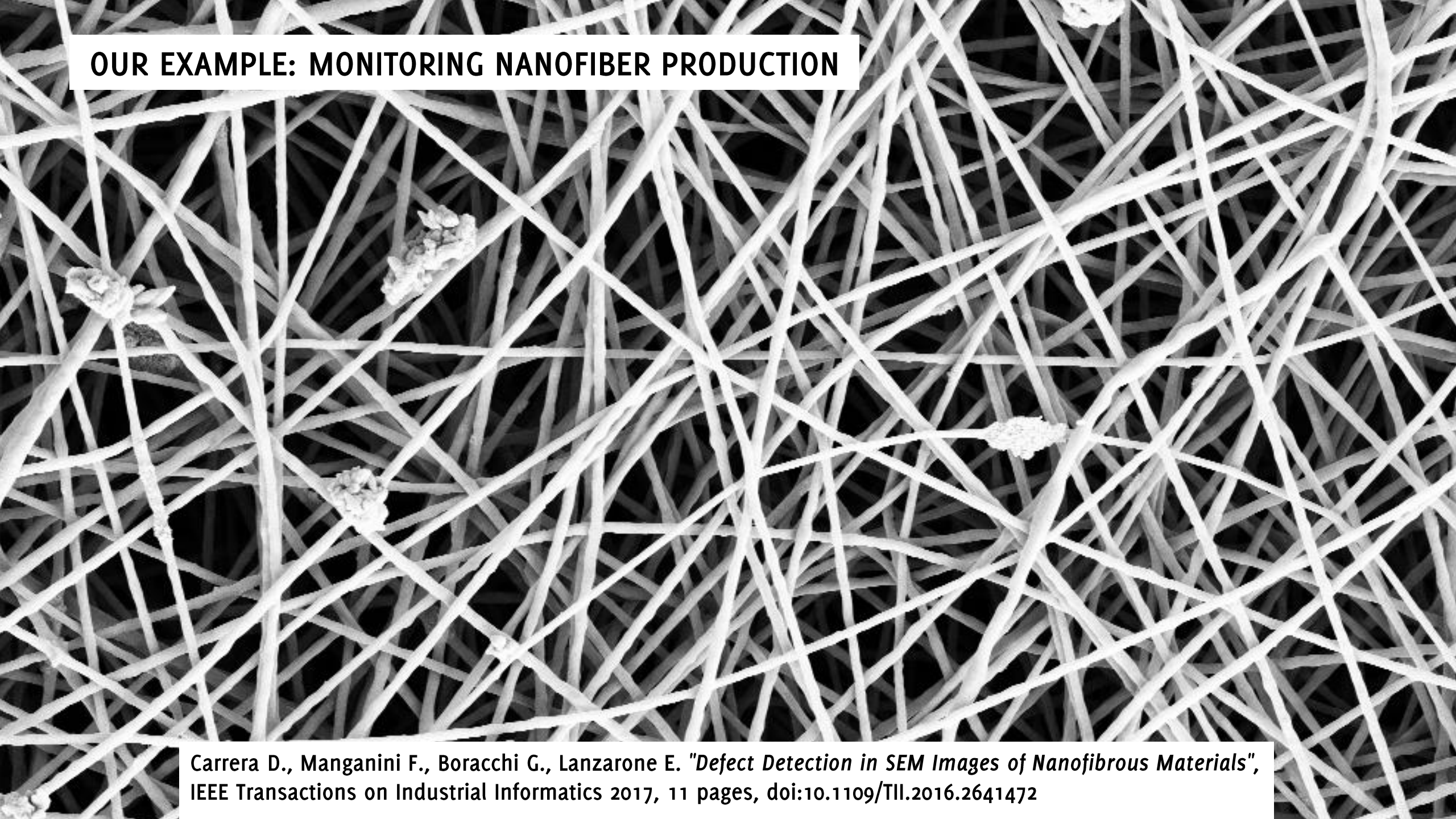


s



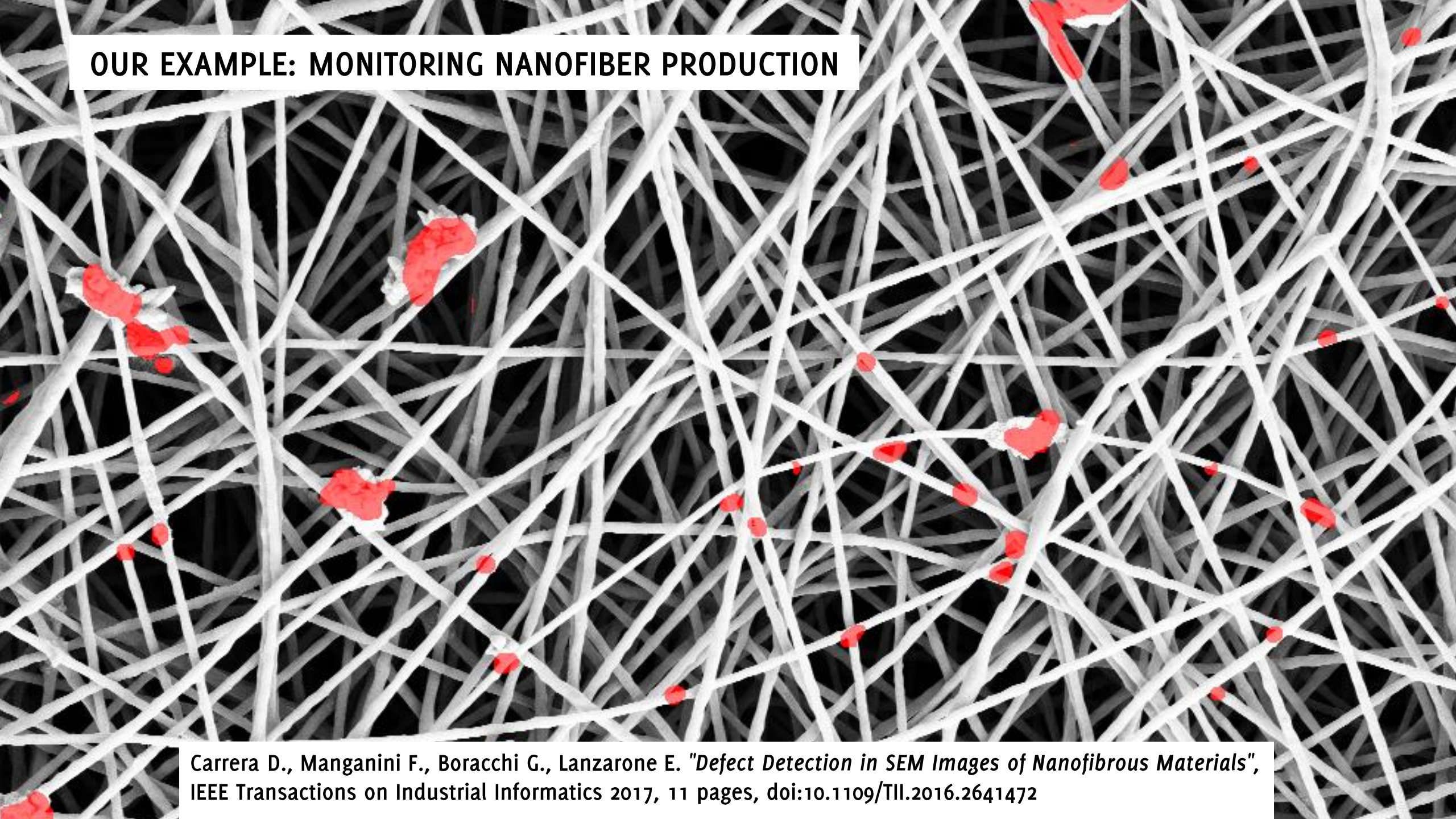
Ω

OUR EXAMPLE: MONITORING NANOFIBER PRODUCTION



Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

OUR EXAMPLE: MONITORING NANOFIBER PRODUCTION



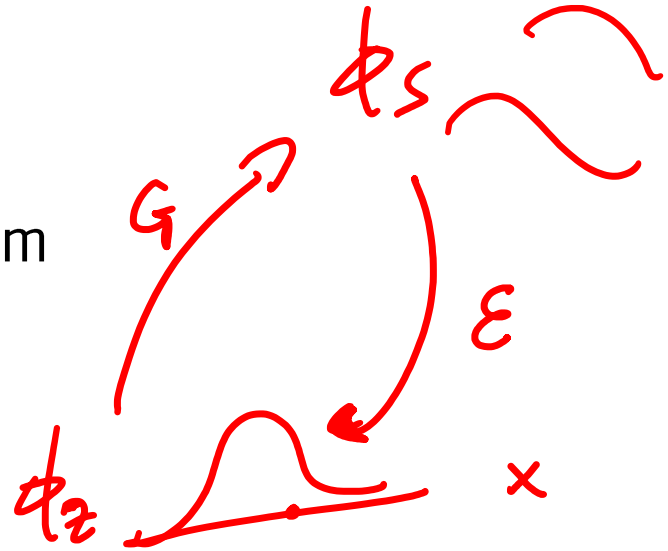
Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

GANs and Anomaly Detection

NORMAL IMAGES ONLY / PATCHES

A Generator $\mathcal{G}$ trained by a GAN provides already a mapping from

- The space of random vectors $z \sim \phi_z$
- The manifold where images live $s \sim \phi_s$



Thus, If we could invert the GAN, we would have already an AD model

Given a test image $s \rightarrow \mathcal{G}^{-1}(s) \rightarrow \phi_z(\mathcal{G}^{-1}(s))$ would be an anomaly score

Unfortunately, it is not possible to invert $\mathcal{G}$… some neural network need to be trained for this purpose!

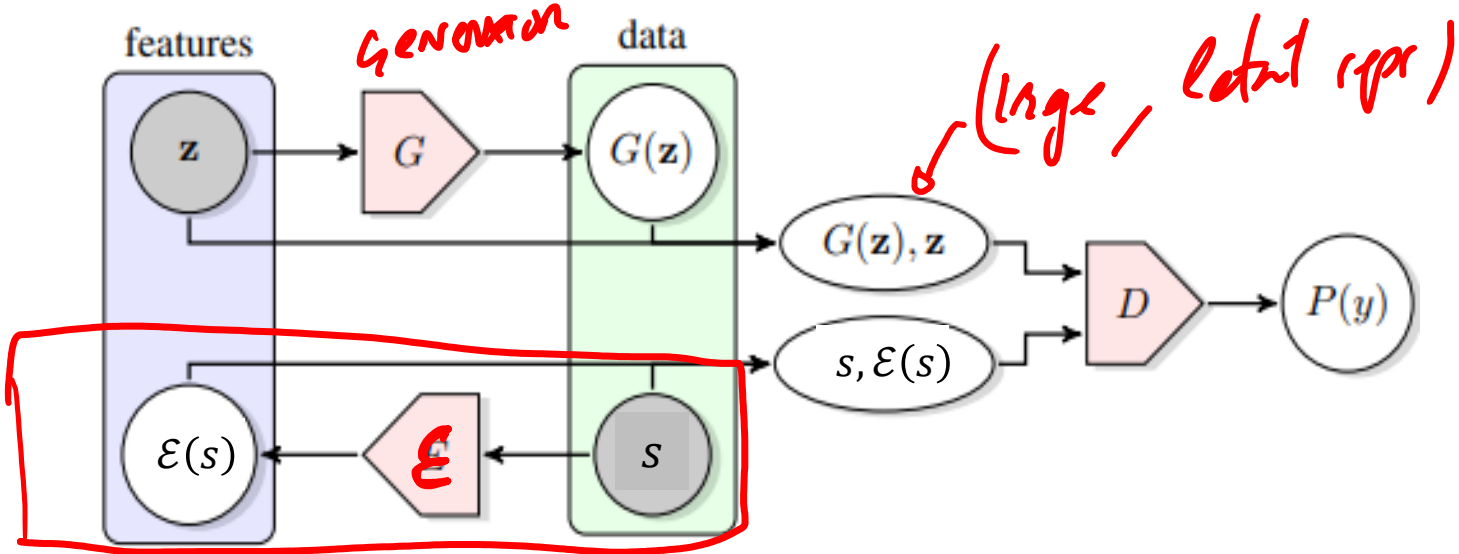
BidirectionalGANs (BiGANs)

The BiGAN adds an encoder E to the adversarial game which

- Brings an image back to the space of "noise vectors"
- Can be used to reconstruct an input image s (as in autoencoders) $\mathcal{G}(\mathcal{E}(s))$
- The discriminator $\mathcal{D}$ takes as input (image, latent repr.) as in conditional GAN

$$\min_{G,E} \max_{\mathcal{D}} V(\mathcal{D}, E, G)$$

$$V(\mathcal{D}, E, G) = \mathrm{E}_{s\sim\phi_S}[\log \mathcal{D}(s, E(s))] + \mathrm{E}_{z\sim\phi_z}[1 - \log \mathcal{D}(G(z), z))]$$



Donahue, J., Krähenbühl, P., & Darrell, T. (2016). Adversarial feature learning. *arXiv preprint arXiv:1605.09782*.

BidirectionalGANs (BiGANs) and Anomaly Detection

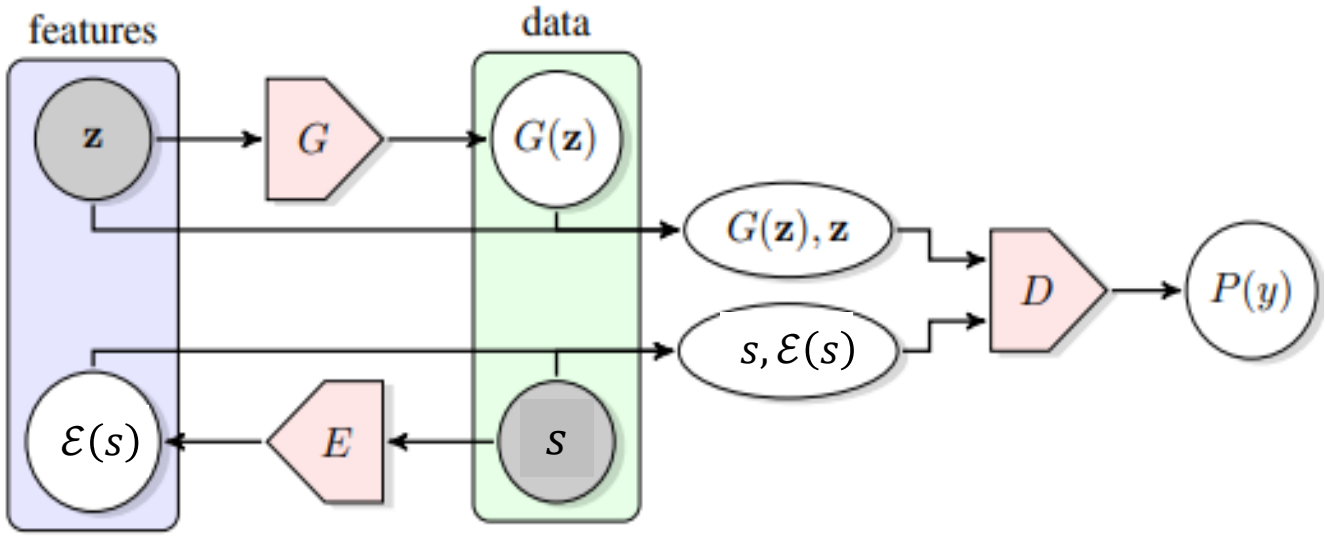
In principle, the encoder $\mathcal{E}(\cdot)$ can be used for anomaly detection by computing the likelihood of $\phi_z(\mathcal{E}(\boldsymbol{s}))$ and consider as anomalous all the images $\boldsymbol{s}$ corresponding to a low likelihood (provided that ϕ_z was not a uniform distribution)

$$\phi_z(\mathcal{E}(\boldsymbol{s}))$$

Another option is to use the posterior of the discriminator as anomaly score

$$\mathcal{D}(\boldsymbol{s}, \mathcal{E}(\boldsymbol{s}))$$

since the discriminator will consider the anomalous sample as fake.



Donahue, J., Krähenbühl, P., & Darrell, T. (2016). Adversarial feature learning. *arXiv preprint arXiv:1605.09782*.

