Imperial College London

Imperial College London

Machine Learning for Imaging
Image Registration

Daniel Rueckert
Department of Computing
Imperial College London, UK

Inverse problems in imaging

- Observe:

$$y = Ax + n$$

Inverse problems in imaging

- Observe:

$$y = Ax + n$$

- Goal:

Recover x from y

Inverse problems in imaging

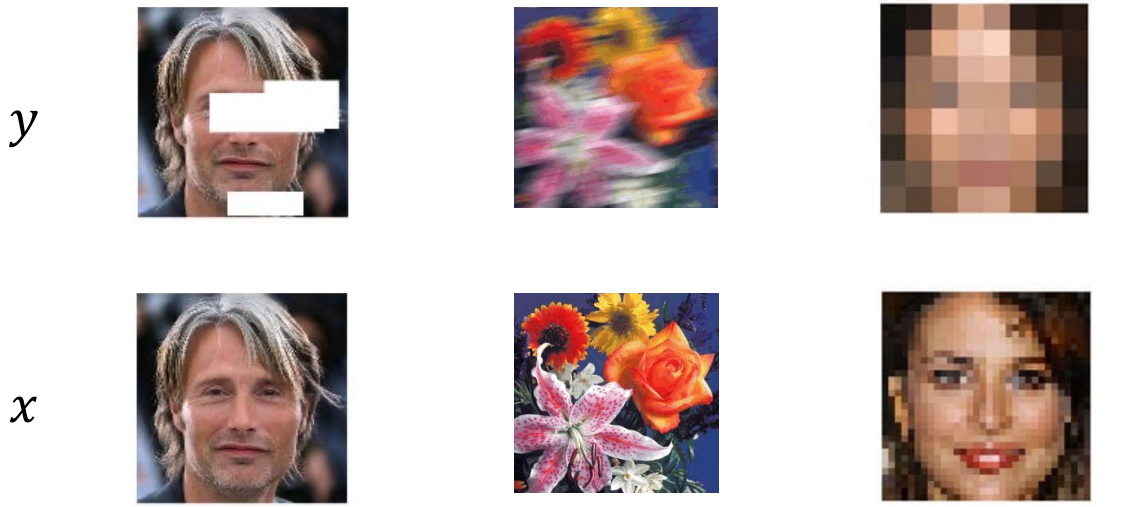
- Observe:

$$y = Ax + n$$

- Goal:

Recover x from y

- Inpainting
- Deblurring
- Denoising
- Super-resolution
- Image Reconstruction (Medical imaging)



Classical Approach

- Example: deblurring

Classical Approach

- Example: deblurring
- Least squares problem:

$$\arg\min_x \mathcal{D}(Ax, y)$$

$$\mathcal{D}(Ax, y) = \frac{1}{2}\|Ax - y\|_2^2$$

Operator A describes the forward map (here a linear operation)

Classical Approach

- Example: deblurring

- Least squares problem:

$$\arg\min_x \mathcal{D}(Ax, y)$$

$$\mathcal{D}(Ax, y) = \frac{1}{2}\|Ax - y\|_2^2$$

- Least squares solution:

$$\hat{x} = (A^T A)^{-1} A^T y$$

Classical Approach

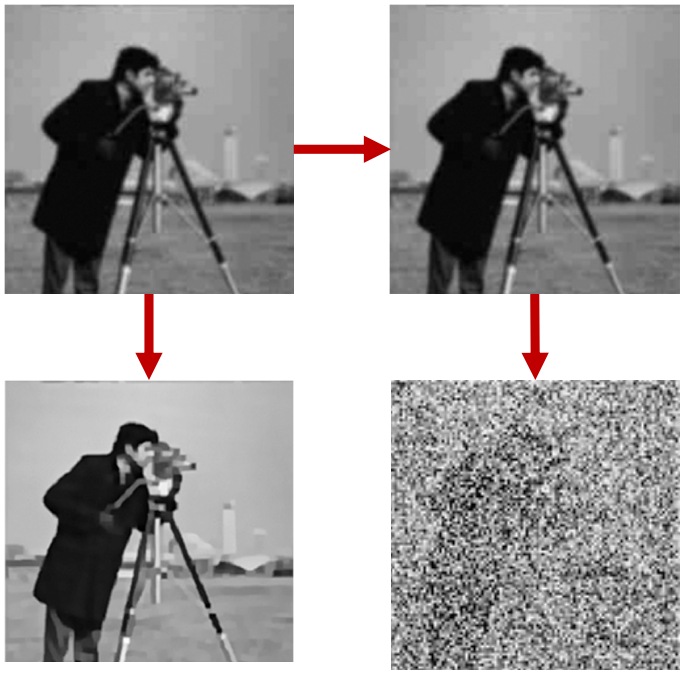
- Example: deblurring
- Least squares problem:

$$\arg\min_x \mathcal{D}(Ax, y)$$

$$\mathcal{D}(Ax, y) = \frac{1}{2}\|Ax - y\|_2^2$$

- Least squares solution:

$$\hat{x} = (A^T A)^{-1} A^T y$$



Wildly different solutions

Classical Approach

- Example: deblurring

- Least squares problem:

$$\arg\min_x \mathcal{D}(Ax, y) + \lambda \mathcal{R}(x)$$

Regularisation

$$\mathcal{D}(Ax, y) = \frac{1}{2}\|Ax - y\|_2^2 \quad \text{and} \quad \mathcal{R}(x) = \frac{1}{2}\|x\|_2^2$$

Classical Approach

- Example: deblurring

- Least squares problem:

$$\arg\min_x \mathcal{D}(Ax, y) + \lambda\mathcal{R}(x)$$

Regularisation

$$\mathcal{D}(Ax, y) = \frac{1}{2}\|Ax - y\|_2^2 \quad \text{and} \quad \mathcal{R}(x) = \frac{1}{2}\|x\|_2^2$$

- Least squares solution with Tikhonov regularisation:

$$\hat{x} = \underbrace{(A^T A + \lambda I)^{-1}}_{\text{Better conditioned (and noise suppression)}} A^T y$$

General approach for images

$$\arg\min_x \mathcal{D}(Ax, y) + \lambda \mathcal{R}(x)$$

$\mathcal{D}(Ax, y)$	Data consistency term
$\mathcal{R}(x)$	Regularisation term (encoding prior knowledge on x)
λ	Regularisation parameter

General approach for images

$$\arg\min_x \mathcal{D}(Ax, y) + \lambda\mathcal{R}(x)$$

$\mathcal{D}(Ax, y)$	Data consistency term
$\mathcal{R}(x)$	Regularisation term (encoding prior knowledge on x)
λ	Regularisation parameter

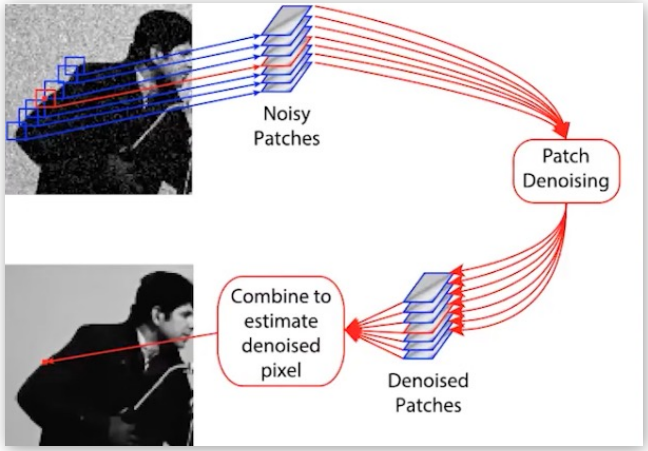
Common regularisers:

$\mathcal{R}(x) = \|\nabla x\|_2^2$	Tikhonov regularisation
$\mathcal{R}(x) = \|\nabla x\|_{2,1} = \sum_{i=0}^{n} \sqrt{\sum_d \left(\|\nabla x\|_i^{(d)}\right)^2}$	Total variation regularisation
$\mathcal{R}(x) = \|\Psi x\|_1$	Wavelet regularisation

General approaches for images

Total variation

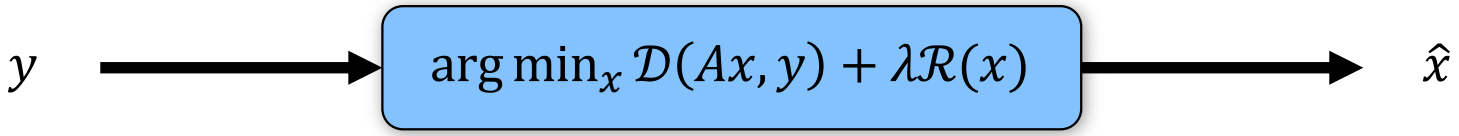
Wavelet decomposition



Patches and manifold

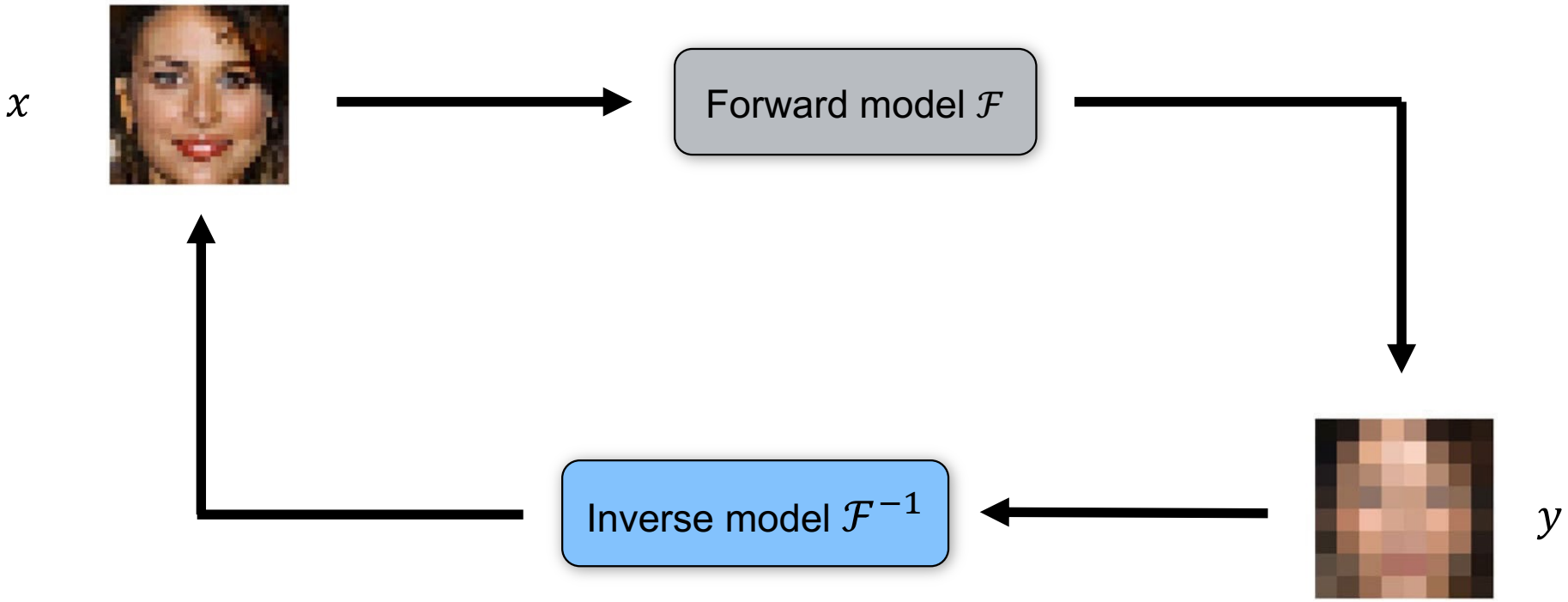
Figure adapted from Rebecca Willett

Regularisation in inverse problems

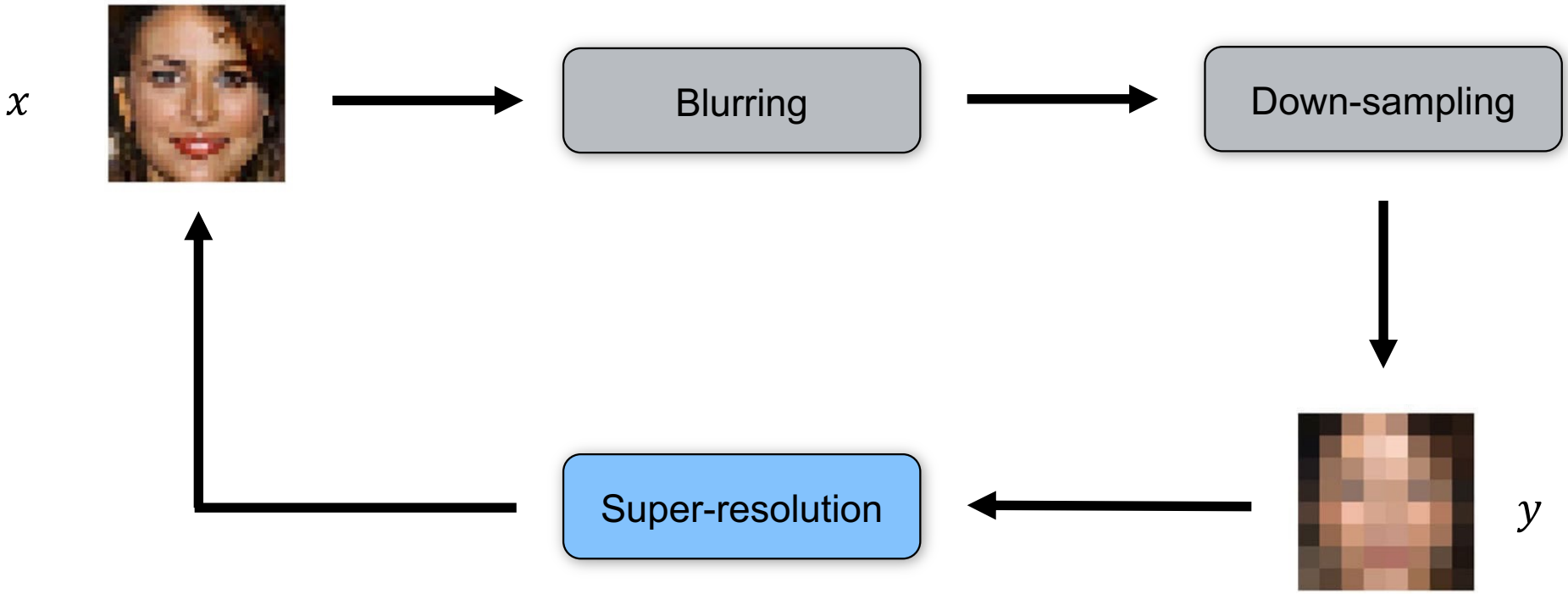


Instead of choosing $\mathcal{R}$ a-priori based on a simple (geometric or other) model of image, learn $\mathcal{R}$ from training data