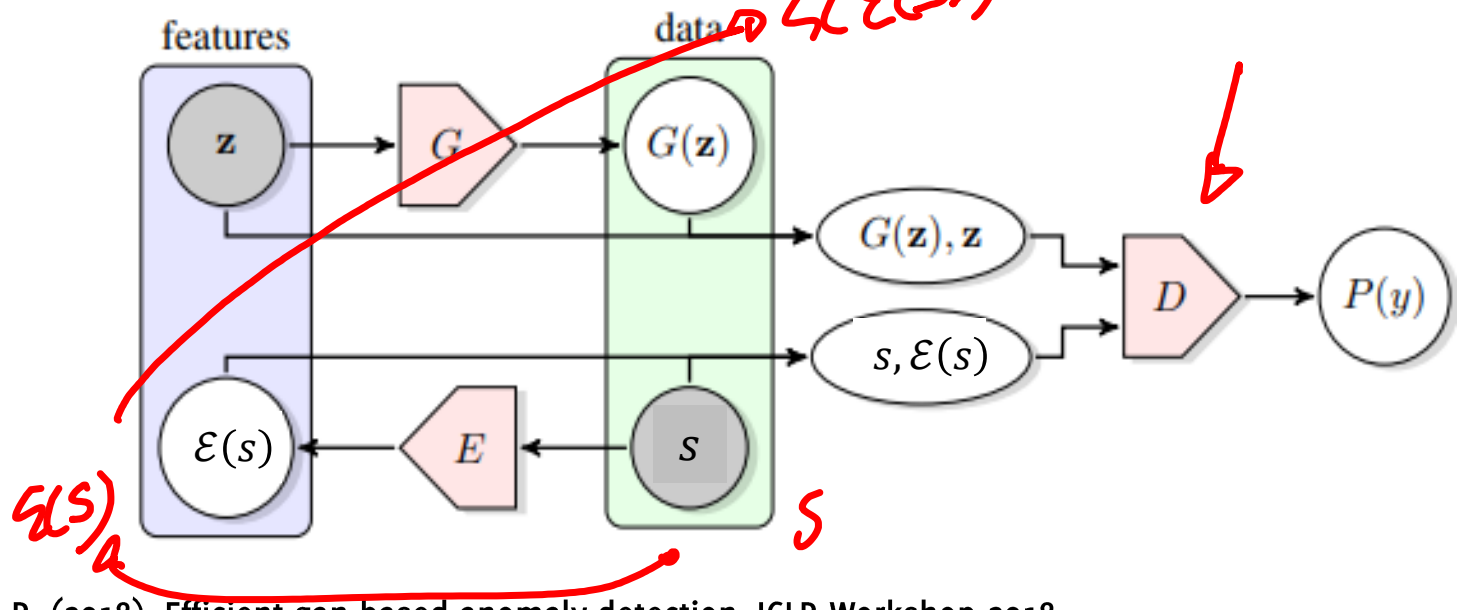
Anomaly detection with BidirectionalGANs (BiGANs)

However, there are more effective anomaly scores

$$A(s) = (1-\alpha)\left|\left|\mathcal{G}(\mathcal{E}(s)) - s\right|\right|_2 + \alpha\left|\left|f\left(\mathcal{D}(s,\mathcal{E}(s))\right) - f\left(\mathcal{D}\left(\mathcal{G}(\mathcal{E}(s)),\mathcal{E}(s)\right)\right)\right|\right|_2$$

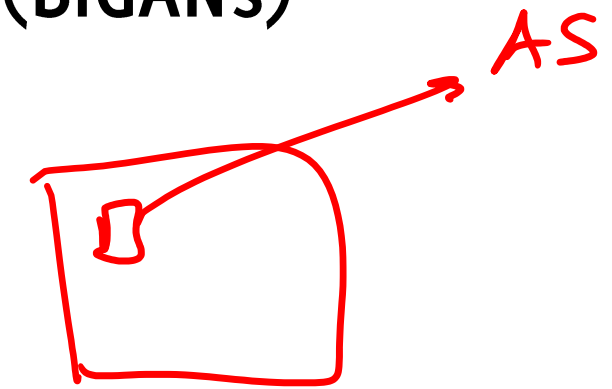
Reconstruction loss

Distance among latent representations of $\mathcal{D}$.
f is the operation of extracting a latent representation



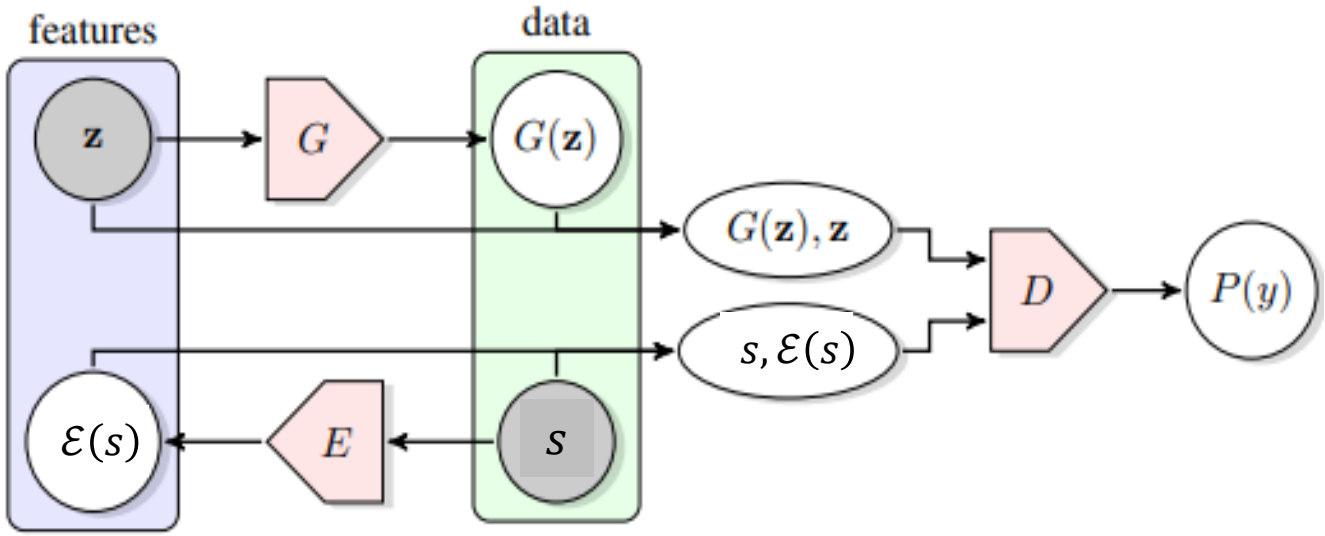
Zenati, H., Foo, C. S., Lecouat, B., Manek, G., & Chandrasekhar, V. R. (2018). Efficient gan-based anomaly detection. ICLR Workshop 2018

Anomaly detection with BidirectionalGANs (BiGANs)



Limitations

- Image-wise / patch-wise training and testing
- Little stability during training
- No way to promote better quality of reconstructed images

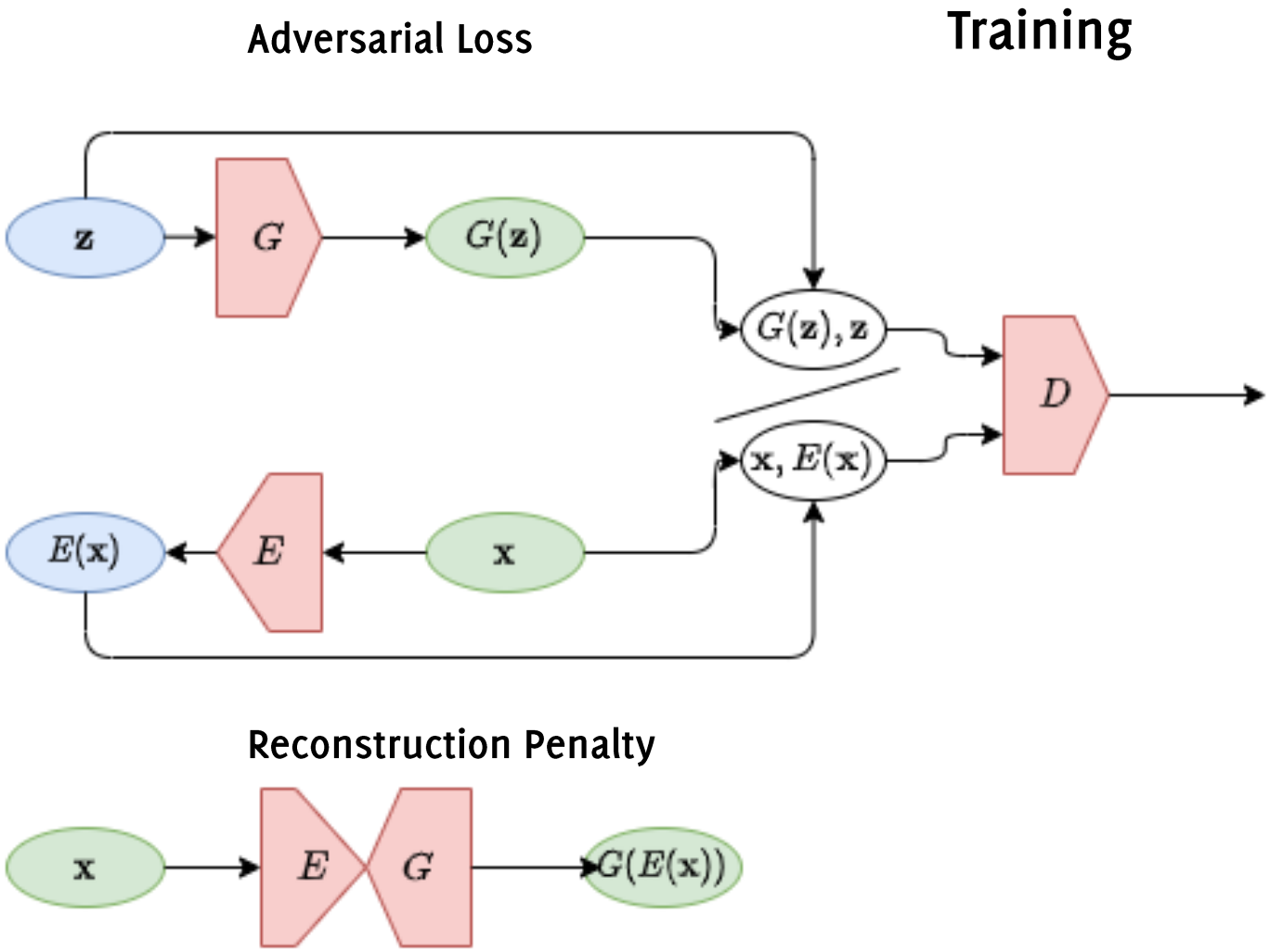


Zenati, H., Foo, C. S., Lecouat, B., Manek, G., & Chandrasekhar, V. R. (2018). Efficient gan-based anomaly detection. ICLR Workshop 2018

Fully Convolutional Anomaly Detection by GANs

Training

- All the layers are made fully convolutional (much more efficient processing)
- LS-losses used to train the model (this improves stability)



Credits Stefano Pecchia

Fully Convolutional Anomaly Detection by GANs

Inference

- **Anomaly score:** combines

I. image reconstruction error

$$||\mathcal{G}(\mathcal{E}(\boldsymbol{s})) - \boldsymbol{s}|\Big|_{\mathbf{2}}$$

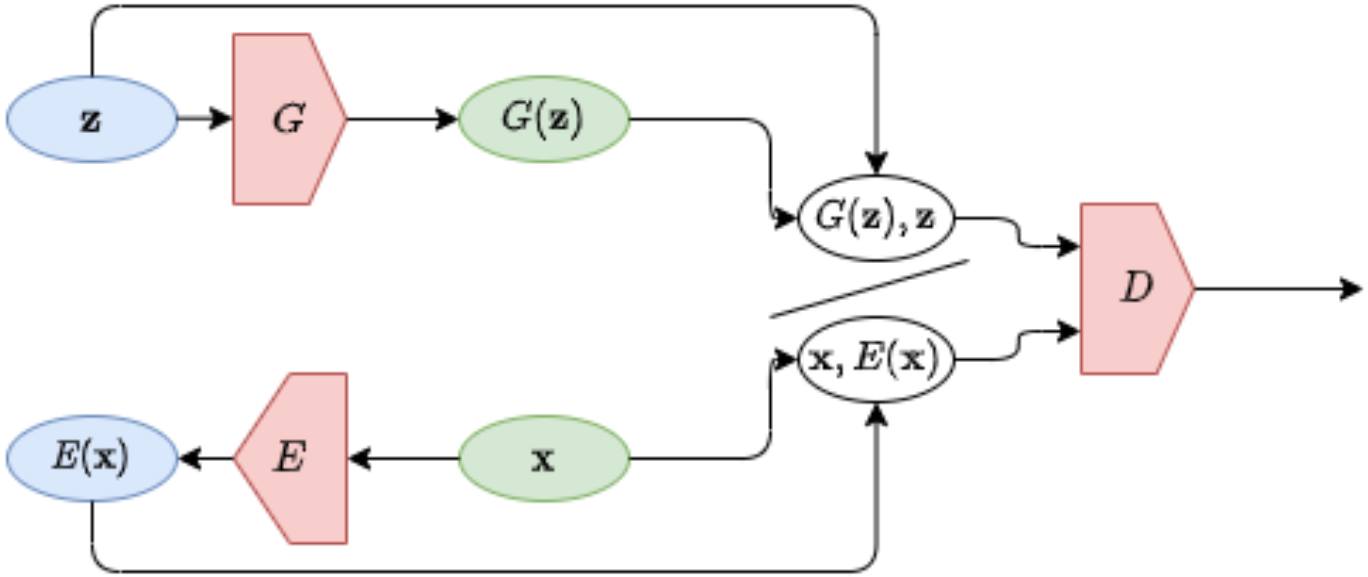
II. discriminator loss

$$\left(\mathcal{D}(s, \mathcal{E}(s)) - 1\right)^2$$

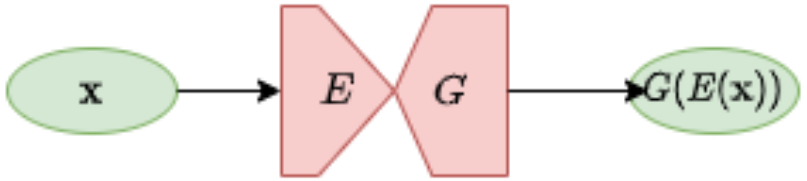
(since normal images are assumed to return 1)