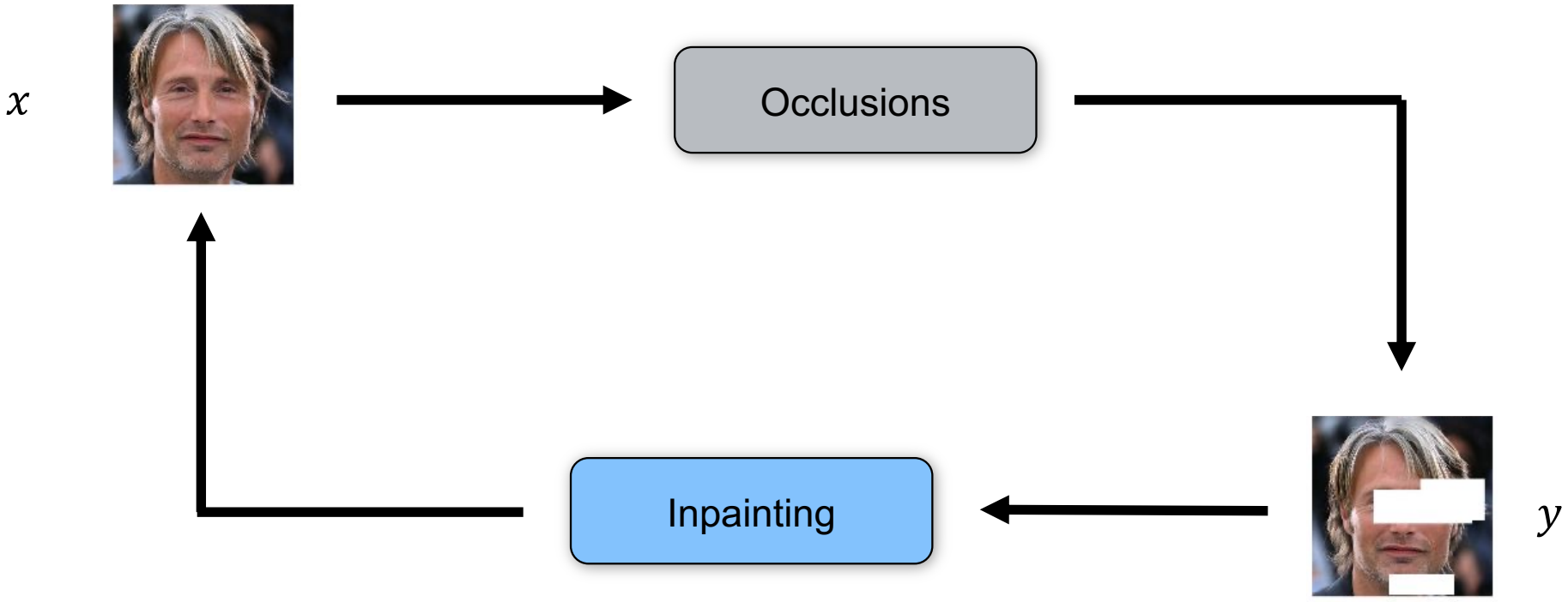
Forward model/Inverse model



Forward model/Inverse model



Forward model/Inverse model



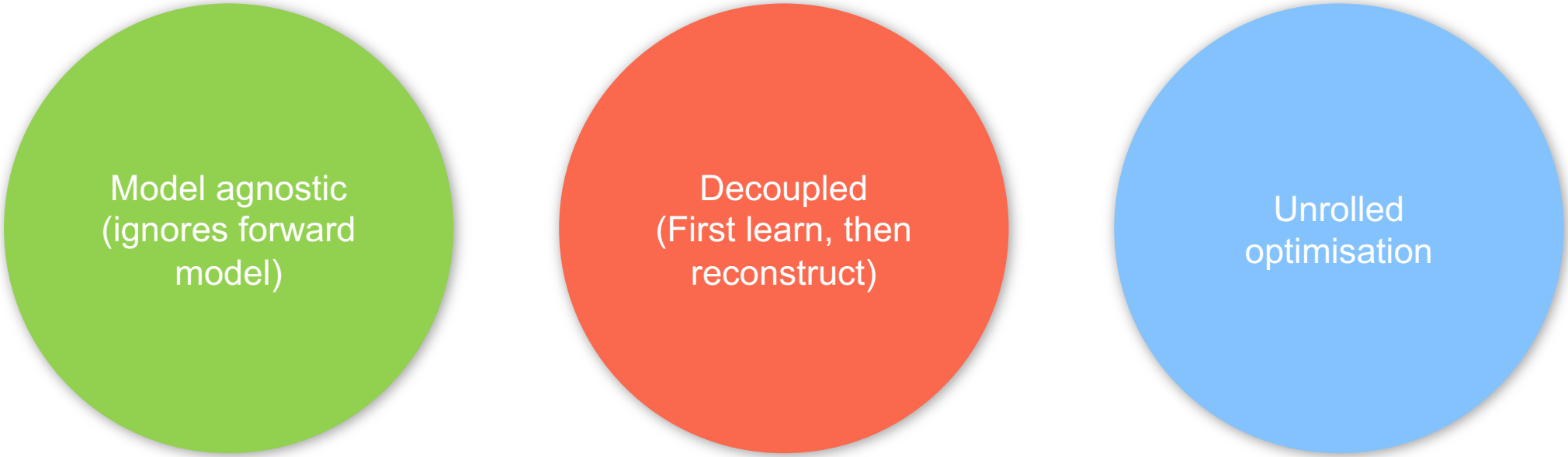
Imperial College London

Imperial College London

Solving Inverse Problems with Deep Learning

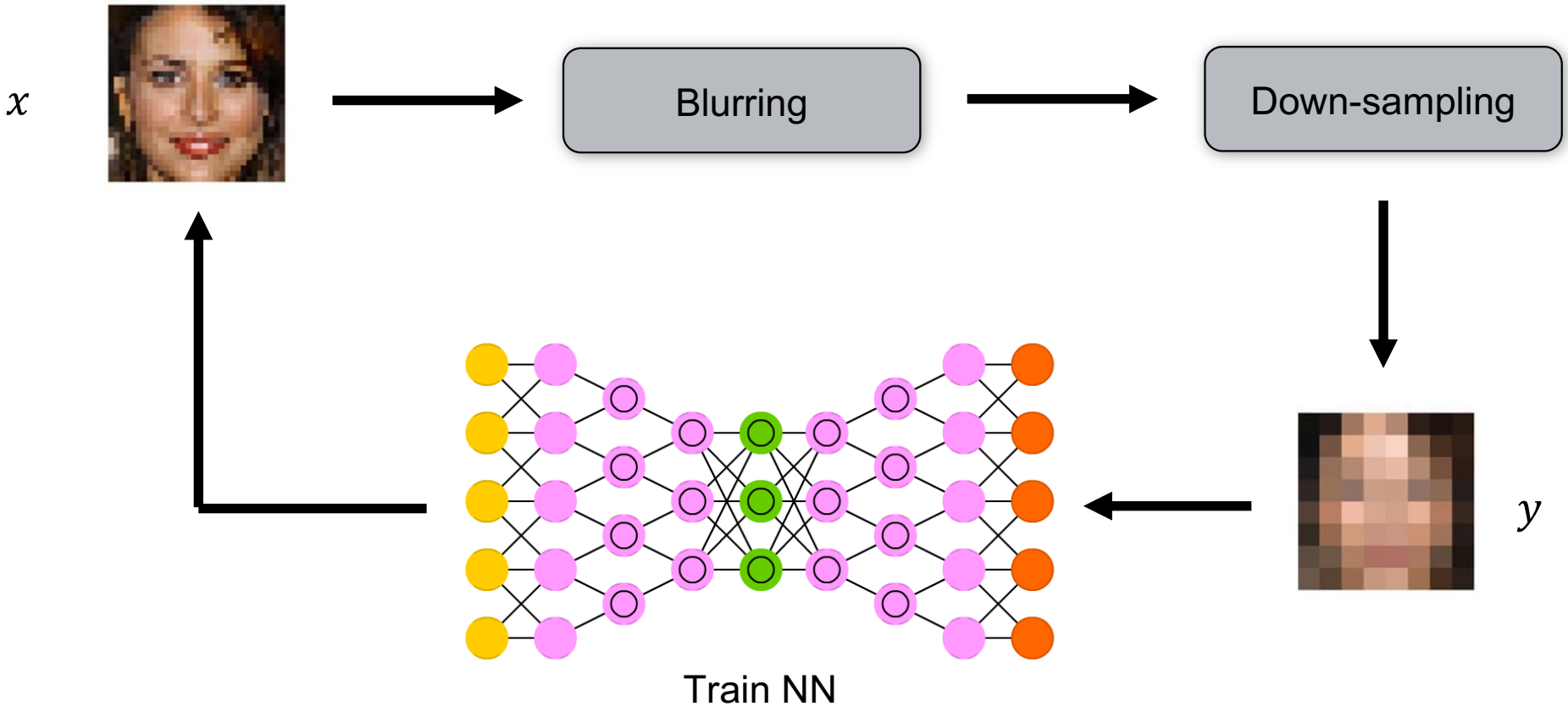
Daniel Rueckert
Department of Computing
Imperial College London, UK

Classes of methods



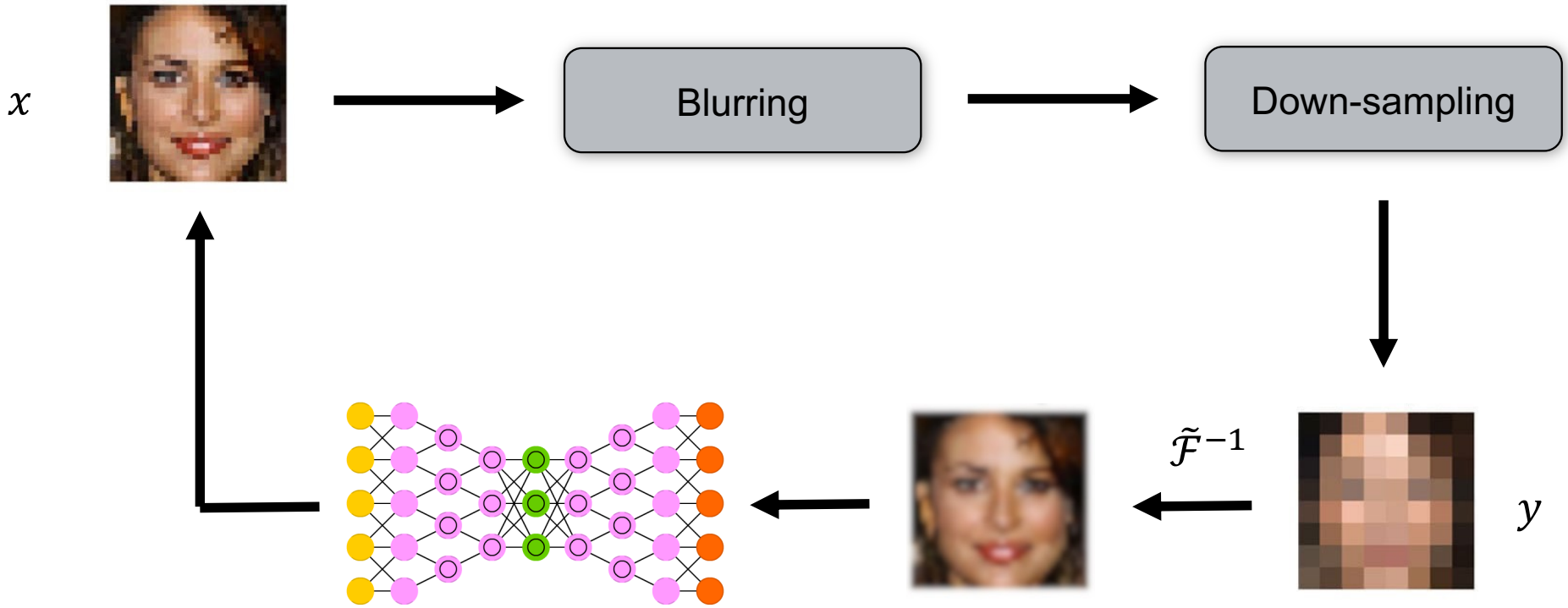
Solving the inverse problem

Model agnostic



Solving the inverse problem

(partly) model agnostic



Solving the inverse problem: Deep proximal gradient

Decoupled (First learn, then reconstruct)

$$y \longrightarrow \boxed{\arg\min_x \mathcal{D}(Ax, y) + \lambda \mathcal{R}(x)} \longrightarrow \hat{x}$$

set $\hat{x}^{(1)}$ and stepsize η

for $k = 1, 2, \cdots$

$$z^k = \hat{x}^{(k)} + \eta A^T \left(y - A\hat{x}^{(k)}\right)$$

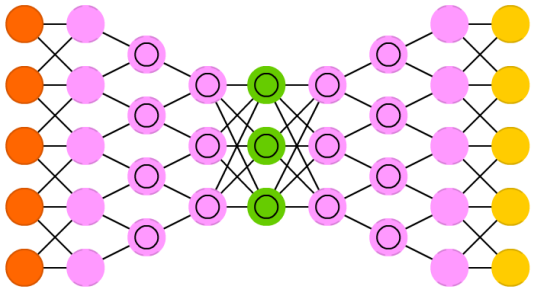
gradient descent (DC)

$$\hat{x}^{(k+1)} = \boxed{\arg\min_x \left\|z^k - x\right\|_2^2 + \eta\lambda\mathcal{R}(x)}$$

denoising

replace with learned NN

z^k



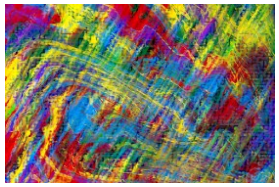
$\hat{x}^{k+1}$

Solving the inverse problem: GANs

Decoupled
(First learn, then reconstruct)

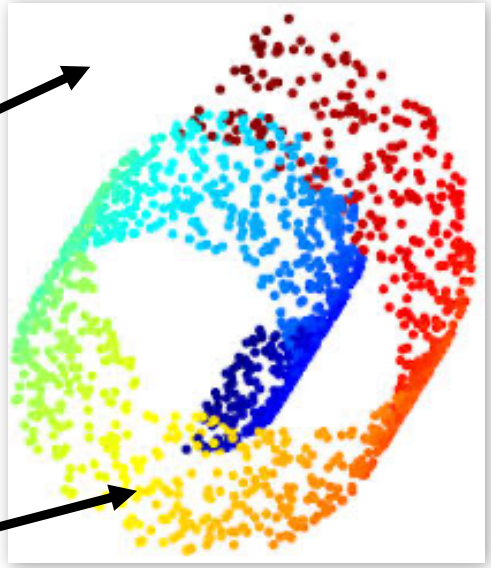
$$y \rightarrow \boxed{\arg\min_x \mathcal{D}(Ax, y) + \lambda \mathcal{R}(x)} \rightarrow \hat{x}$$

$$\mathcal{R}(x) = \begin{cases} 0 & x \text{ on image manifold} \\ \infty & \text{otherwise} \end{cases}$$

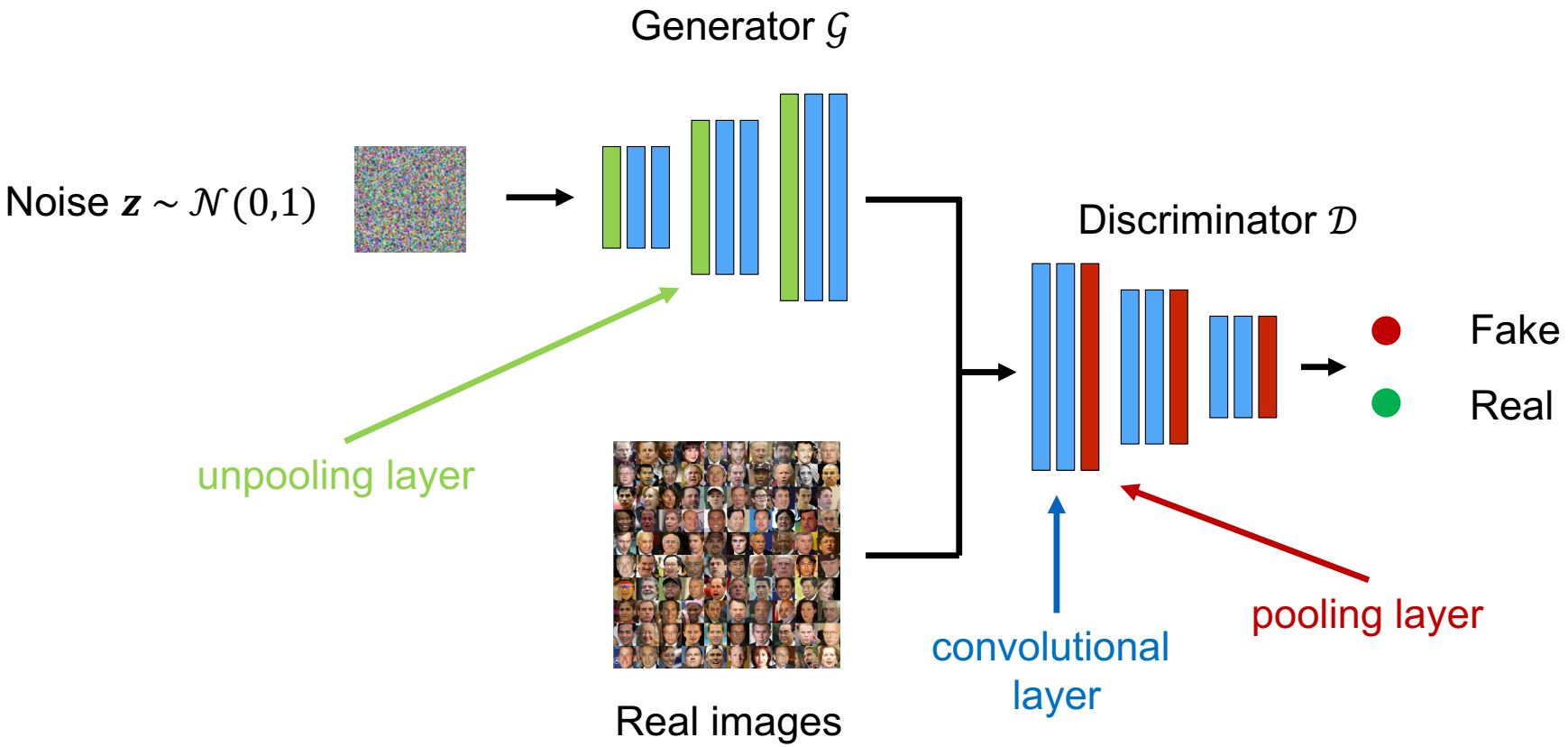


Non-natural image
(bad)

Natural image
(good)



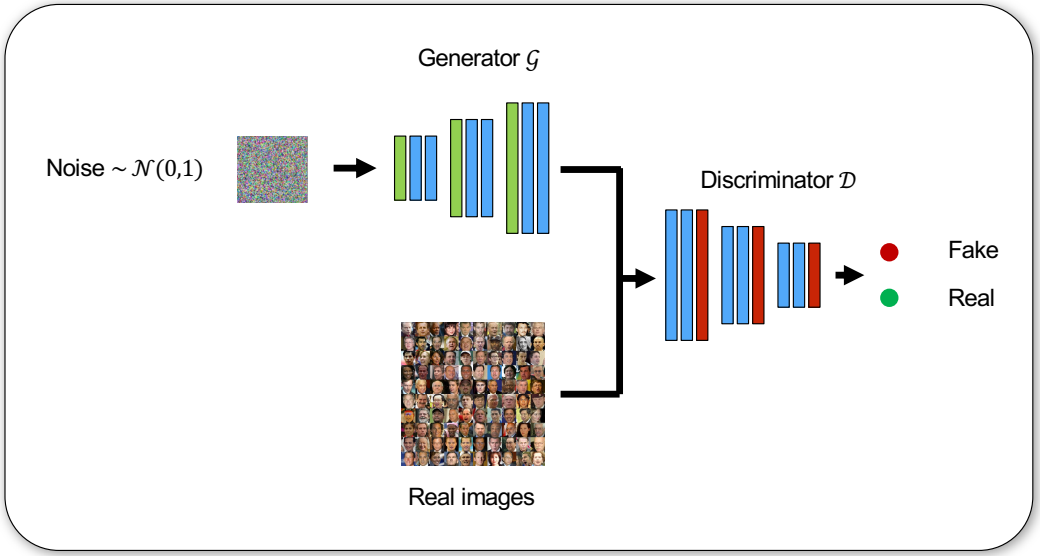
Generative adversarial networks (GANs)



I. Goodfellow, NIPS, 2014

Generative adversarial networks (GANs)

- $\mathcal{D}(x)$ represents the probability that x came from the real data rather than from $\mathcal{G}$.
- $\mathcal{D}$ is trained to maximize the probability of assigning the correct label to both real examples and samples from $\mathcal{G}$.



- Simultaneously train $\mathcal{G}$ to minimize $\log(1 - \mathcal{D}(\mathcal{G}(\boldsymbol{z}))$, e.g. $\mathcal{D}$ and $\mathcal{G}$ play a two-player minimax game with value function $V(\mathcal{G}, \mathcal{D})$:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} V(\mathcal{G}, \mathcal{D}) = \underbrace{\mathbb{E}_{x \sim p_{data}(x)}[\log \mathcal{D}(x)]}_{\text{real}} + \underbrace{\mathbb{E}_{z \sim p_z(z)}[\log(1 - \mathcal{D}(\mathcal{G}(\boldsymbol{z}))]}_{\text{synthetic (aka “fake”)}}$$

Solving the inverse problem: GANs

Decoupled (First learn, then reconstruct)

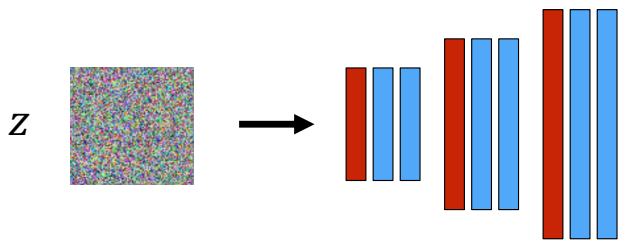
$$y \rightarrow \boxed{\arg\min_x \mathcal{D}(Ax, y) + \lambda \mathcal{R}(x)} \rightarrow \hat{x}$$

$$\mathcal{R}(x) = \begin{cases} 0 & x \text{ on image manifold} \\ \infty & \text{otherwise} \end{cases}$$

- Learn generator $\mathcal{G}$ that outputs $x \in \mathbb{R}^d$ given $x \in \mathbb{R}^{d'}$ for $d' < d$

$$\mathcal{R}(x) = \begin{cases} 0 & x \in \text{range}(\mathcal{G}) \\ \infty & \text{otherwise} \end{cases}$$

Generator $\mathcal{G}$



Solving the inverse problem: GANs

Decoupled (First learn, then reconstruct)

$$y \rightarrow \boxed{\arg\min_x \mathcal{D}(Ax, y) + \lambda \mathcal{R}(x)} \rightarrow \hat{x}$$

$$\mathcal{R}(x) = \begin{cases} 0 & x \text{ on image manifold} \\ \infty & \text{otherwise} \end{cases}$$

- Learn generator $\mathcal{G}$ that outputs $x \in \mathbb{R}^d$ given $x \in \mathbb{R}^{d'}$ for $d' < d$

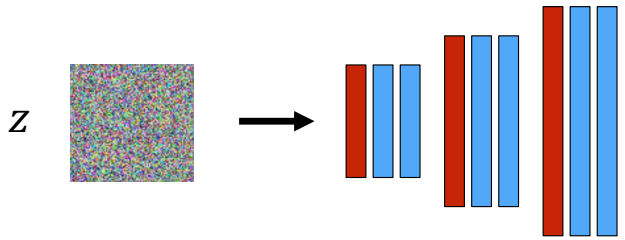
$$\mathcal{R}(x) = \begin{cases} 0 & x \in \text{range}(\mathcal{G}) \\ \infty & \text{otherwise} \end{cases}$$

- Chose $x \in \text{range}(\mathcal{G})$ that best fits the data

$$\hat{x} = \arg \min_{x \in \text{range}(\mathcal{G})} \|y - Ax\|_2^2 = \mathcal{G}(\text{z})$$

$$\hat{z} = \arg \min_z \|y - A\mathcal{G}(\text{z})\|_2^2$$

Generator $\mathcal{G}$



Solving the inverse problem: Gradient descent networks

Unrolled optimisation

Let's assume that $\mathcal{R}(x)$ is differentiable

$$\arg\min_x \mathcal{D}(Ax, y) + \lambda\mathcal{R}(x)$$

Solving the inverse problem: Gradient descent networks

Unrolled optimisation

Let's assume that $\mathcal{R}(x)$ is differentiable

$$\arg\min_x \|y - Ax\|_2^2 + \mathcal{R}(x)$$

set $\hat{x}^{(1)}$ and stepsize η

for $k = 1, 2, \cdots$

$$\hat{x}^{(k+1)} = \hat{x}^{(k)} + \eta A^T\left(y - A\hat{x}^{(k)}\right) - \boxed{\eta \nabla \mathcal{R}(\hat{x}^{(k)})}$$

replace with learned NN

Solving the inverse problem: Gradient descent networks

Unrolled optimisation

Let's assume that $\mathcal{R}(x)$ is differentiable

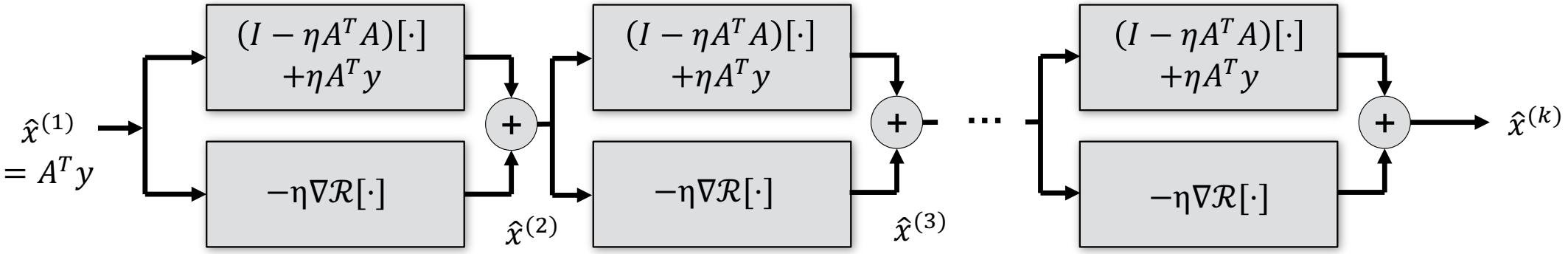
$$\arg\min_x \|y - Ax\|_2^2 + \mathcal{R}(x)$$

set $\hat{x}^{(1)}$ and stepsize η

for $k = 1, 2, \cdots$

$$\hat{x}^{(k+1)} = \hat{x}^{(k)} + \eta A^T\left(y - A\hat{x}^{(k)}\right) - \eta\nabla\mathcal{R}(\hat{x}^{(k)})$$