- Possibly also likelihood w.r.t estimated distribution of $\mathcal{E}(s)$

Adversarial Loss

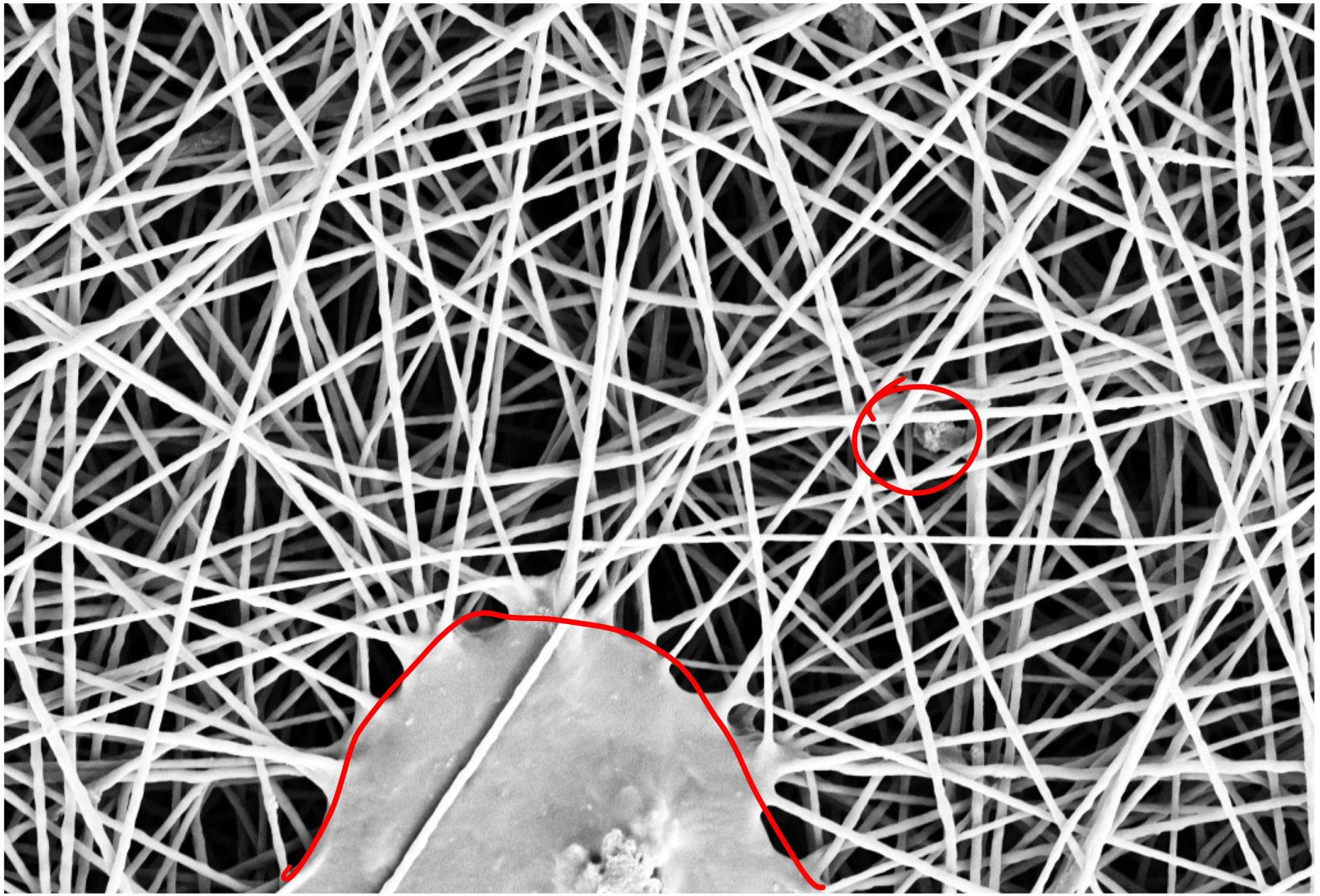


Reconstruction Penalty



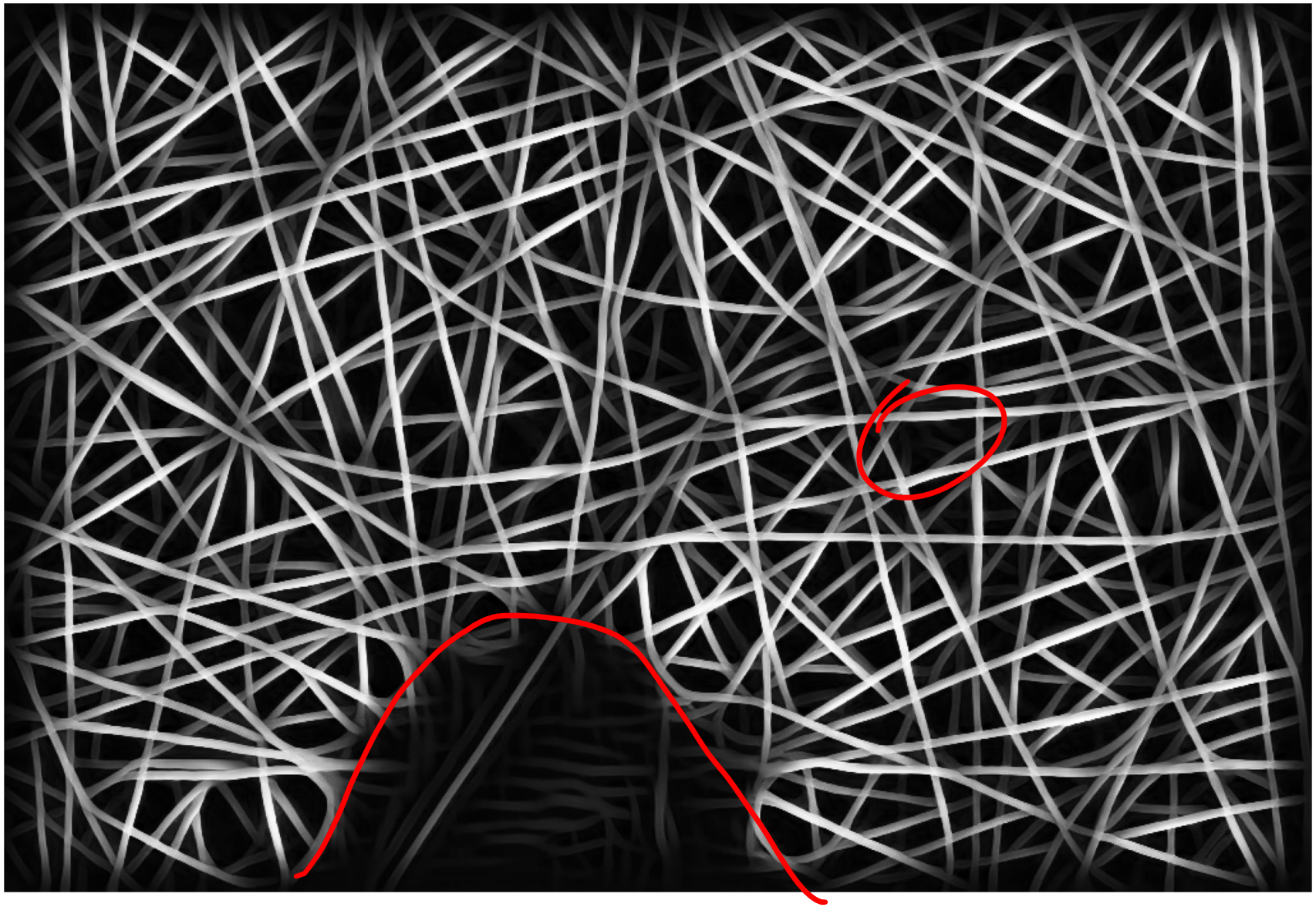
Credits Stefano Pecchia

Input Image



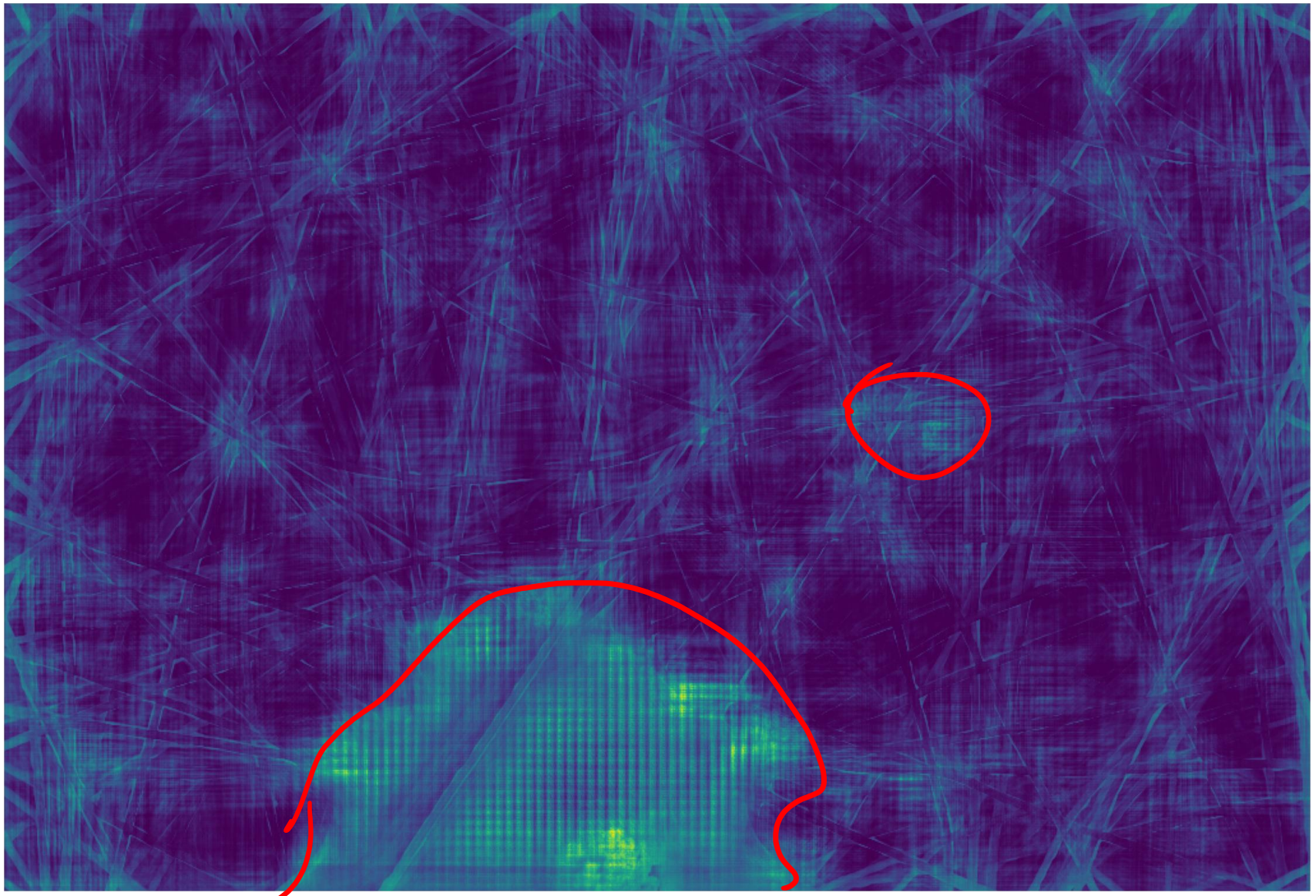
Credits Stefano Pecchia

Reconstruction $\mathcal{G}(\mathcal{E}(\boldsymbol{s}))$



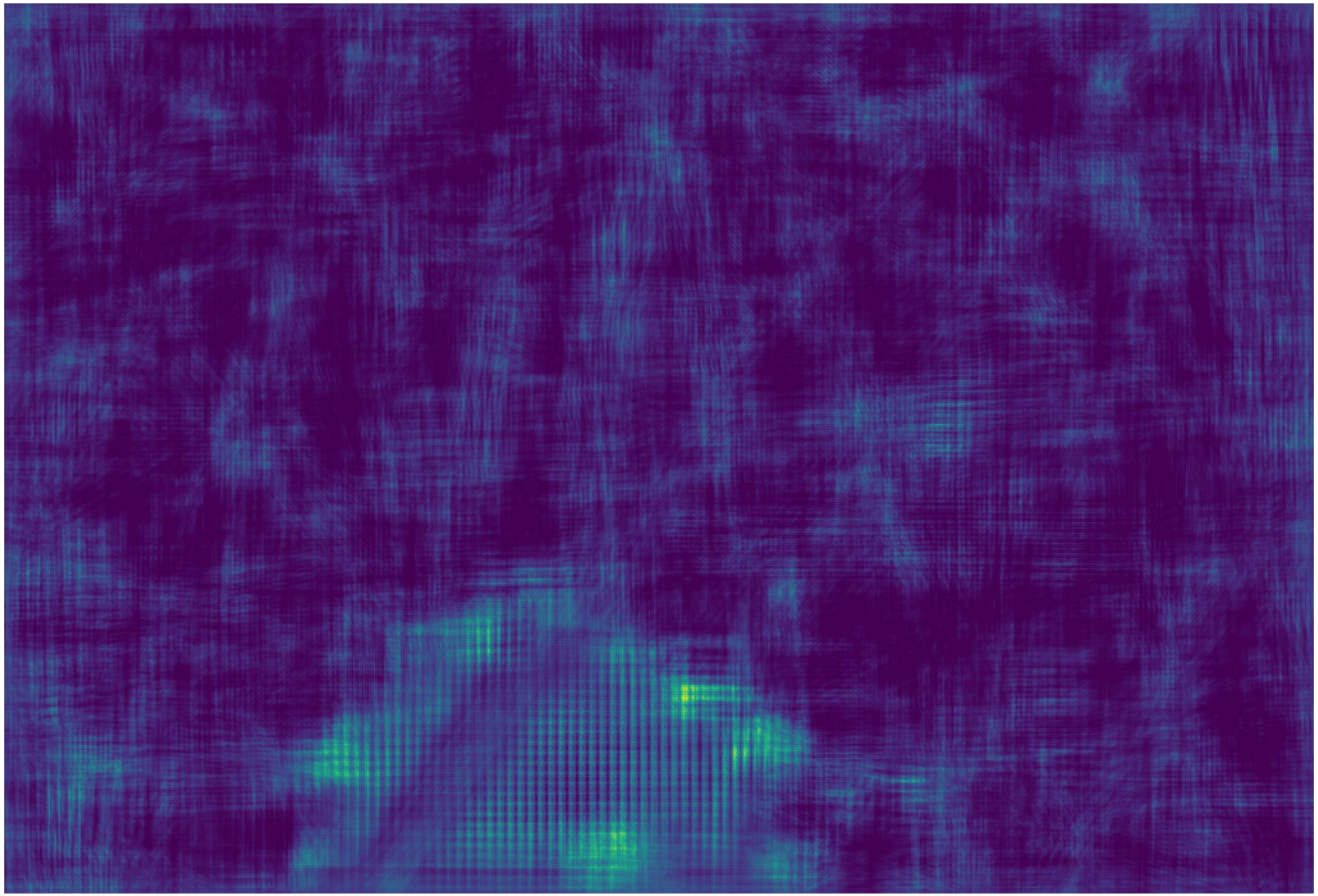
Credits Stefano Pecchia

Reconstruction Loss $||\mathcal{G}(\mathcal{E}(\boldsymbol{s})) - \boldsymbol{s}||_2$



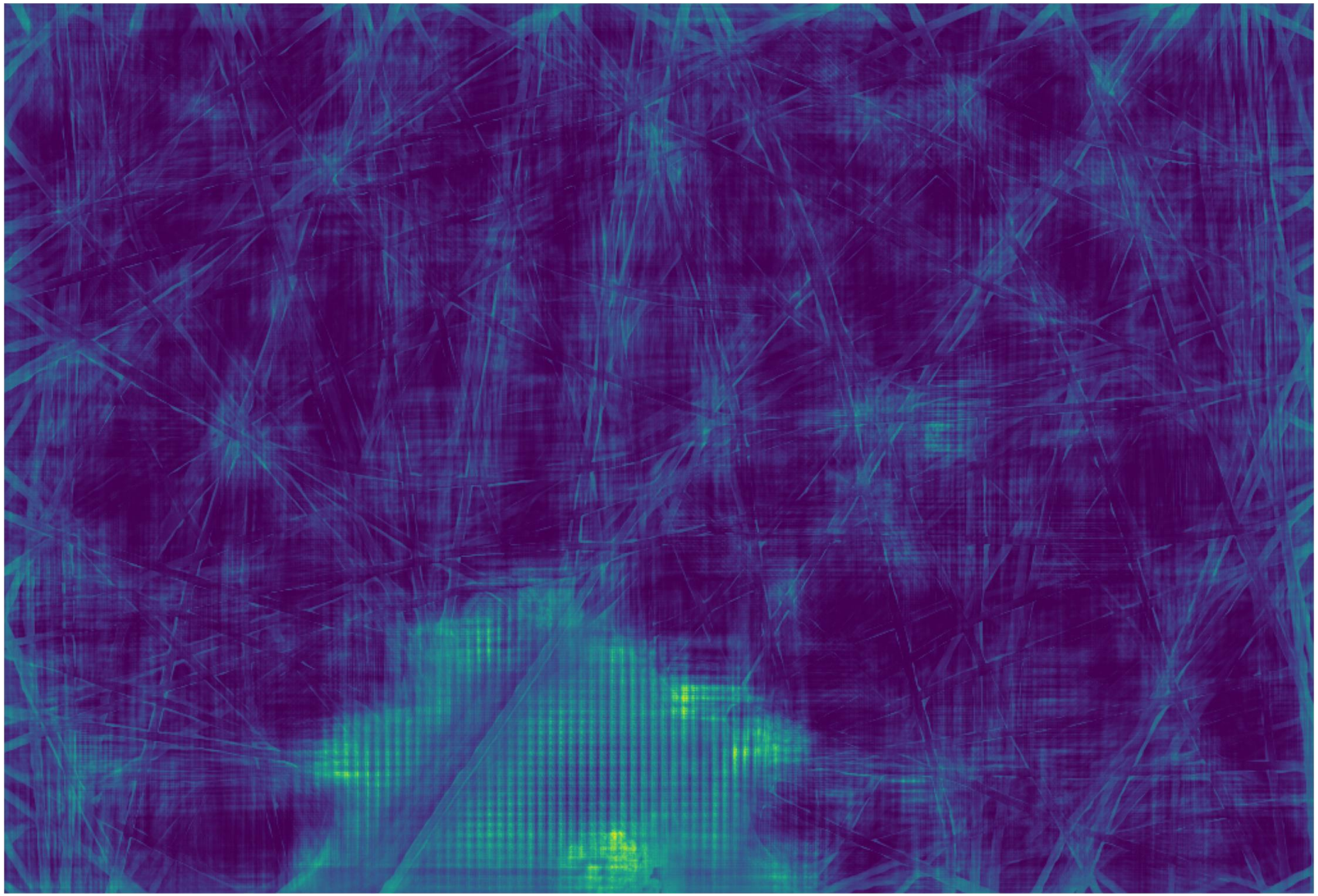
Credits Stefano Pecchia

Discriminator Score $\left(\mathcal{D}\left(s, \mathcal{E}(s)\right) - 1\right)^2$



Credits Stefano Pecchia

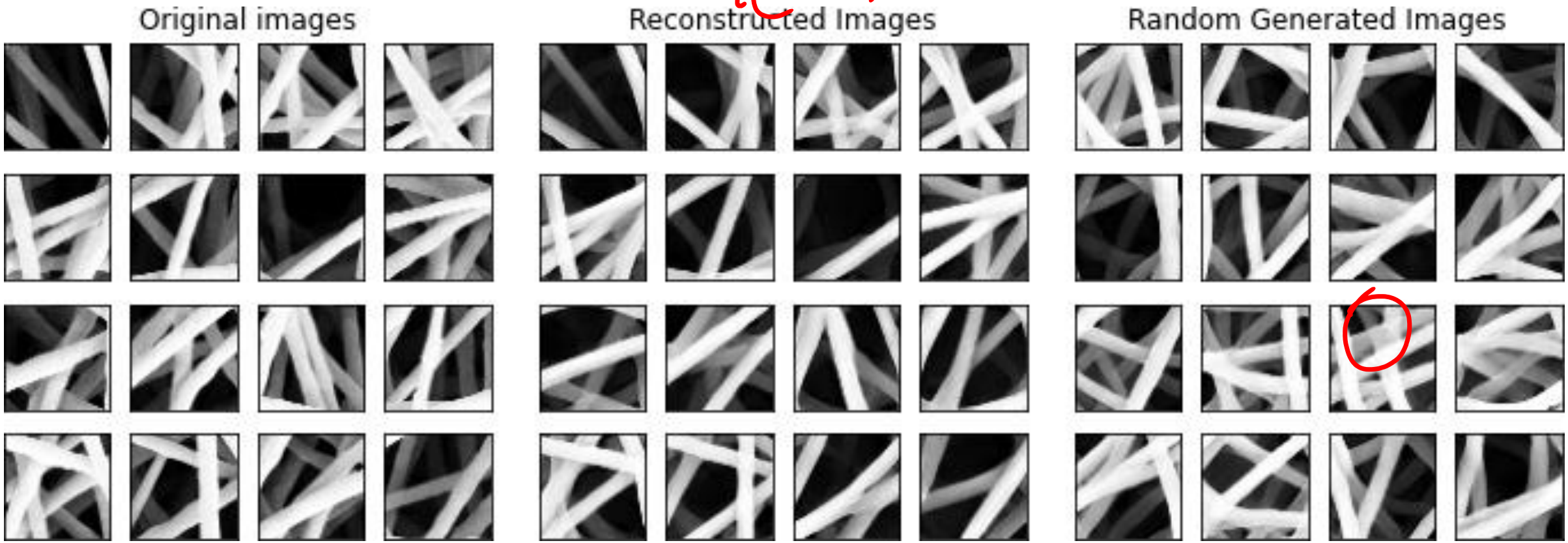
Anomaly Score $\alpha\left(\mathcal{D}\left(s, \mathcal{E}(s)\right) - 1\right)^2 + (1-\alpha)||\mathcal{G}\left(\mathcal{E}(\boldsymbol{s})\right) - \boldsymbol{s}||_{\boldsymbol{2}}$