replace with learned NN



Solving the inverse problem: Gradient descent networks

Unrolled optimisation

Let's assume that $\mathcal{R}(x)$ is differentiable

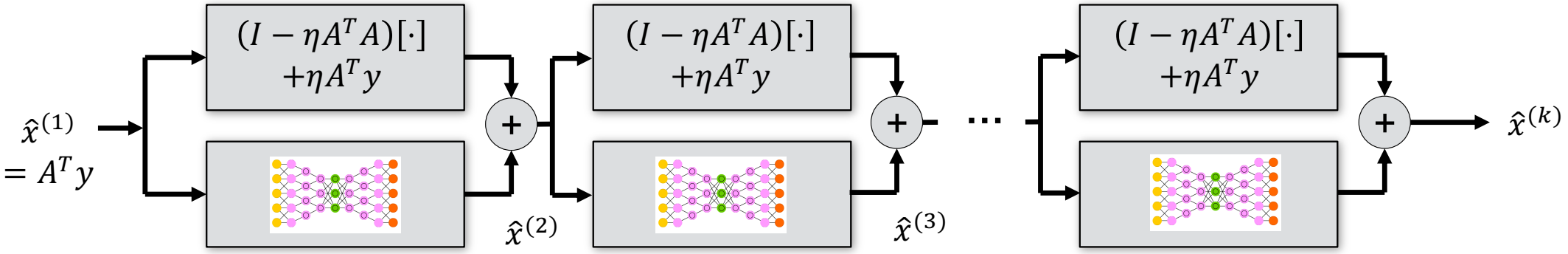
$$\arg\min_x \|y - Ax\|_2^2 + \mathcal{R}(x)$$

set $\hat{x}^{(1)}$ and stepsize η

for $k = 1, 2, \cdots$

$$\hat{x}^{(k+1)} = \hat{x}^{(k)} + \eta A^T\left(y - A\hat{x}^{(k)}\right) - \eta\nabla\mathcal{R}(\hat{x}^{(k)})$$

replace with learned NN



Unrolled optimization framework **trained end-to-end**

Imperial College London

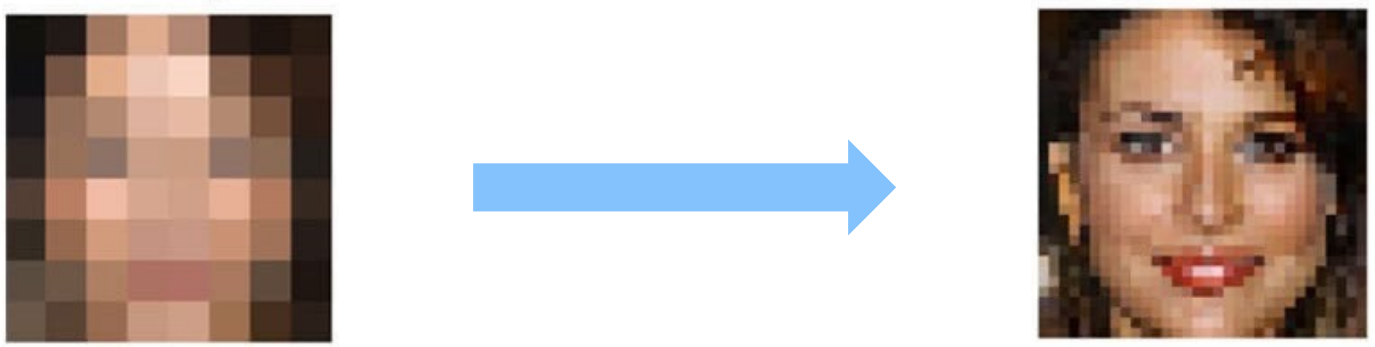
Imperial College London

Deep Learning for Image Super-Resolution

Daniel Rueckert
Department of Computing
Imperial College London, UK

Problem formulation

- Upsample low-resolution (LR) image to high-resolution (HR or SR) image



- Forward model (going from high- to low-resolution) is straightforward and involves some image degradation followed by downsampling
- Inverse model: e.g. interpolation-based models

Super-resolution framework: Post-upsampling super-resolution

- Directly upsamples LR image into SR
- Requires learnable upsampling layers

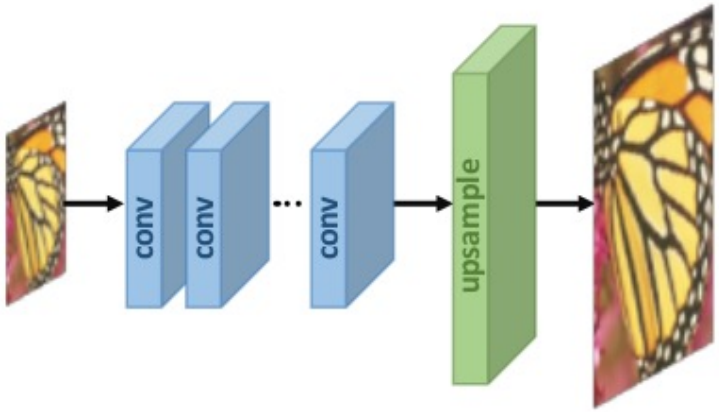


Figure from: Deep Learning for Image Super-resolution: A Survey
https://arxiv.org/abs/1902.06068

- Advantages:
  - Fast and low memory requirements
- Disadvantages:
  - Network has to learn entire upsampling pipeline
  - Network typically limited to a specific up-sampling factor

Super-resolution framework: Pre-upsampling super-resolution

- Two stage process:
  1. First use traditional upsampling algorithm (e.g. linear interpolation) to obtain SR images
  2. Then refining upsampled using a deep neural network (usually a CNN)

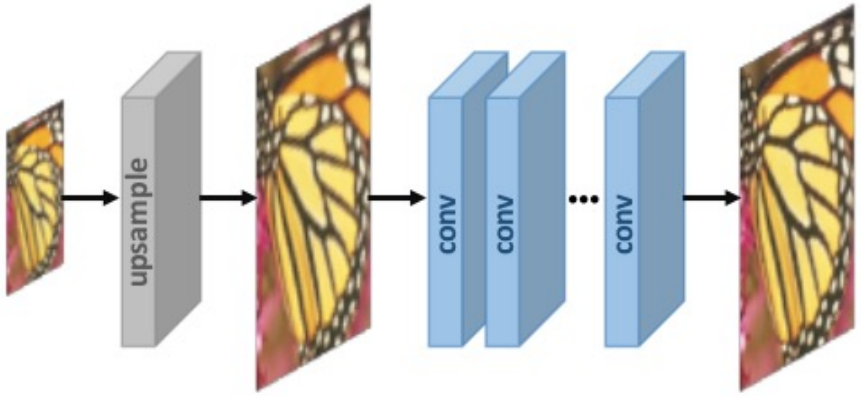


Figure from: Deep Learning for Image Super-resolution: A Survey
https://arxiv.org/abs/1902.06068

- Advantages:
  - Upsampling operation is performed using interpolation, then correct smaller details
  - Can be applied to a range of upscaling factors and image sizes
- Disadvantages:
  - Operates on SR image, thus requires more computation and memory

Super-resolution framework: Progressive upsampling super-resolution

- Multi-stage process:
  - Use a cascade of CNNs to progressively reconstruct higher-resolution images.
  - At each stage, the images are upsampled to higher resolution and refined by CNNs

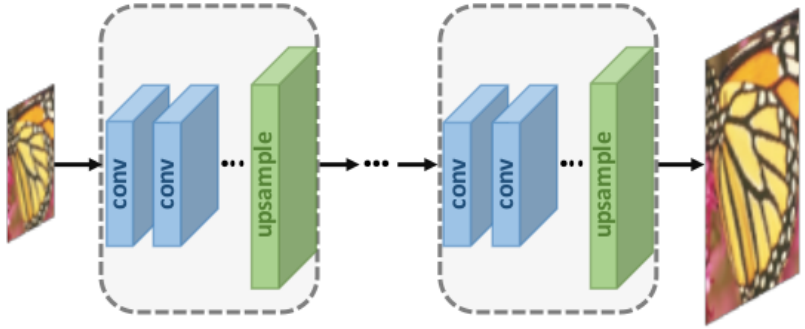


Figure from: Deep Learning for Image Super-resolution: A Survey
https://arxiv.org/abs/1902.06068

- Advantages:
  - Decomposes complex task into simple tasks
  - Reasonable efficiency
- Disadvantages:
  - Sometimes difficult to train very deep models

Super-resolution framework: Iterative up-and-down sampling super-resolution

- Approach:
  - Alternate between upsampling and downsampling (back-projection) operations
  - Mutually connected up- and down-sampling stages

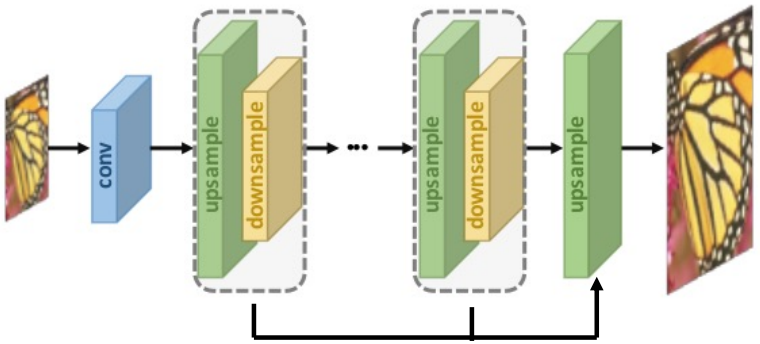


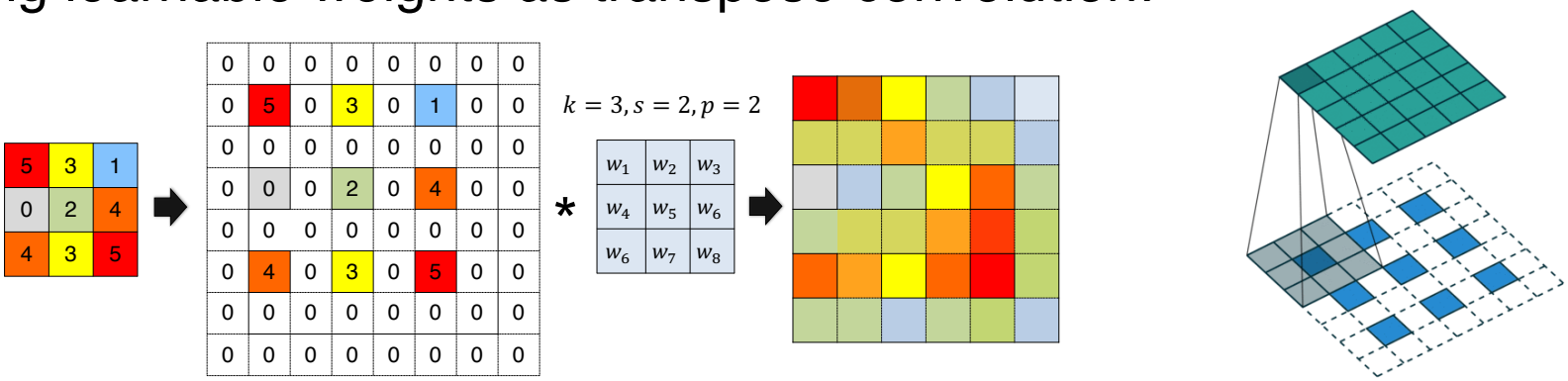
Figure from: Deep Learning for Image Super-resolution: A Survey
https://arxiv.org/abs/1902.06068

- Advantages:
  - Has shown superior performance as it allows error feedback
  - Easier training of deep networks

How to implement upsampling?

- Traditional interpolation-based upsampling (NN, bi-linear) can be implemented as a convolutional layer with fixed weights:

- Or using learnable weights as transpose convolution:



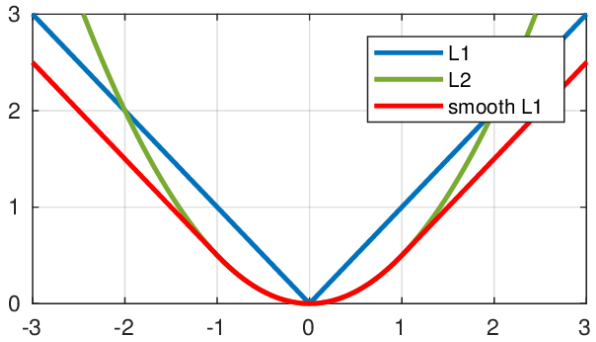
Loss functions for super-resolution

- Pixel-wise loss function (either L1 or L2)

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} |x_i - \hat{x}_i|^p$$

- Alternative: Huber loss function

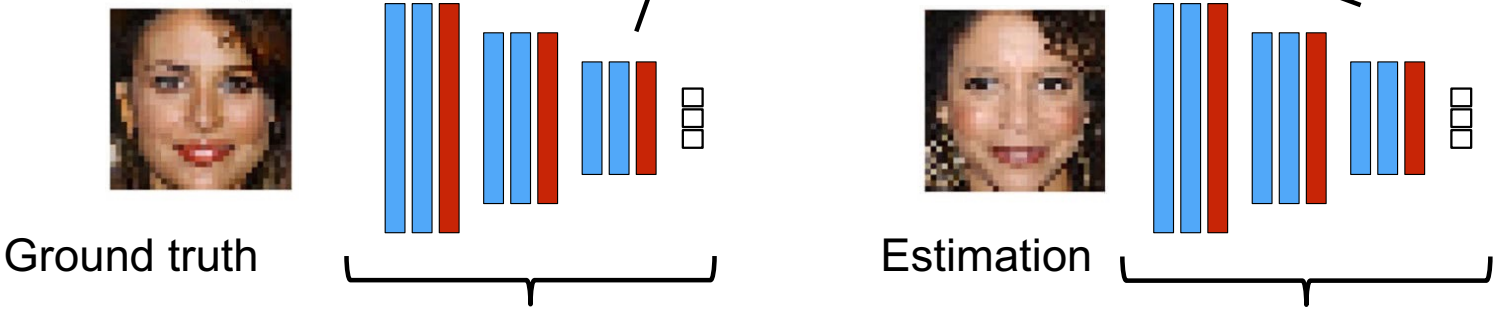
$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \delta(x_i - \hat{x}_i) \qquad \delta(a) = \begin{cases} 0.5a^2 & \text{for } |a| \leq 1 \\ |a| - 0.5 & \text{otherwise} \end{cases}$$



Loss functions for super-resolution

- Perceptual loss: Computes loss on the output θ of an intermediate layer l of a pre-trained network:

$$\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\left(\Theta_i^{(l)}(x) - \Theta_i^{(l)}(\hat{x})\right)^2$$



Loss functions for super-resolution

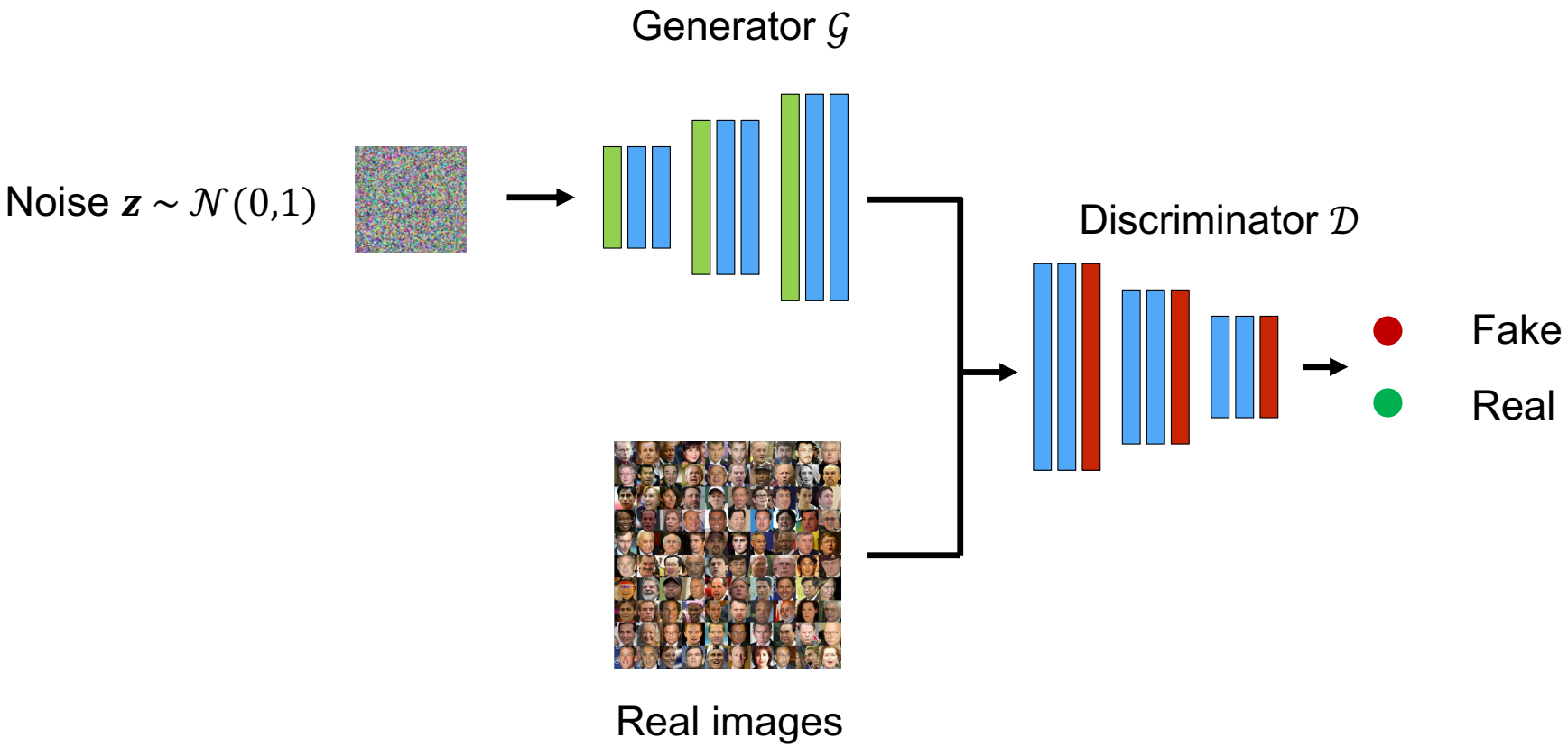
- Total variation

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\sum_{d}\left(|\nabla x|_i^{(d)}\right)^2}$$

- Assumption: Absolute value of gradient of the image is low, e.g. image is piecewise constant
- Implementation for 2D image (using forward differences to approximate the gradient):

$$\mathcal{L} = \frac{1}{n_1 n_2}\sum_{i=1}^{n_1-1}\sum_{j=1}^{n_2-1}\sqrt{\left|x_{i-1,j} - x_{i,j}\right|^2 + \left|x_{i,j-1} - x_{i,j}\right|^2}$$

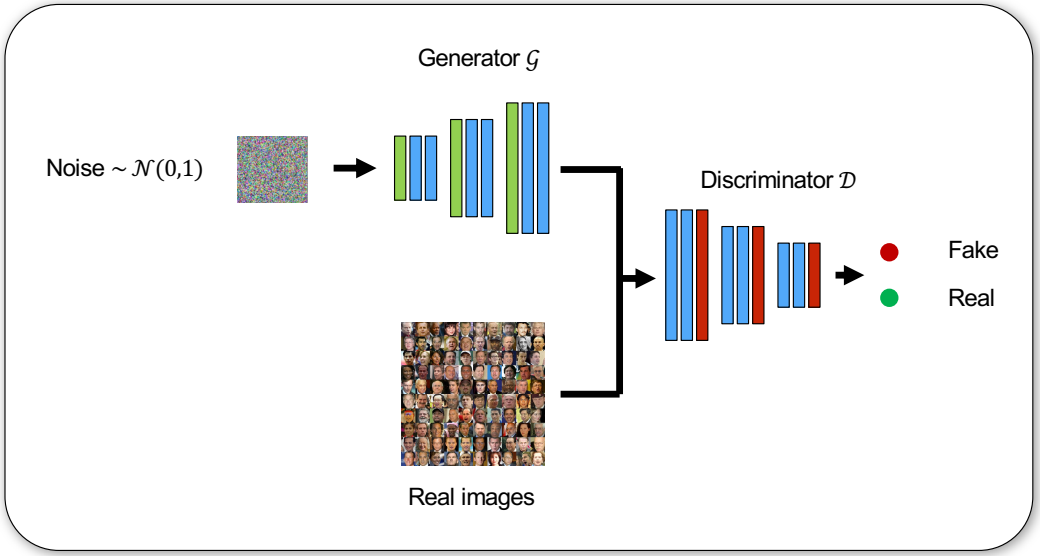
Generative adversarial networks (GANs)



I. Goodfellow, NIPS, 2014

Generative adversarial networks (GANs)

- $\mathcal{D}(x)$ represents the probability that x came from the real data rather than from $\mathcal{G}$.
- $\mathcal{D}$ is trained to maximize the probability of assigning the correct label to both real examples and samples from $\mathcal{G}$.



- Simultaneously train $\mathcal{G}$ to minimize $\log(1 - \mathcal{D}(\mathcal{G}(\boldsymbol{z}))$, e.g. $\mathcal{D}$ and $\mathcal{G}$ play a two-player minimax game with value function $V(\mathcal{G}, \mathcal{D})$:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} V(\mathcal{G}, \mathcal{D}) = \underbrace{\mathbb{E}_{x \sim p_{data}(x)}[\log \mathcal{D}(x)]}_{\text{real}} + \underbrace{\mathbb{E}_{z \sim p_z(z)}[\log(1 - \mathcal{D}(\mathcal{G}(\boldsymbol{z}))]}_{\text{synthetic (aka “fake”)}}$$

Loss functions for super-resolution based on GANs

- Minimax game for super-resolution:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \mathbb{E}_{I_{HR} \sim p_{train}} [\log \mathcal{D}(I_{HR})] + \mathbb{E}_{I_{LR} \sim P_{\mathcal{G}}(I_{LR})} [\log(1 - \mathcal{D}(\mathcal{G}(I_{LR}))]$$

- Aim: To detect
  - Which images are (real) high-resolution?
  - Which images are super-resolved?