Credits Stefano Pecchia

Normal Image Generation By Our GAN

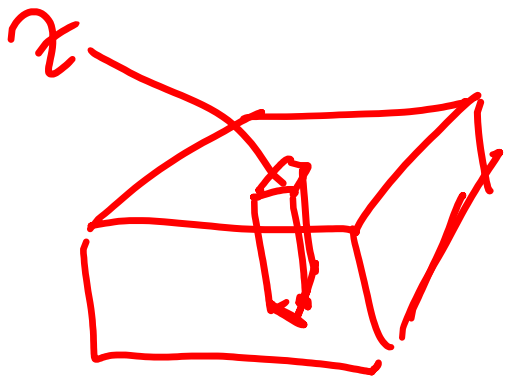
$z \in \mathbb{R}^{128}$

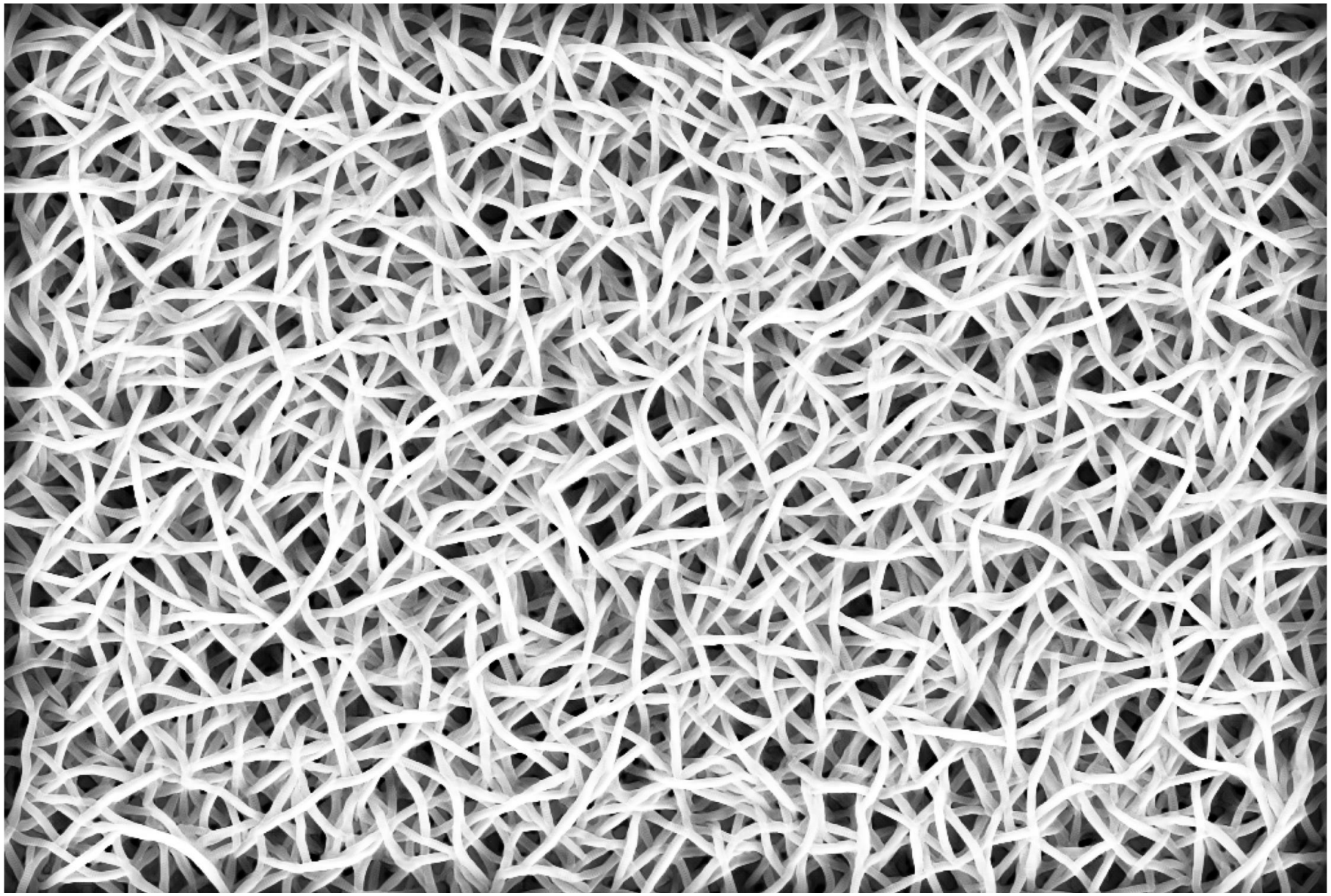


Credits Stefano Pecchia Thesis

It is a Genertive model!

$z \in \mathbb{R}^{80 \times 120 \times 128}$





Local regions are well connected, but the GAN do not enforce a global image structure

Credits Stefano Pecchia

Mathematical Models and Methods for Image Processing

Spring 2022, for Mathematical Engineering

What is this course about?

What is this course about?

*It is about **algorithms** for processing **images** and solving image-related problems.*

Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K. *"Image denoising by sparse 3-D transform-domain collaborative filtering"* TIP 2007

What is this course about?

*It is about **algorithms** for processing **images** and solving image-related problems.*

..like denoising

Dabov, K., Foi, A., Katkovnik, V., Egiazarian, K. *"Image denoising by sparse 3-D transform-domain collaborative filtering"* TIP 2007

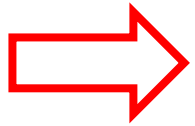
Who cares about images?

Who cares about images?

Everybody!

We will see algorithms solving problems customarily addressed in our phones,

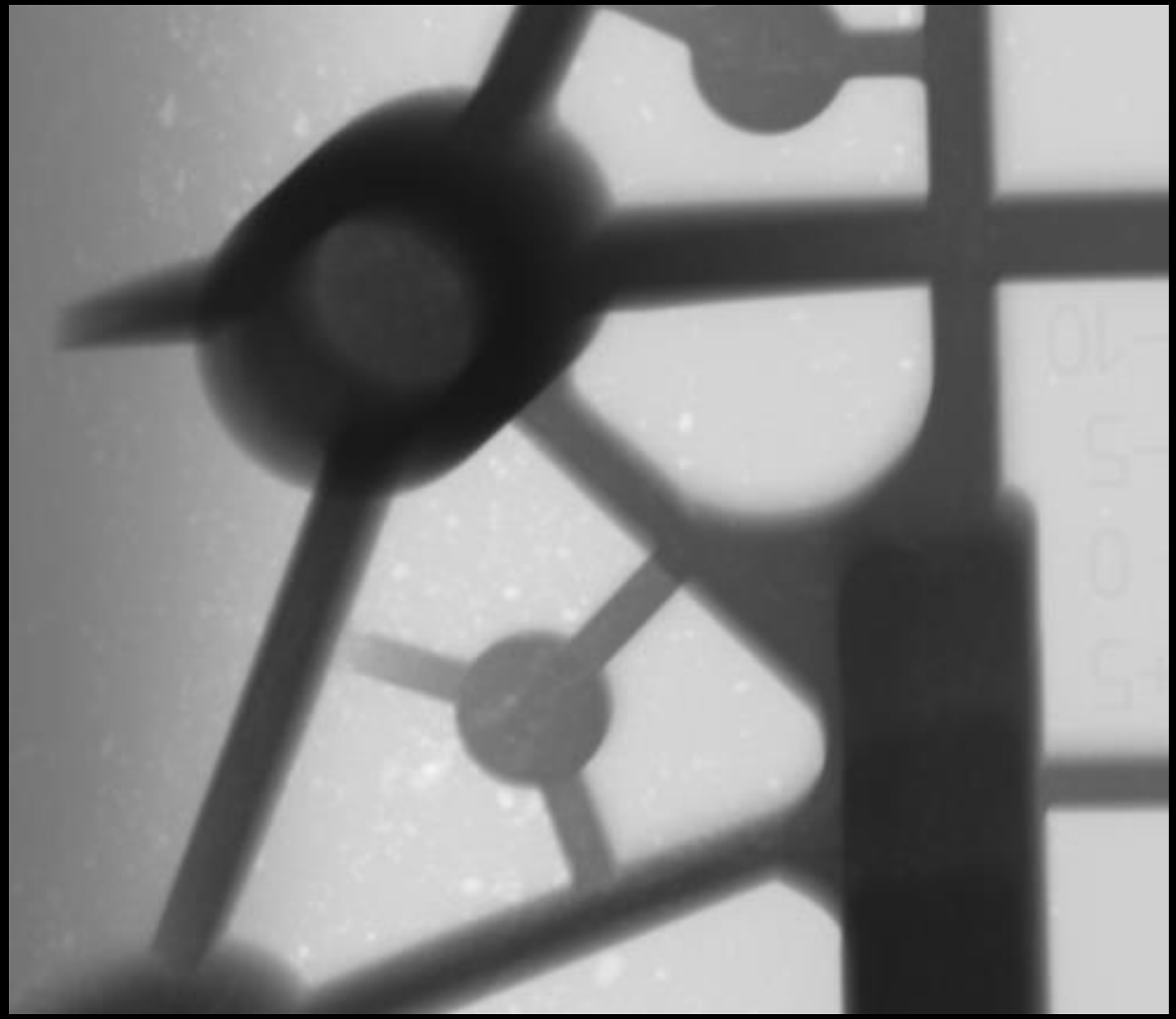
$$z = y + \eta\,, \qquad \eta \sim \mathcal{N}(0, \sigma^2)$$



$$\hat{y} \approx y$$

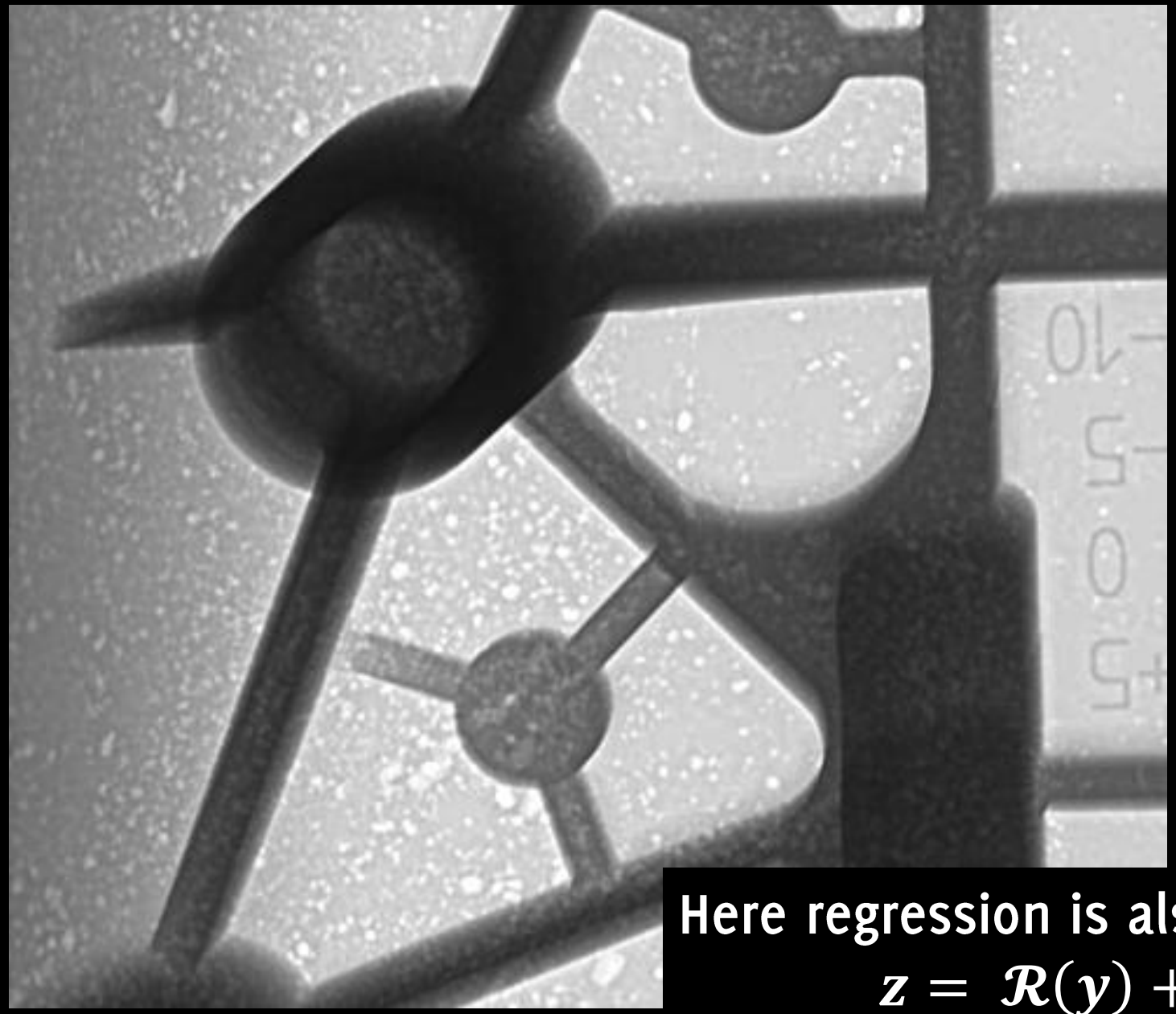
Denoising is a regression problem: given the noisy z, estimate $\hat{y}$ close to the unknown y

Who cares about images? Quality Inspection



G. Boracchi

Who cares about images? Quality Inspection

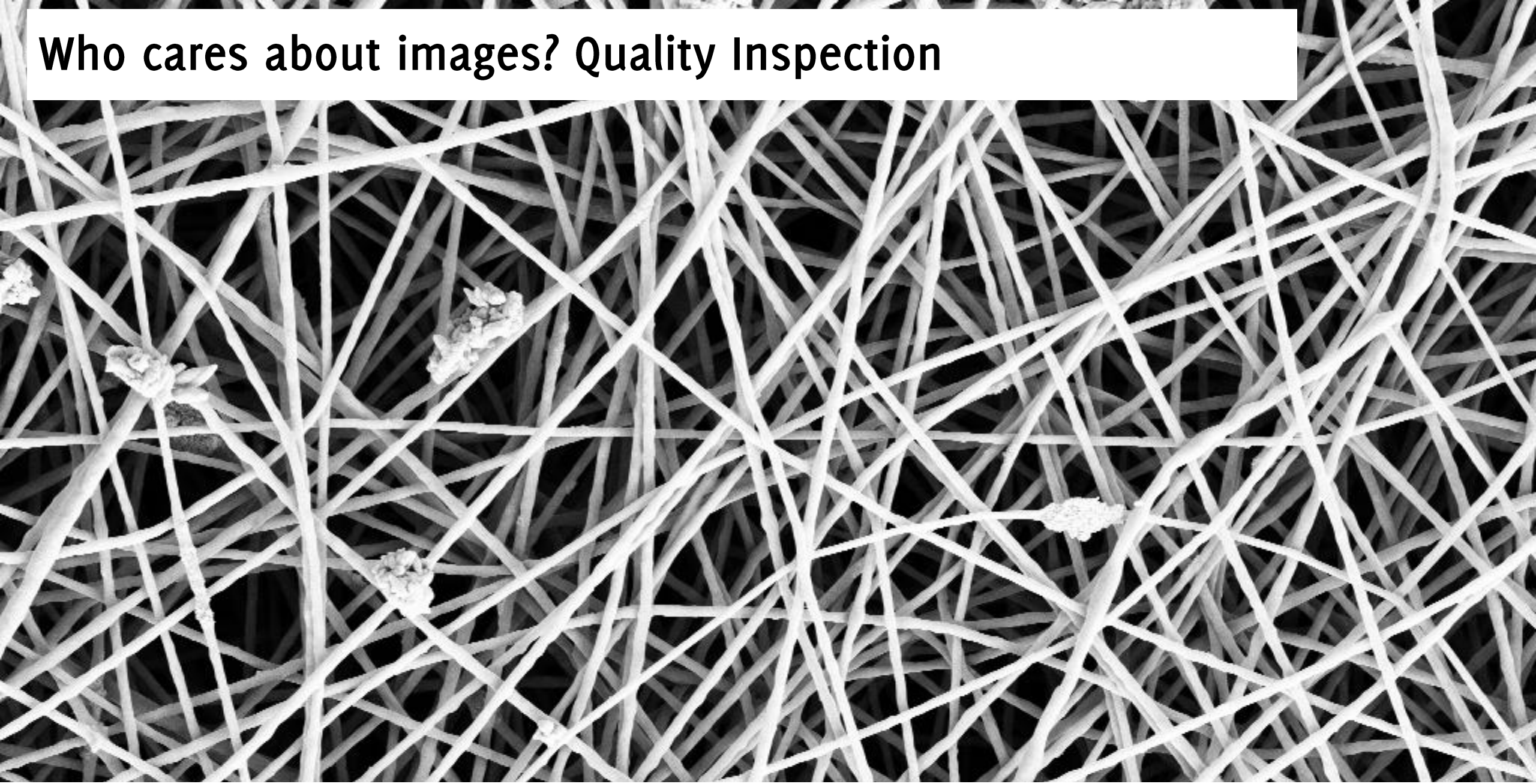


Here regression is also crucial

$$z = \mathcal{R}(y) + \eta$$

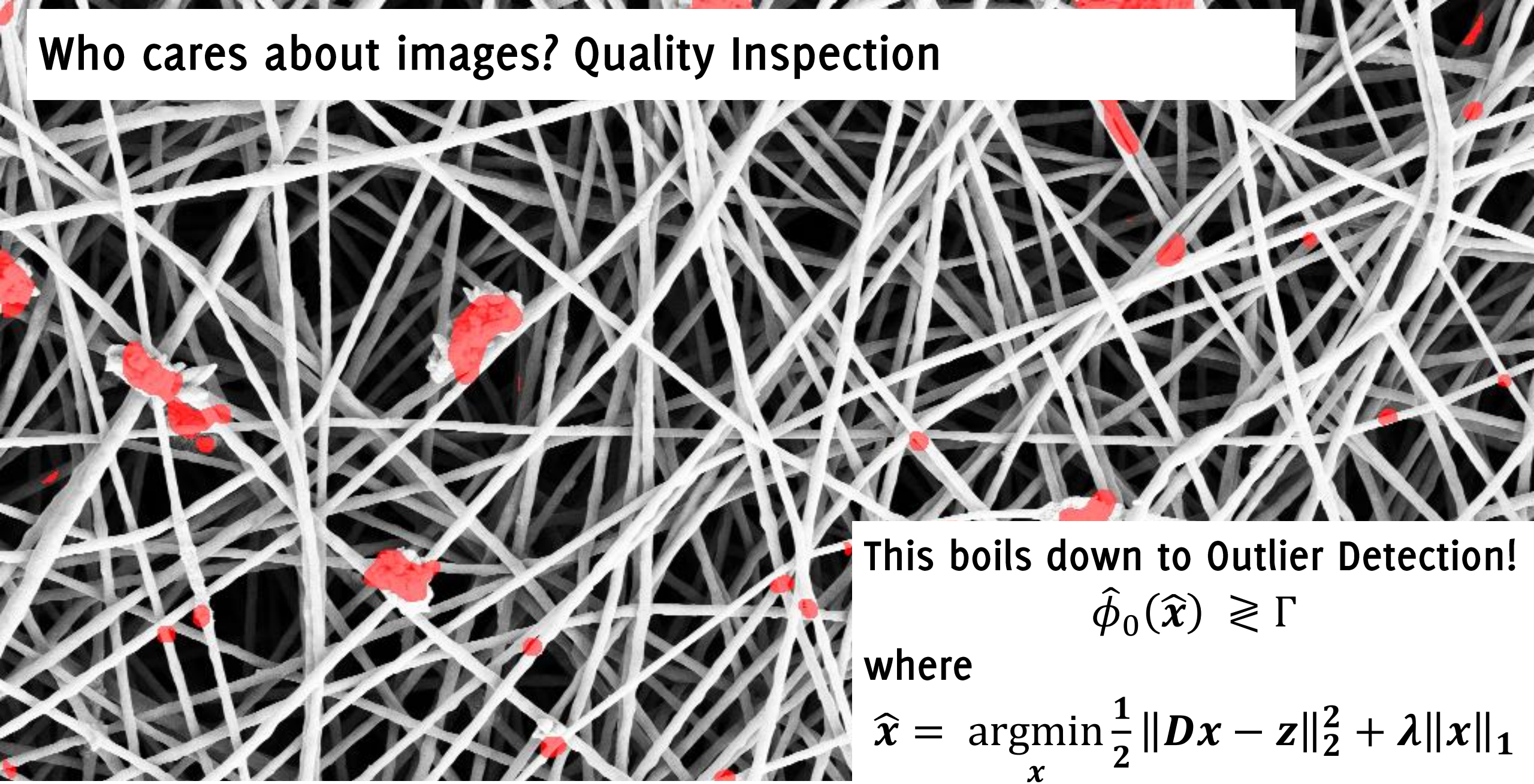
G. Boracchi

Who cares about images? Quality Inspection



Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

Who cares about images? Quality Inspection



This boils down to Outlier Detection!

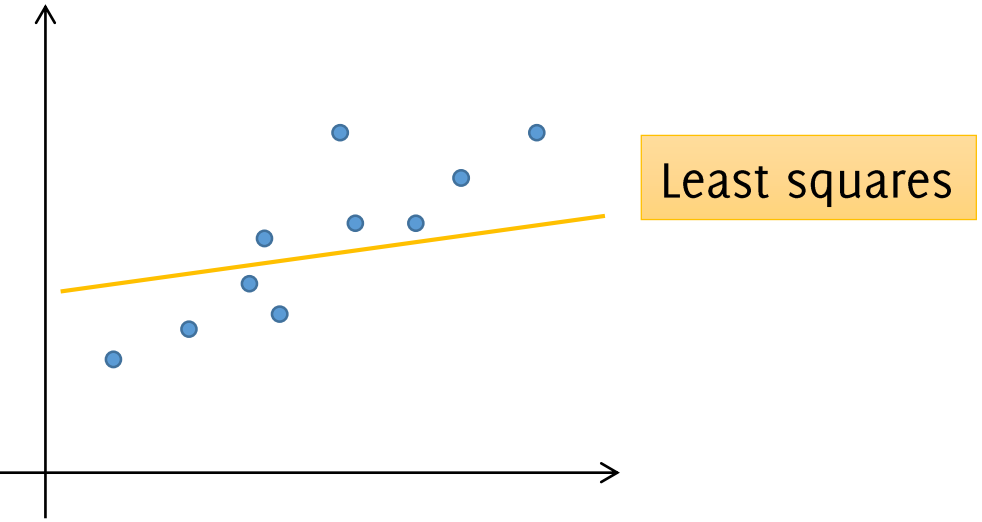
$$\hat{\phi}_0(\hat{\boldsymbol{x}}) \gtrless \Gamma$$

where

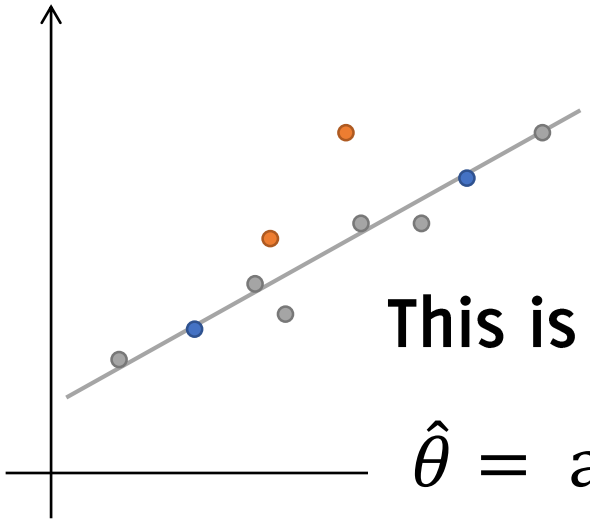
$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \frac{1}{2} \|\boldsymbol{D}\boldsymbol{x} - \boldsymbol{z}\|_2^2 + \boldsymbol{\lambda}\|\boldsymbol{x}\|_1$$

Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

Who cares about images? *visual recognition systems*



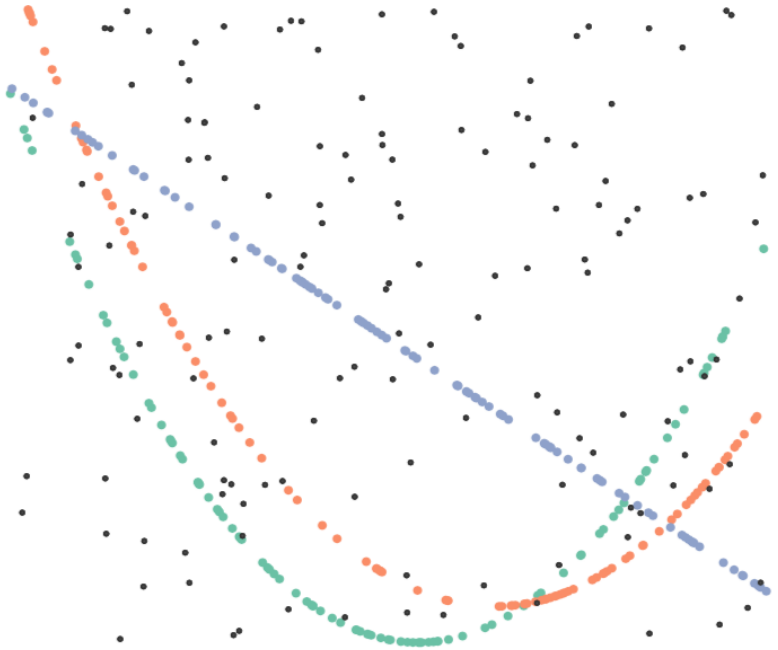
Least squares

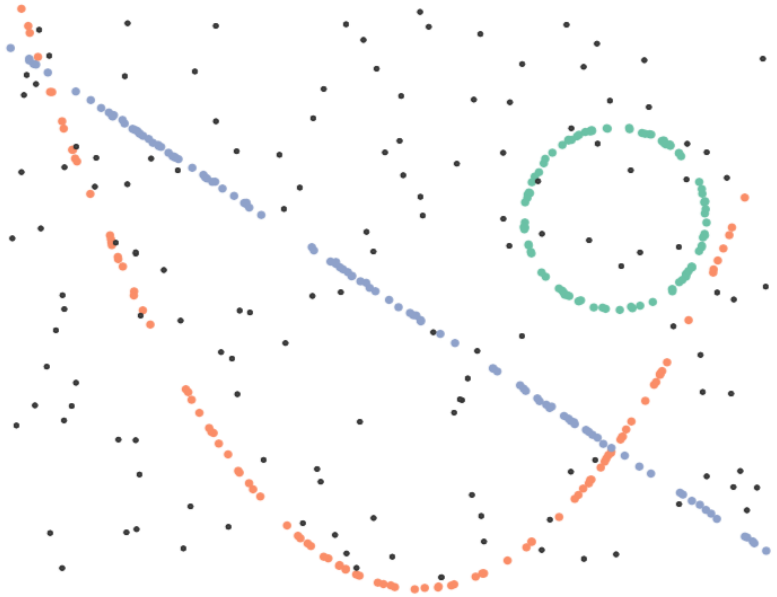


Robust Fit
(RANSAC)

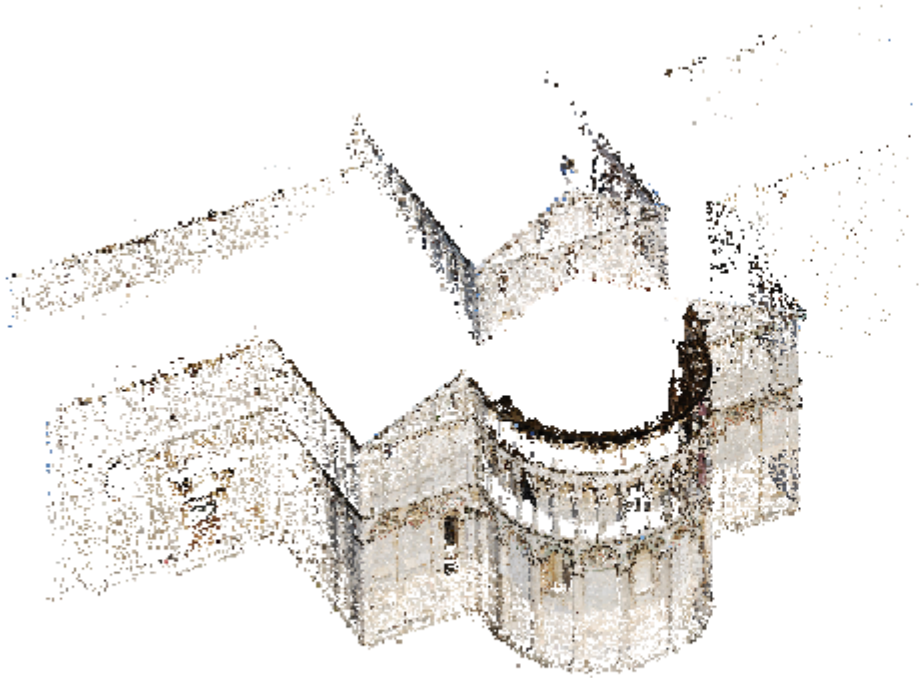
This is a (robust) fitting problem

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \sum_{\boldsymbol{x_i}} \rho(\operatorname{dist}(\boldsymbol{x_i}, \mathcal{M}_\theta))$$

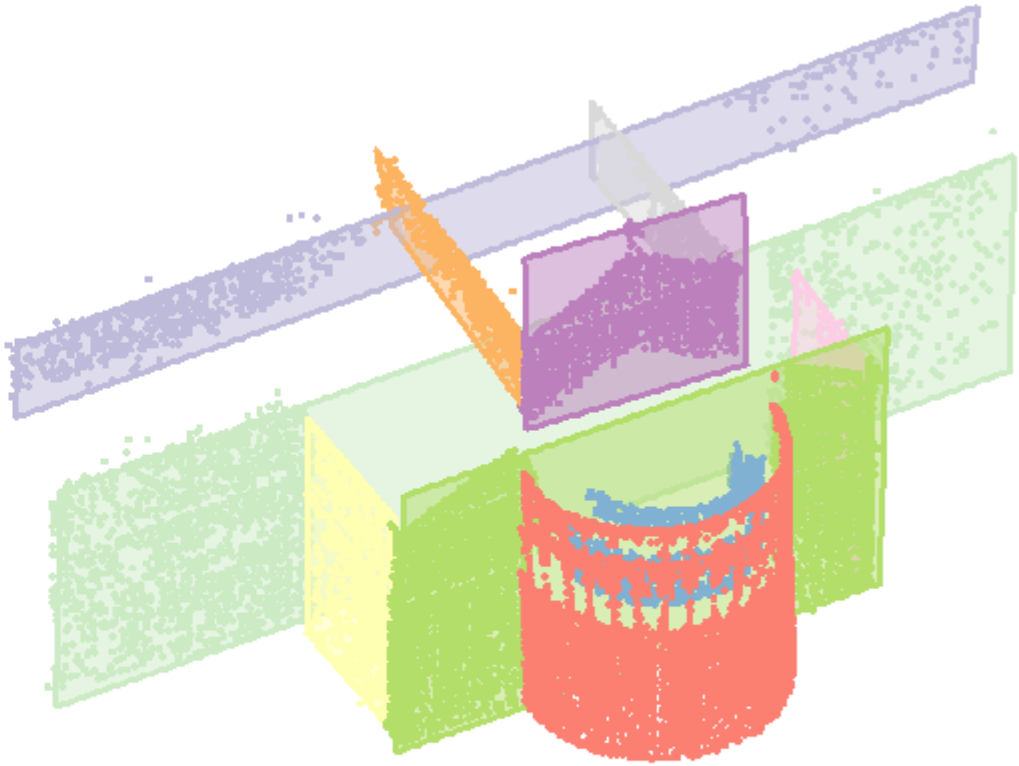




Who cares about images? *visual recognition systems*



(a) Input point cloud



(b) Recovered structures

This is a (robust) fitting problem

Magri, Leveni, Boracchi “MultiLink: Multi-class Structure Recovery via Agglomerative Clustering and Model Selection”, CVPR 2021

Who cares about images? *visual recognition systems*

Who cares about images? *visual recognition systems*

This is a (robust) fitting problem

Is this interesting for a (perspective) Mathematical Engineer?

Is this interesting? Sure!

All the algorithms build upon:

- a **clear problem formulation**
- a **simple mathematical model (…often linear combinations!)**
- Sound mathematical solutions (linear algebra, least squares, convex optimization)

…and the result is not just a number… it's an image!

Ok, to recap

Mathematical Models and Methods for Image Processing (5 CFU)

The primary ***goal*** *of this laboratory course is to* ***let the students design, implement and practice algorithms*** *based* ***on***

simple mathematical models *from* ***linear algebra and convex optimization,***

and ***solve*** *challenging inverse* ***problems in image processing*** *(denoising, deblurring, inpainting, anomaly detection)*

Mathematical Models and Methods for Image Processing (5 CFU)

The course **topics include**:

- **Image models based on orthonormal bases** (Fourier, wavelets), **data-driven basis** (PCA, Gram-Schmidt) and **local polynomial approximation**.

- **Sparsity and redundancy**.
  - Away from Orthonormal Basis, redundant set of generators
  - Sparse coding with ℓ^0 (OMP) or ℓ^1 norm (convex optimization ISTA, IRLS, LASSO)
  - Dictionaries yielding sparse representations and dictionary learning (KSVD)
- **Applications of sparse models** to image denoising, inpainting, anomaly detection and classification.

- **Robust fitting** methods (RANSAC, LMEDS, HOUGH) and their sequential counterparts for object detection in images.

Course Organization

Lectures: 20 hours

Laboratory: 30 hours

There will be short theory recap and then you will be invited to develop and practice presented algorithms. Some demo code to fill in will be provided.

Simple assignment provided during lectures, oral exam.

Frequently Asked Questions

Q: Any specific background?

A: linear algebra, statistics and calculus

Q: Any programming skill required?

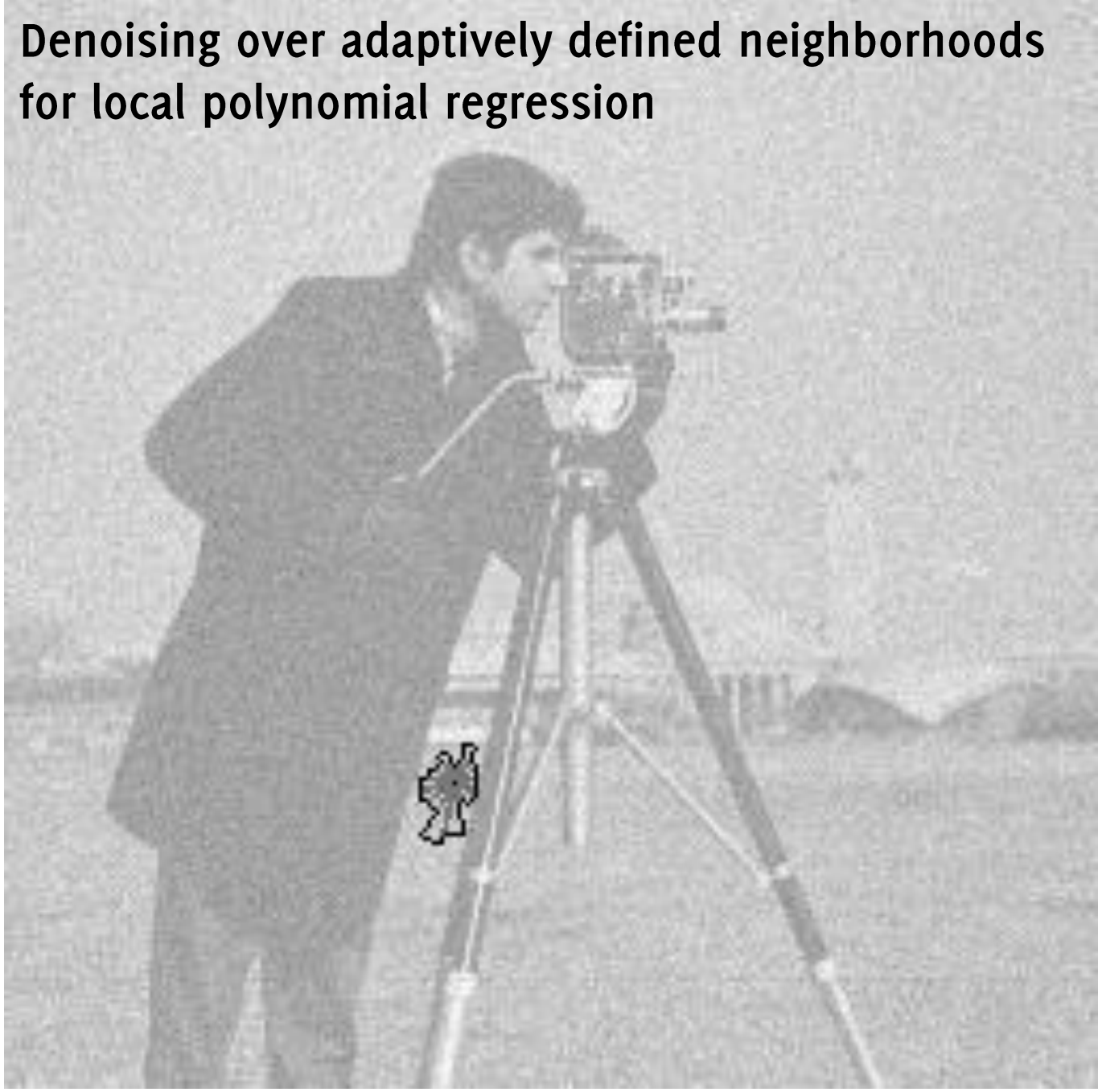
A: Proficiency in Matlab or Python

Q: Plenty of neural networks then?

A: No way. No neural networks allowed here ☺*
Only expert-driven algorithms designed upon a clear mathematical modeling that admits closed-form solutions / sound optimization schemes.

*** Interested in neural networks? Refer to «Artificial Neural Network and Deep Learning» in the first semester**

Questions?


Denoising over adaptively defined neighborhoods
for local polynomial regression

A few project and thesis opportunities

Updated thesis opportunities and furthre information on
https://boracchi.faculty.polimi.it/teaching.html

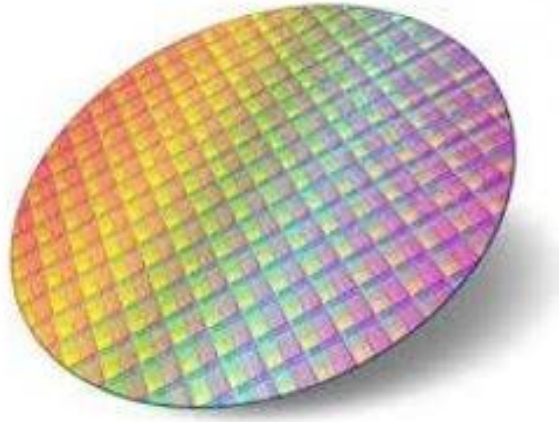
Wafer Defect Maps Monitoring

Semiconductor manufacturing is a **long, complex and expensive** process involving several specialized steps

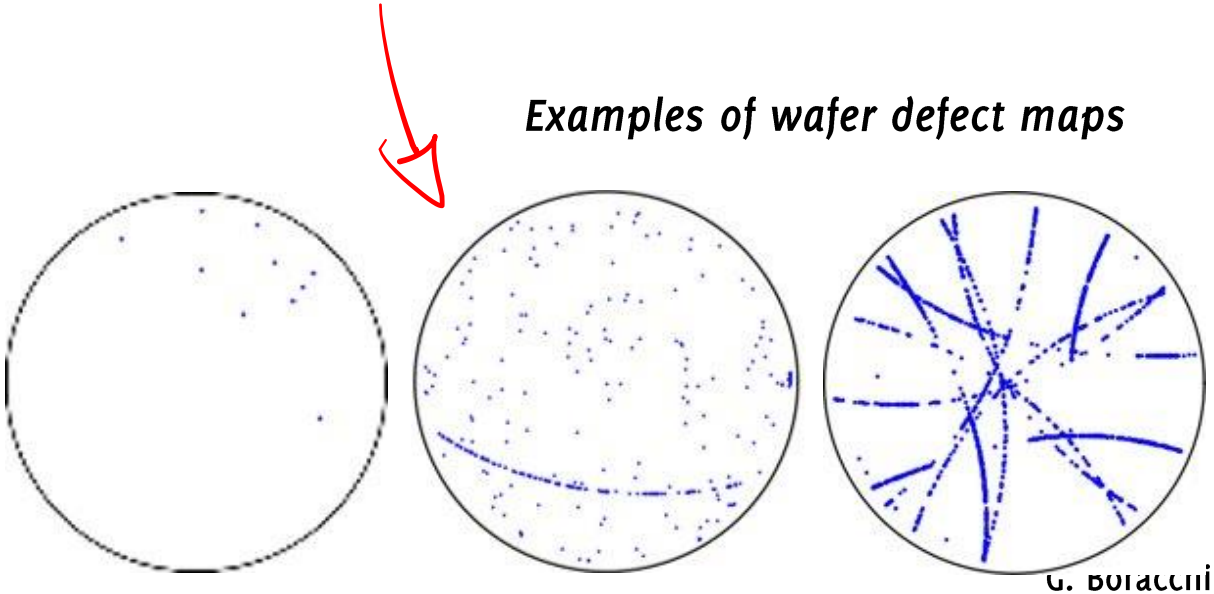
Inspection tools provide a **huge amount of data**, that can be entirely analysed only by **Deep Learning** techniques

Deep Learning enables the **automatic** monitoring of the production lines, **identifying defective patterns**. The results of this procedure allow to discard defective products and might promptly reveal failures of the machines

A wafer containing multiple chips



Examples of wafer defect maps



Stage ST1: Wafer Defect Maps Monitoring

Currently, our automatic solution based on a CNN is classifying WDMs on the major production lines of STMicroelectronics site in Agrate. Future research will address

- The study of **domain adaptation** techniques to handle data coming from different production processes
- Design a solution to **identify the type of malfunctioning** and localize the faulty machinery in the production pipeline (metric learning)
- The development of an **anomaly-detection network** to identify never seen defective patterns

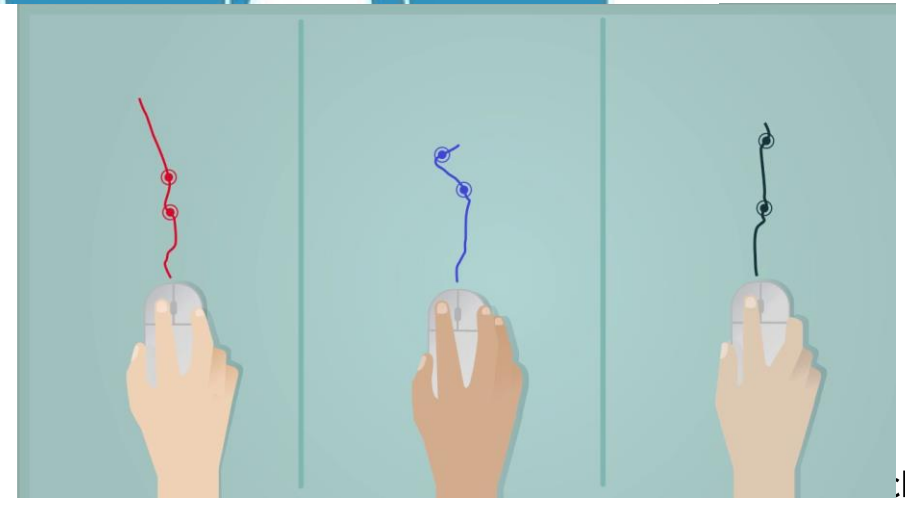
Materials and methods:

- Annotated WDM dataset
- Support from ST's engineers
- Reference algorithm based on sparse convolutions

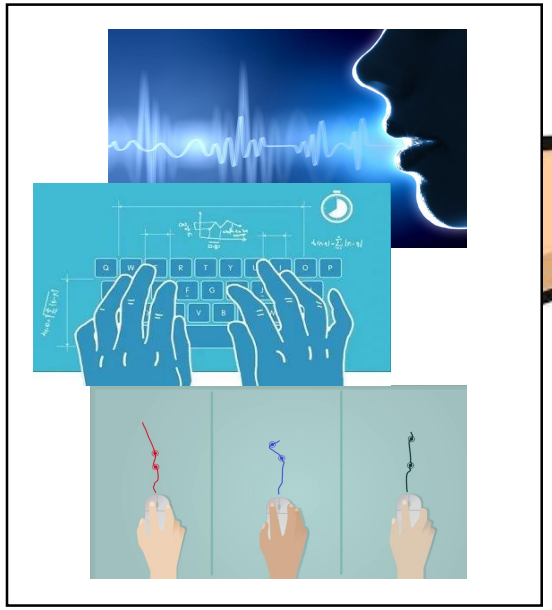
Behavioral biometrics

Behavioral biometrics is the study related to **uniquely identifiable** and **measurable patterns** in human **activities**.

Behavioral biometric verification methods include **keystroke dynamics**, **gait analysis**, **voice ID**, **mouse use characteristics**, **signature analysis** and **cognitive biometrics**.



Behavioral biometrics for fraud detection



.Cleafy

- Discriminate between **humans** and **bots** (prevent **automated attacks**).
- Identify **fraudsters** (previously seen **malicious behaviors**).
- Predict the **credentials** through which the user will **authenticate** himself (likelihood of being a **specific known user**).
- Corroborate the identity of an **authenticated user**.

Thesis: User Identification

Goal: Design and prototype an algorithm able to **verify** the user **identity** through his **behavioral biometrics** during a navigation session.

- Getting familiar with **state-of-the-art** behavioral biometrics verification **methods**.
- Robustly **profile users** through the various aforementioned methods.
- **Continuously** (and **frictionless**) **verify** the users **identities** during sessions.

Materials and Methods:

- Access to a server mounting GPUs will be provided.
- Anonymized data acquired from a large database of users.

Collaboration with

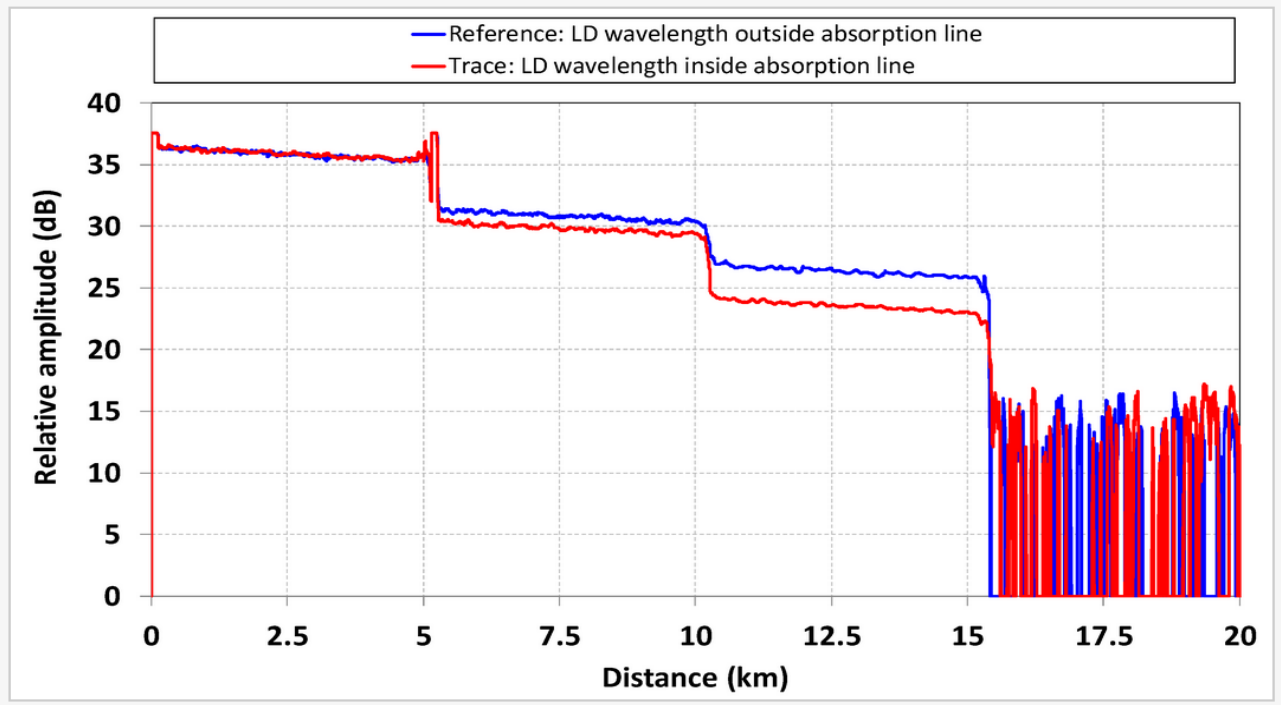
Cleafy

racchi

Stage CISCO: OTDR events recognition

OTDR (optical time domain reflectometer) is a technique to analyse a portion of an optical fiber.

The output of OTDR is a signal representing the optical power (in dB) as a function of the distance (in Km) along the fiber.

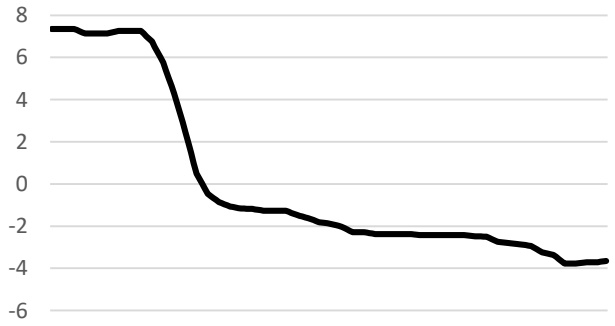


Stage CISCO: OTDR events recognition

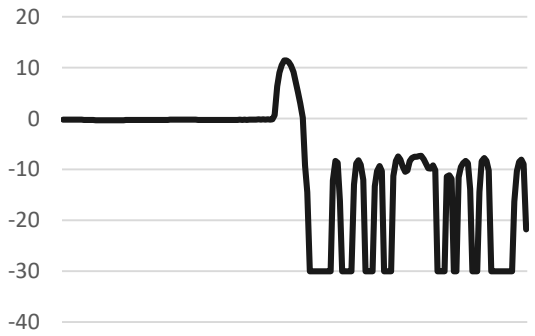
Every span of optical fiber in the field features its own characteristic. ODTR signal has the potential to identify whether some event has occurred in the fiber span.

In practice, by analyzing OTDR signal it is posible to identify at which location in the fiber there has been a cut, a strong attenuation, the presence of reflection sources, or concentrated losses, to name a few examples:

FIBER CUT



ATTENUATION



FIBER KNOTTED

G. Boracchi

Stage CISCO: OTDR events recognition

Goals:

- Implement a deep learning model to detect (multiple) patterns inside an ODTR signal. The model has to be 1-dimensional and able to cope with input signals of different dimensions.
- Define realistic data augmentation procedures in collaboration with CISCO engineers and scientists.
- Data set manipulation and preparation.
- Comparison among different solutions (expert-driven vs data-driven)
- Porting the final model on a real embedded systems.

Materials and Methods:

- OTDR measurements from real fiber deployed in multiple sites.
- CISCO engineers will provide all the guidance and equipments needed.

A Laboratory Course (but not on deep learning)

Next year, second semester (hopefully)

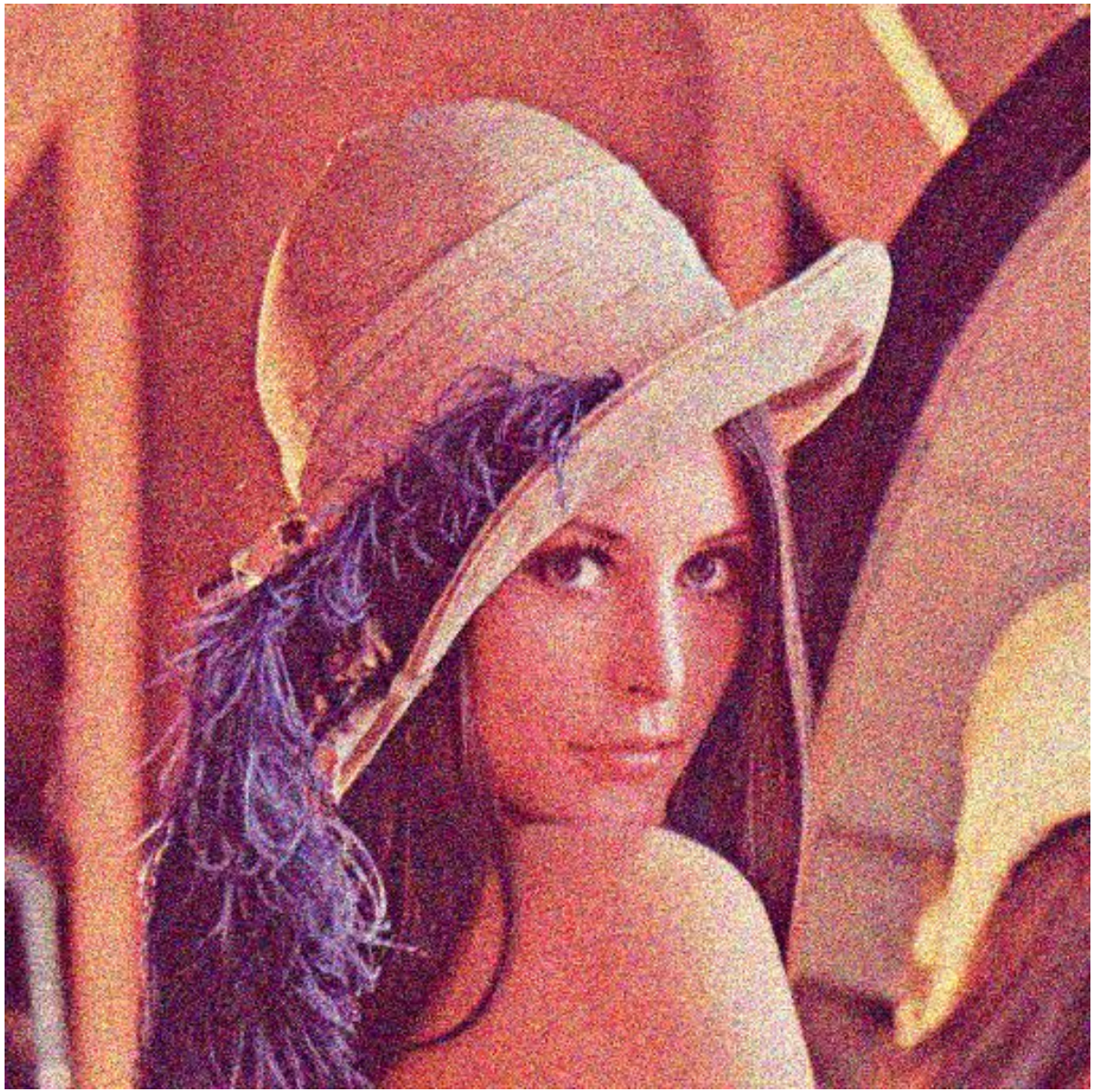
If you are planning to be here in a year…

Laboratory of Mathematical Models and Techniques for Image and Signal Processing (5 CFU)

Spring AA 2021 – 2022, Laboratory under preparation for Mathematical Engineers (and hopefully Computer Science Engineers)

The course presents ***mathematical models underpinning*** *several* ***successful image/signal processing and pattern recognition algorithms,*** *including the solution of inverse problems and fitting in noisy data. Particular emphasis will be given to models describing images as elements of a vector space, and the implications of linear algebra theory in imaging. These include transformed domain methods and other techniques to solve inverse problems.*

Restoration & Inverse Problems

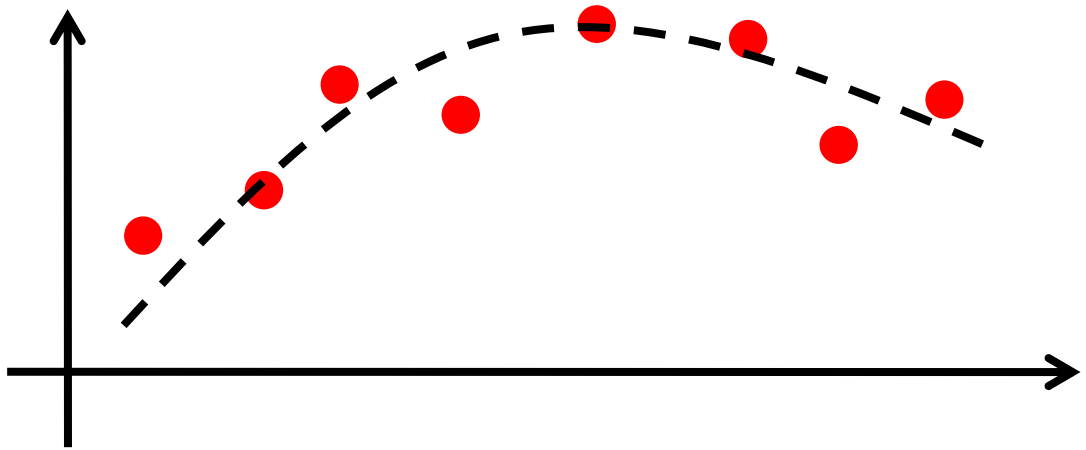


Dabov, K., Foi, A., Katkovnik, V., & Egiazarian, K. Image denoising by sparse 3-D transform-domain collaborative filtering. *TIP 2007*

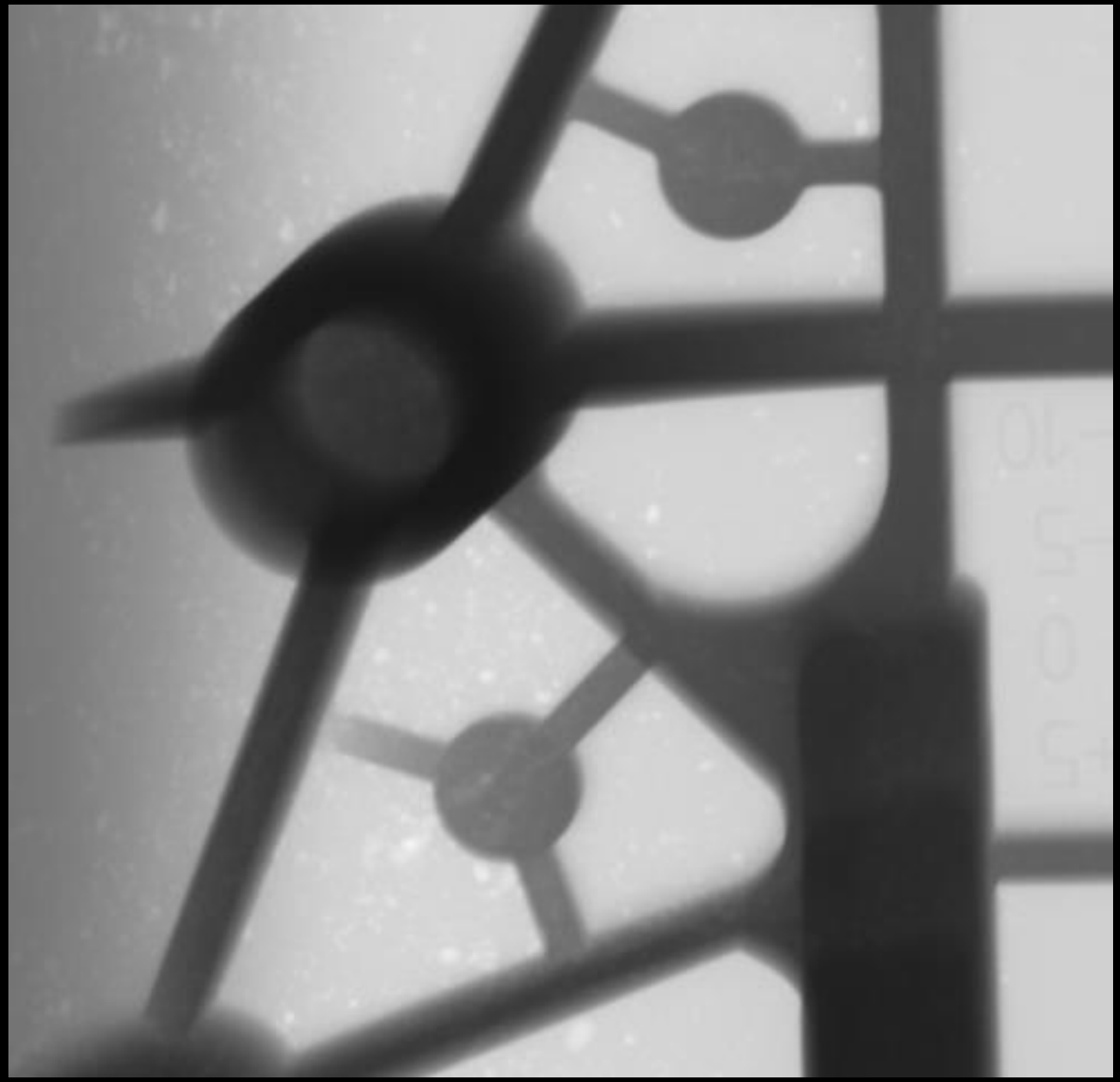
Restoration & Inverse Problems

Dabov, K., Foi, A., Katkovnik, V., & Egiazarian, K. Image denoising by sparse 3-D transform-domain collaborative filtering. *TIP 2007*

Filtering



Input

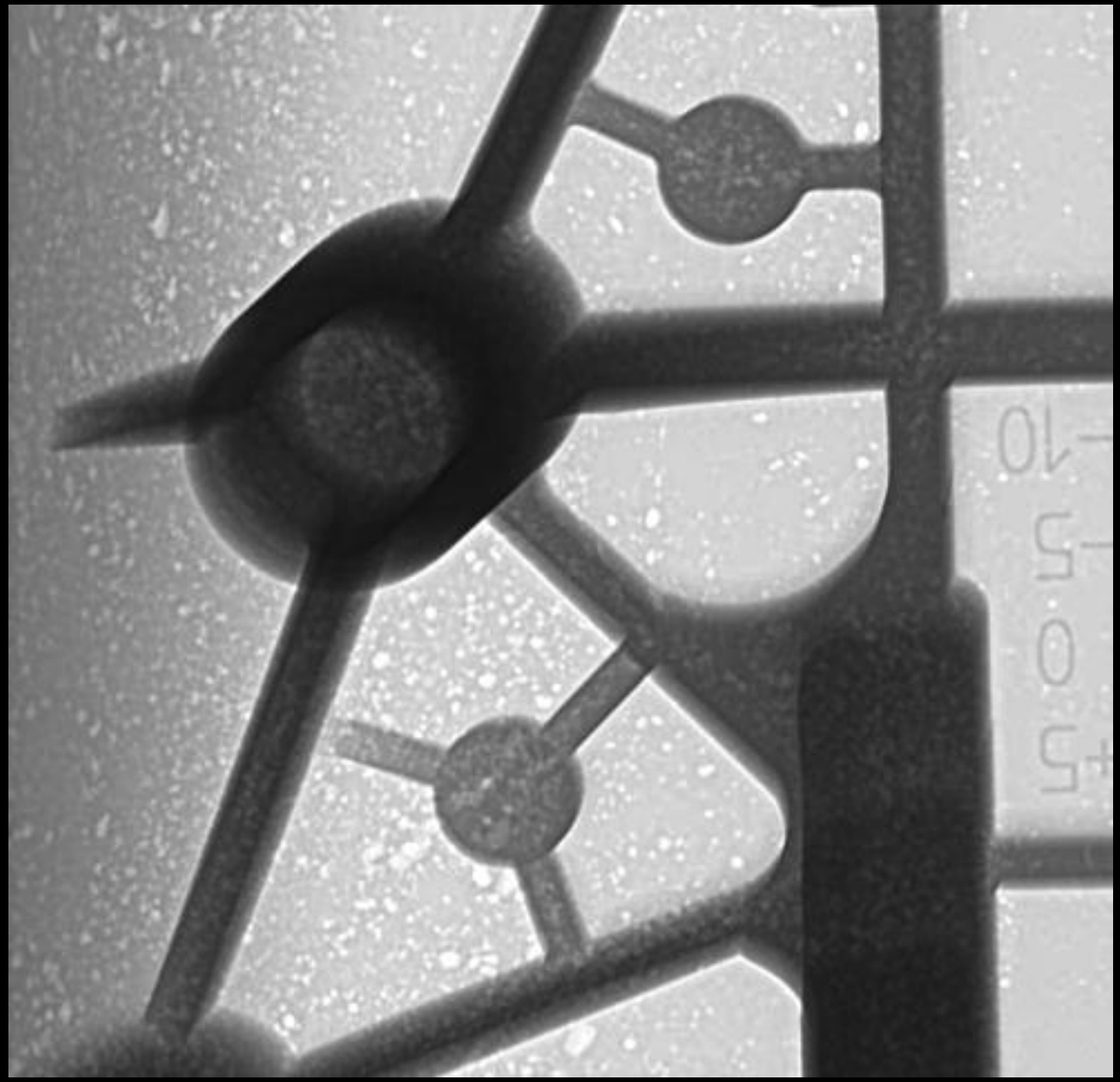


Collaboration with

GILARDONI

G. Boracchi

Enhanced



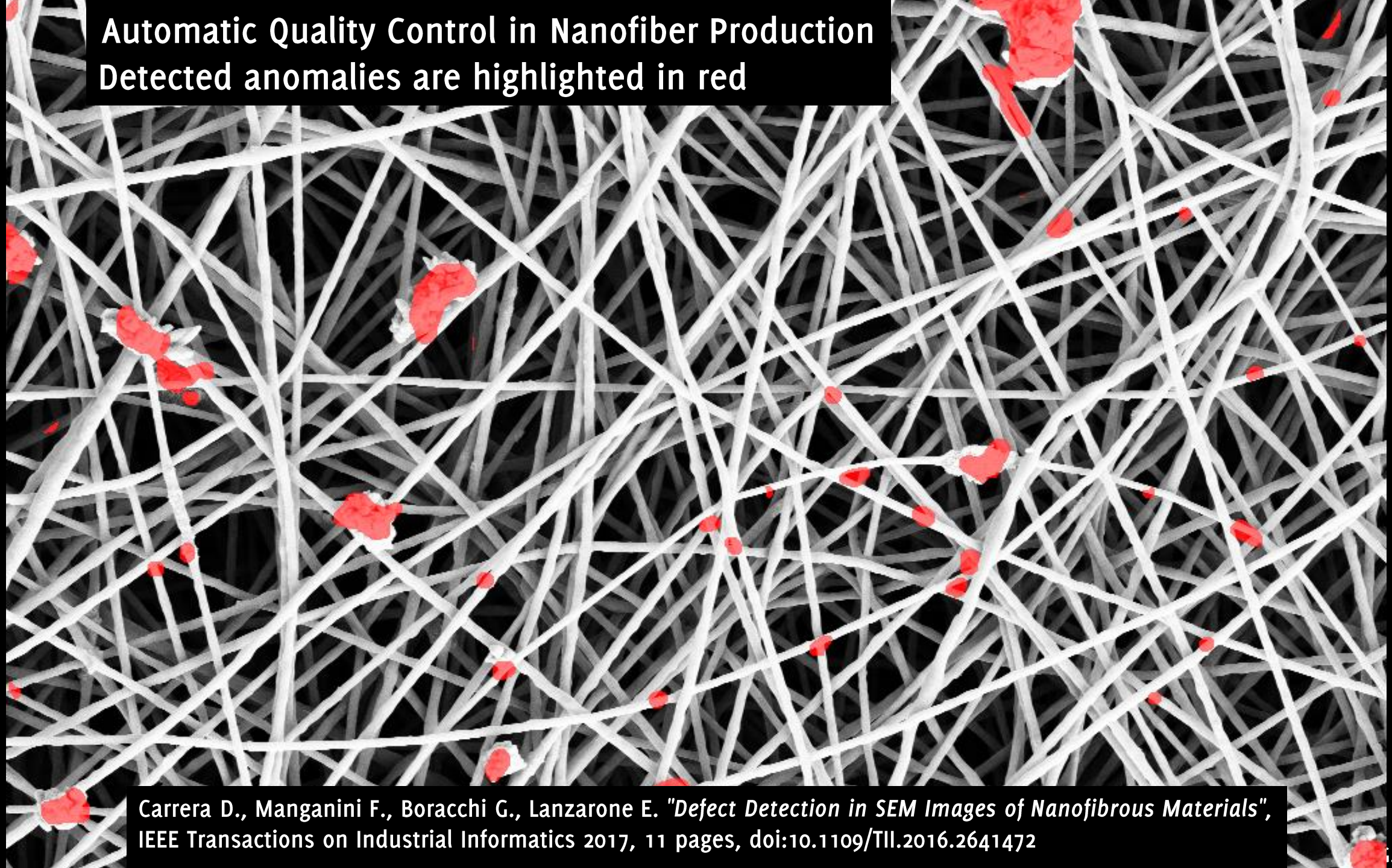
Collaboration with

GILARDONI

G. Boracchi

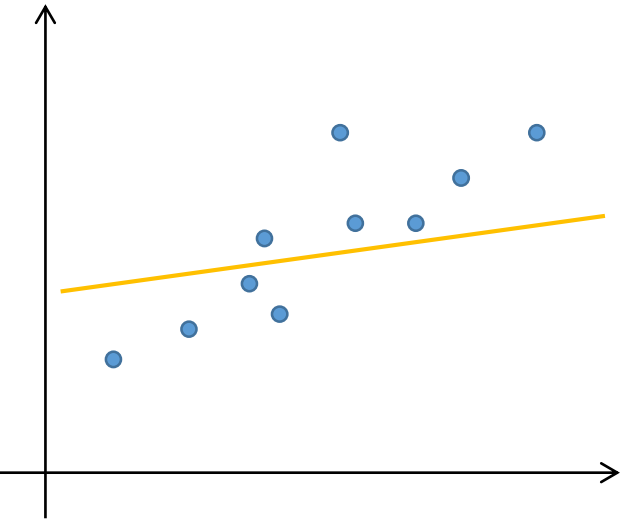
Automatic Quality Control in Nanofiber Production

Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

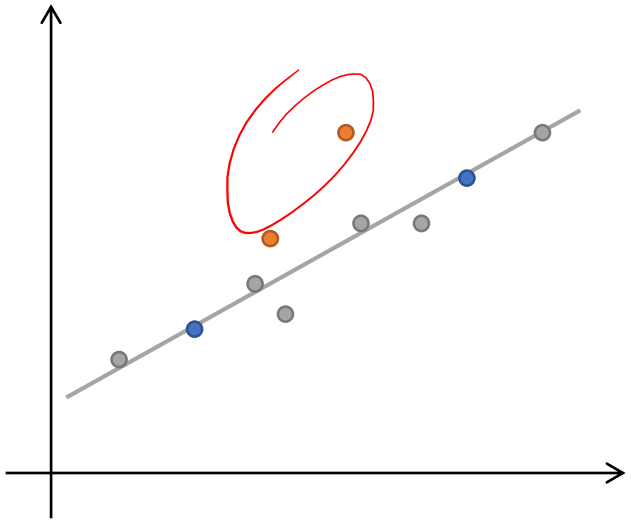


Carrera D., Manganini F., Boracchi G., Lanzarone E. *"Defect Detection in SEM Images of Nanofibrous Materials"*, IEEE Transactions on Industrial Informatics 2017, 11 pages, doi:10.1109/TII.2016.2641472

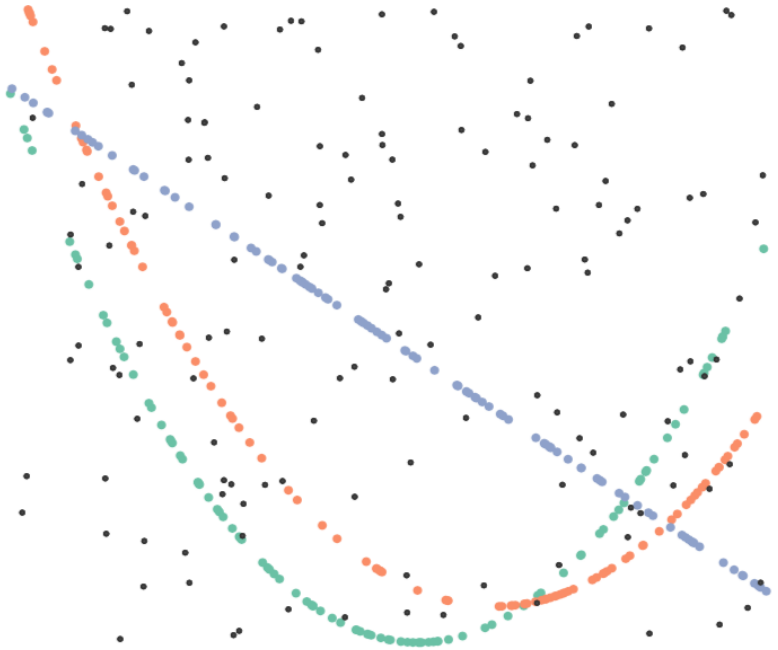
Model Fitting

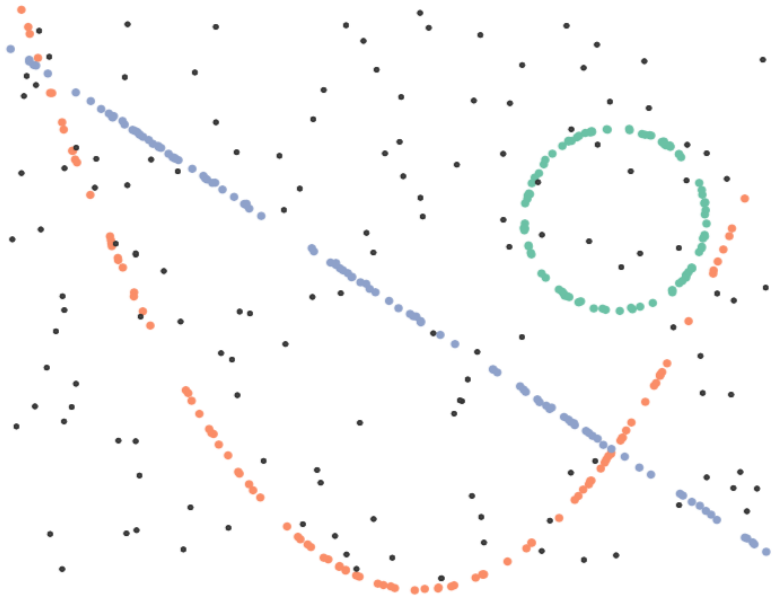


Least squares



Robust Fit
(RANSAC)

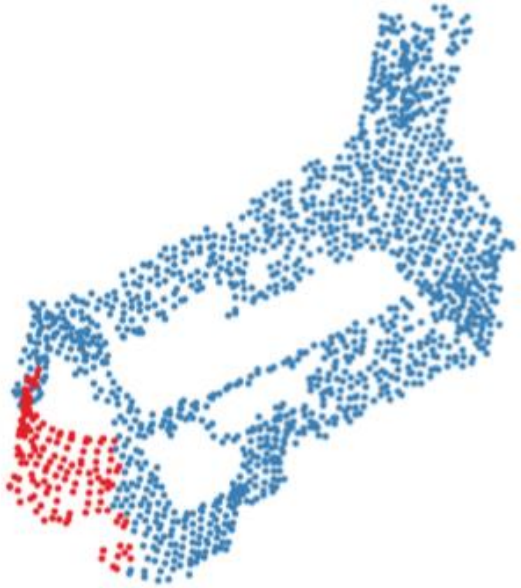




Model Fitting

Model Fitting

Image of the scene



colour coded class

colour coded model

Magri, Luca, and Andrea Fusiello. "Fitting Multiple Heterogeneous Models by Multi-Class Cascaded T-Linkage." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

Learning Sparse Represenations for Image and Singal Modeling

a PhD Course open to MSc Students
on Febrauary 2021

A PhD Course open to MSc Students on February 2021

The main goal of this course is to provide the student with an understanding of the most important aspects of the theory underlying sparse representation and, more in general, of sparsity as a form of regularization in learning problems. Students will have the opportunity to develop and understand the main algorithms for learning sparse models and computing sparse representations. These methods have wide applicability in computer science, and these will be a useful background for their research.

More information on the Course program page

https://boracchi.faculty.polimi.it/teaching/LearningSparse.htm

(dates to be updated)

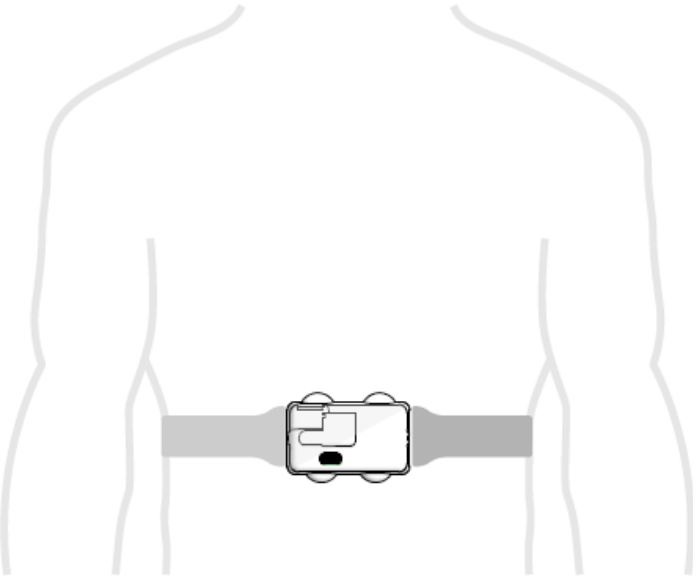
Dates

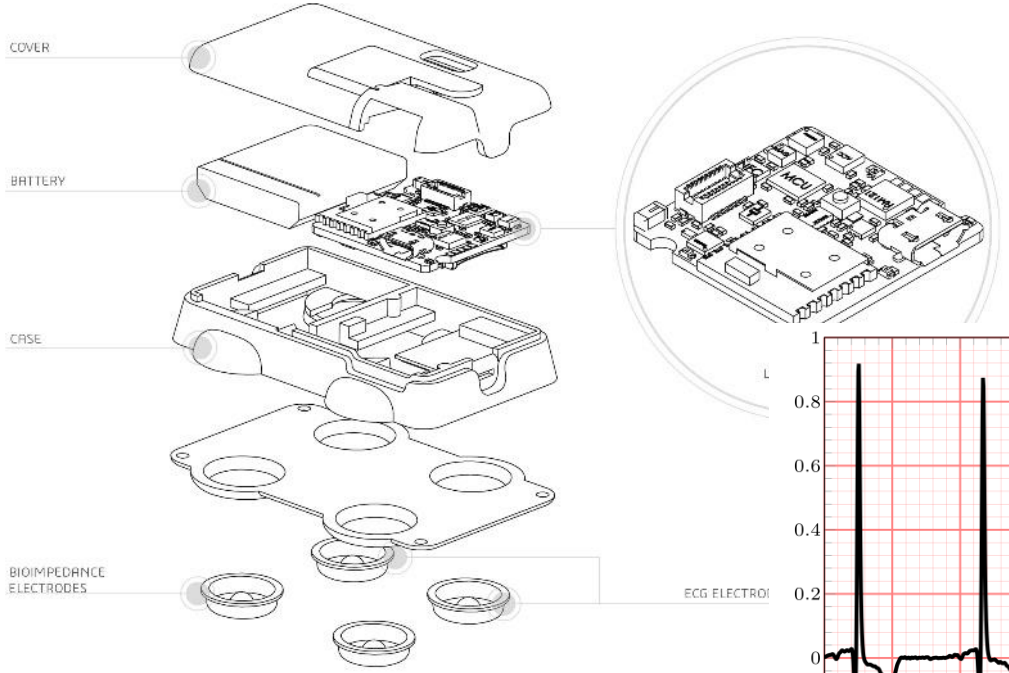
The course consists of traditional classes and computer-laboratory hours. During computer laboratory, students will be guided in the implementation of simplified versions of the presented algorithms. Imaging problems, including denoising, inpainting and anomaly detection, will be considered as running examples through the computer labs. Matlab is the preferred programming language (but you are of course allowed to use Python) and students are invited to work on their own laptop

- 02/02/2021 from 14:00 til 18:00
- 05/02/2021 from 14:00 til 19:00
- 09/02/2021 from 14:00 til 18:00
- 12/02/2021 from 14:00 til 18:00
- 16/02/2021 from 14:00 til 18:00
- 19/02/2021 from 14:00 til 18:00

An Example of algorithm we will see as an application: AD for Long Term ECG monitoring in wearables

Collaboration with





A wearable device to enable **online and long term ECG monitoring**. Automatic **detection of anomalies** (e.g., arrhythmias) and **domain adaptation** to track heart rate variations. The model is based on **a learned and user-specific model** of heartbeats.

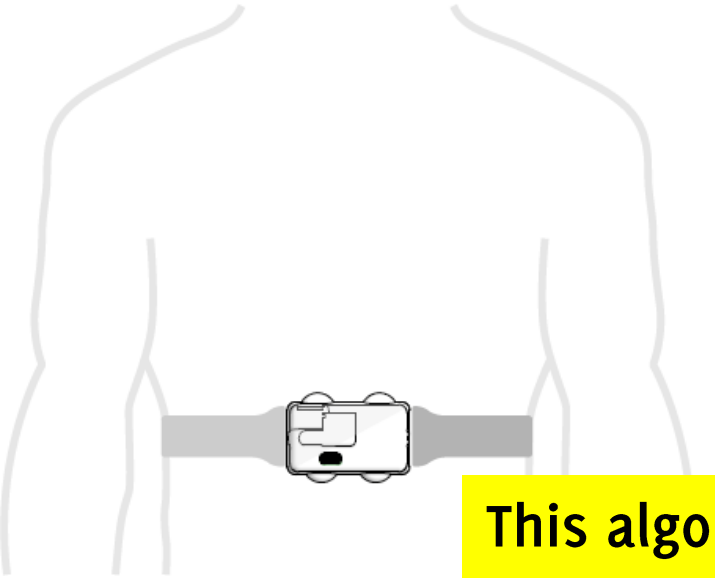
IJCAI Demo webpage: http://home.deib.polimi.it/carrerad/IJCAI_2018_Demo.html

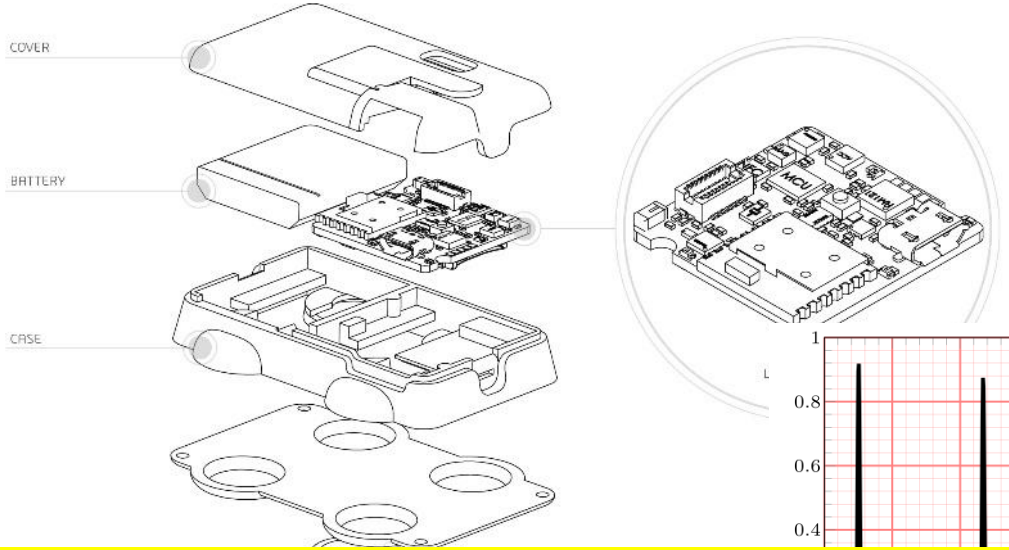
G. Boracchi

An Example of algorithm we will see as an application: AD for Long Term ECG monitoring in wearables

Collaboration with

life.augmented





This algorithm has been implemented on the device and our patent has been granted

A wearable device to enable **online and long term ECG monitoring**. Automatic **detection of anomalies** (e.g., arrhythmias) and **domain adaptation** to track heart rate variations. The model is based on **a learned and user-specific model** of heartbeats.

IJCAI Demo webpage: http://home.deib.polimi.it/carrerad/IJCAI_2018_Demo.html