- Use discriminator output as loss function:

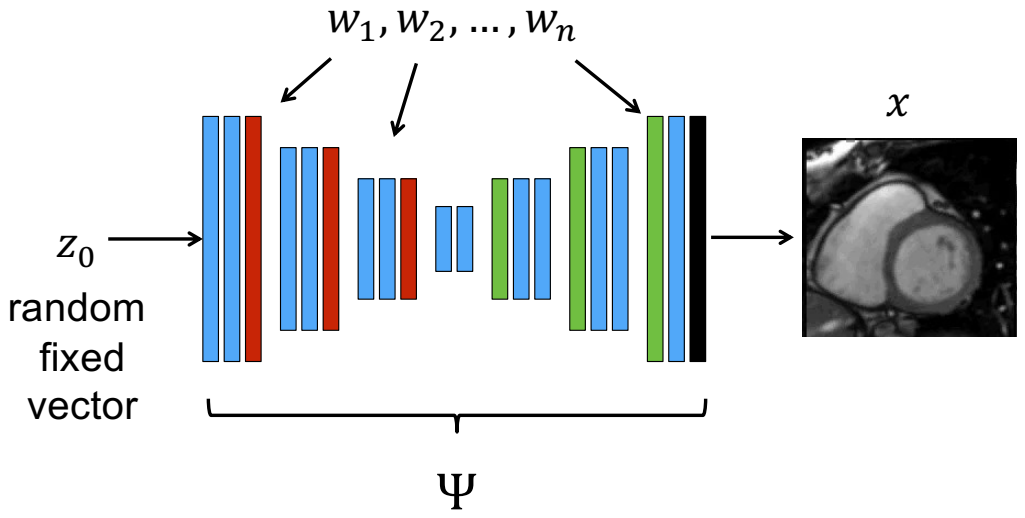
$$\mathcal{L} = -\log \mathcal{D}(\mathcal{G}(I_{LR}))$$

Parameterizing images via generative networks

- Consider a generator network Ψ with a fixed input z_0

- The network parameters w can be thought as a ***parameterization*** of images

$$w \longmapsto x = \Psi(z_0; w)$$



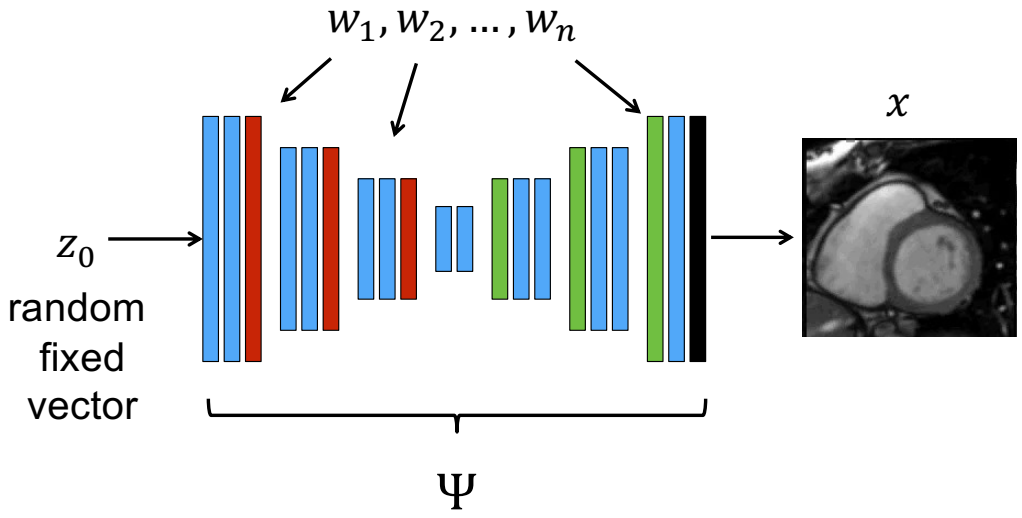
Parameterizing images via generative networks

Fit a network to a single image:

- Start with a random-initialised network
- Given an image x, its parameters w are recovered by solving the following optimization problem:

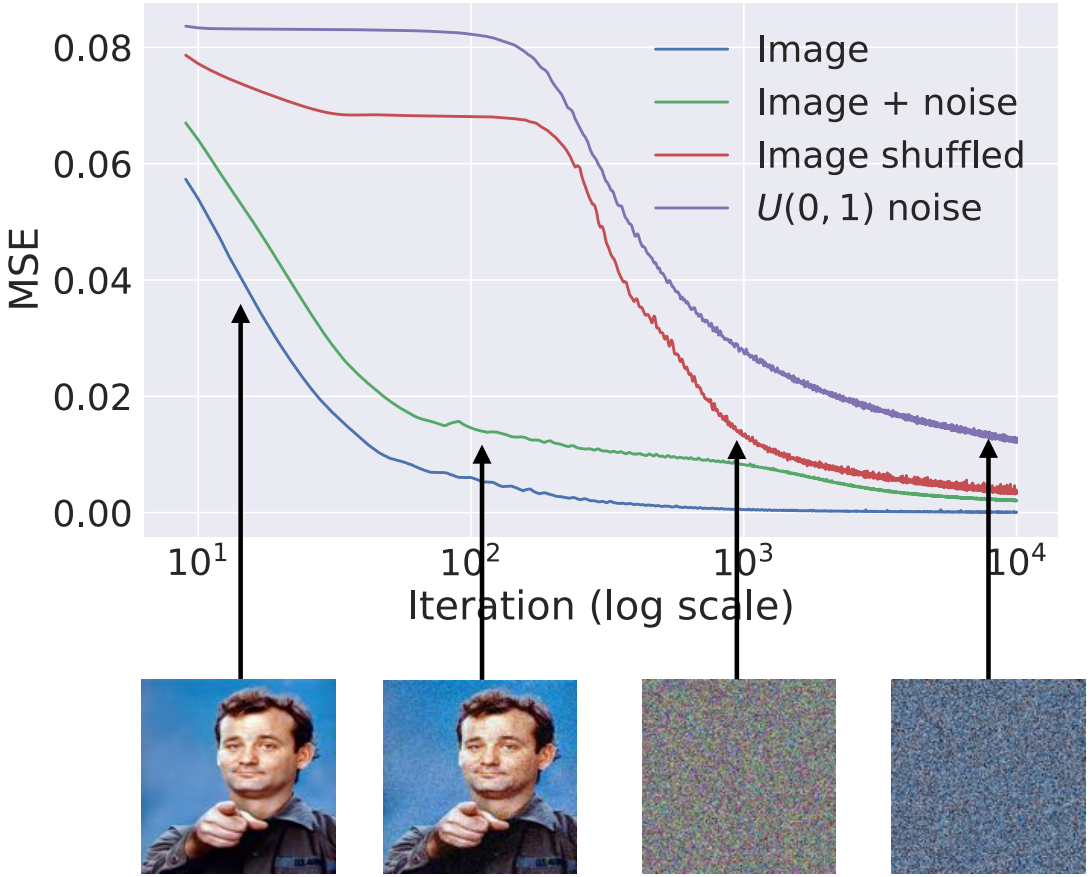
$$\min_{w} \|x - \Psi(z_0; w)\|^2$$

- This is similar to learning a network from a single image



Deep image prior

For most generative networks fitting naturally looking images is easier/faster than fitting others



Deep Image Prior. Ulyanov, Vedaldi & Lempistky, CVPR 2018

Deep image prior: Application to inpainting

For inpainting we only reconstruct the visible pixels, implicitly inferring the pixels that are masked out by mask m:

$$\min_{w} \left\| m \odot \left(x - \Psi(z_0; w) \right) \right\|^2$$

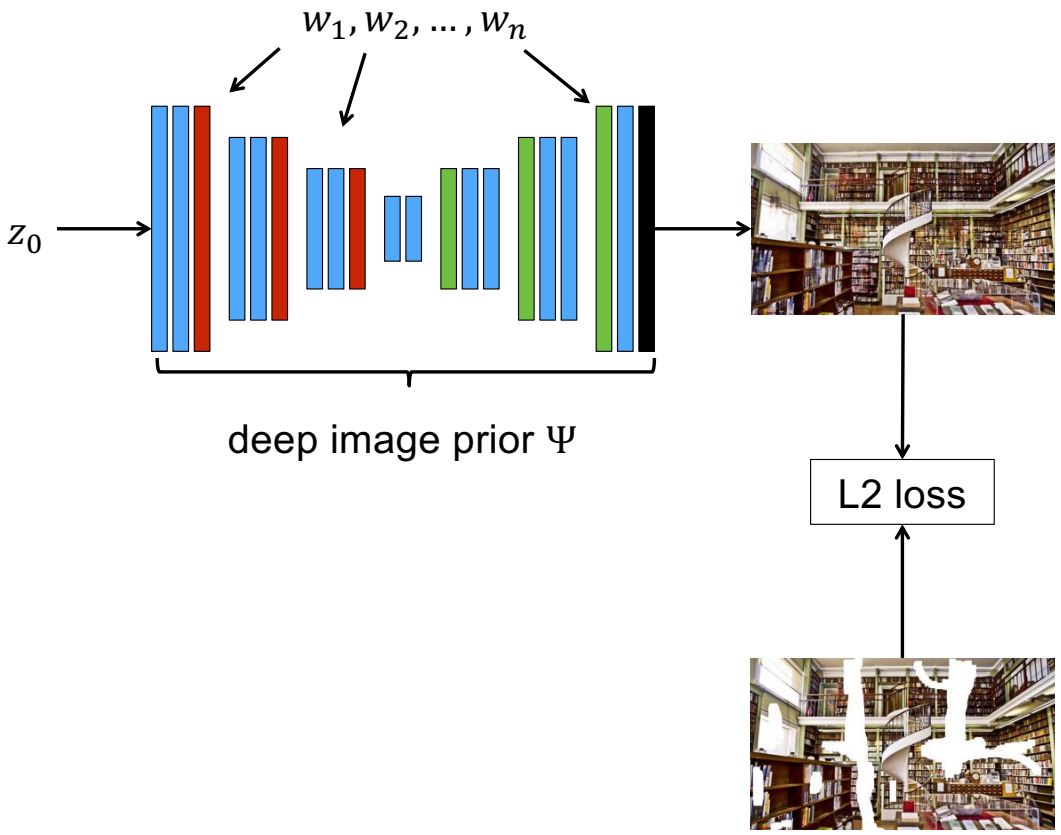
Conventional inpainting

Inpainting using deep image prior

Deep image prior: Application to inpainting

Deep Image Prior. Ulyanov, Vedaldi & Lempistky, CVPR 2018

Inpainting via deep image priors



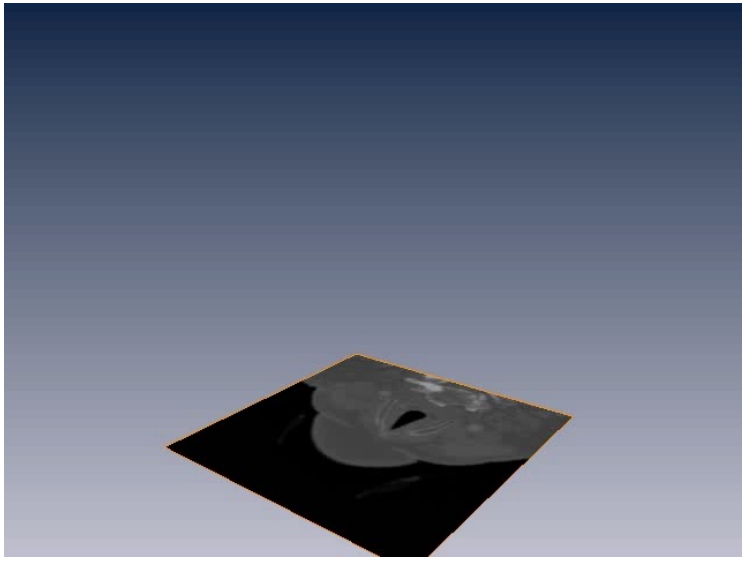
Imperial College London

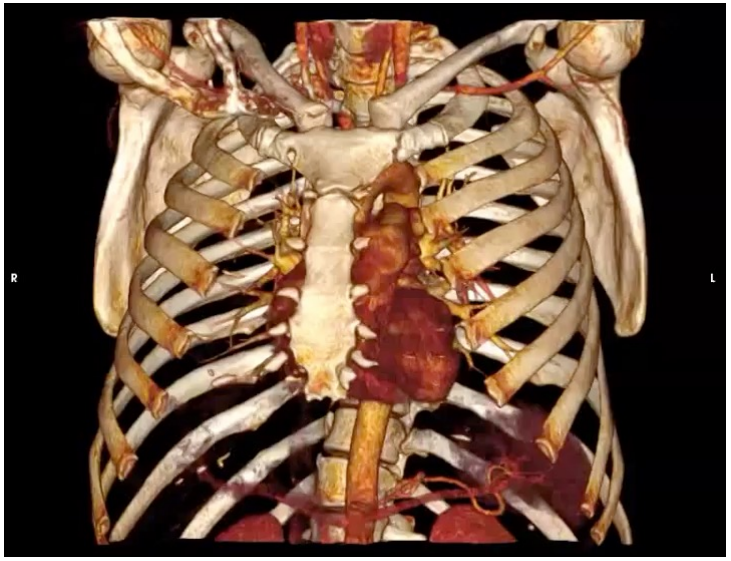
Imperial College London

Deep Learning for Image Reconstruction

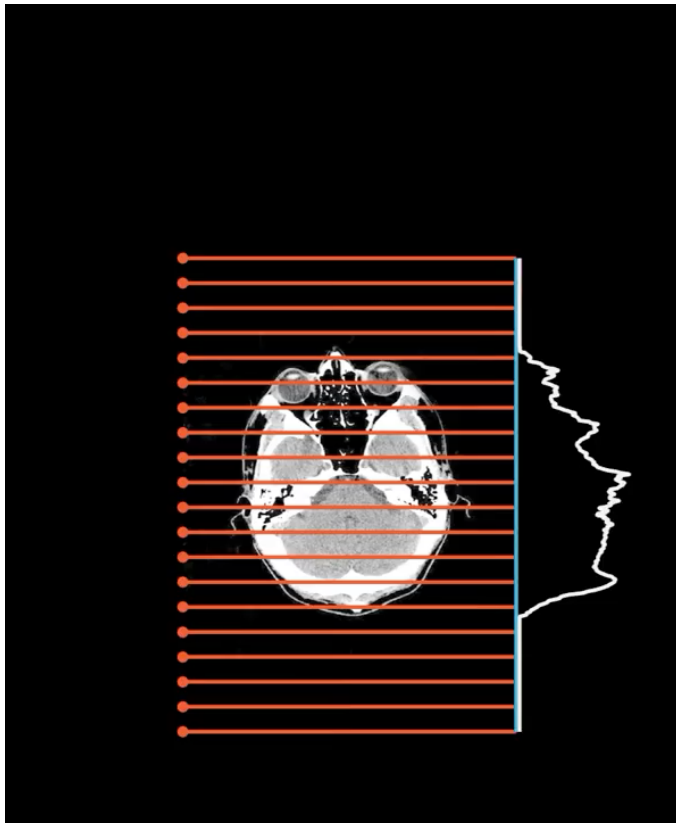
Daniel Rueckert
Department of Computing
Imperial College London, UK

Medical Imaging: X-ray computed tomography (CT)





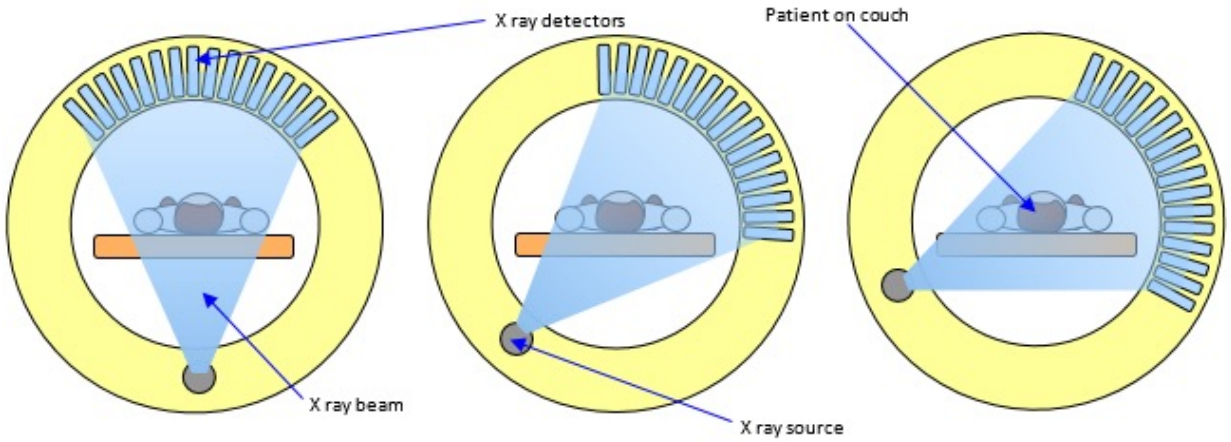
Medical Imaging: X-ray computed tomography (CT)



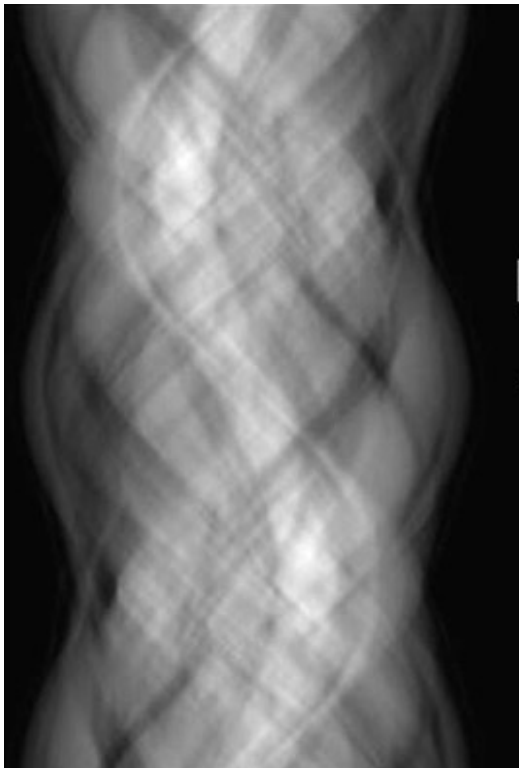
source: https://youtu.be/q7Rt_OY_7tU

Medical Imaging: X-ray computed tomography (CT)

- CT imaging
  - high contrast
  - high spatial resolution
  - fast acquisition
  - but ionising radiation



Medical Imaging: X-ray computed tomography (CT)

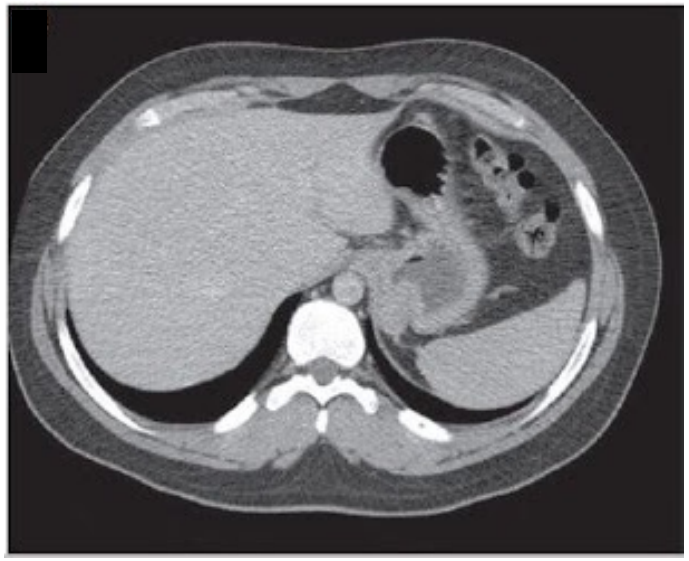


Sinogram

Forward problem
(Radon transform)

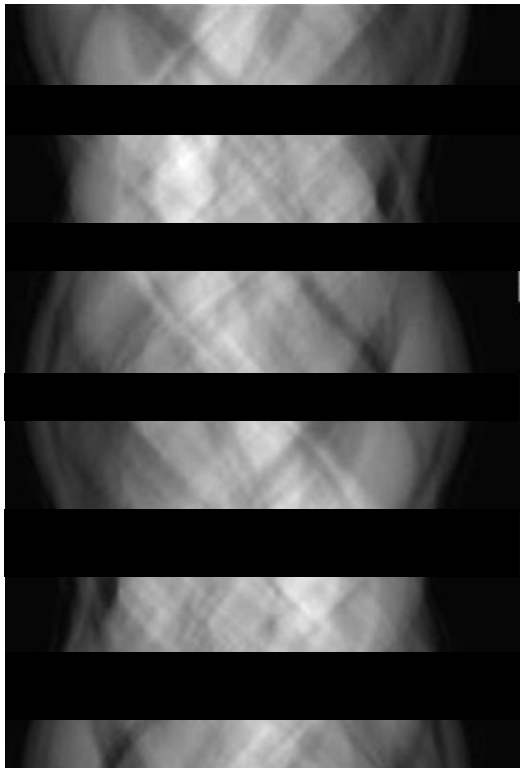
Inverse problem:

Fully sampled
reconstruction



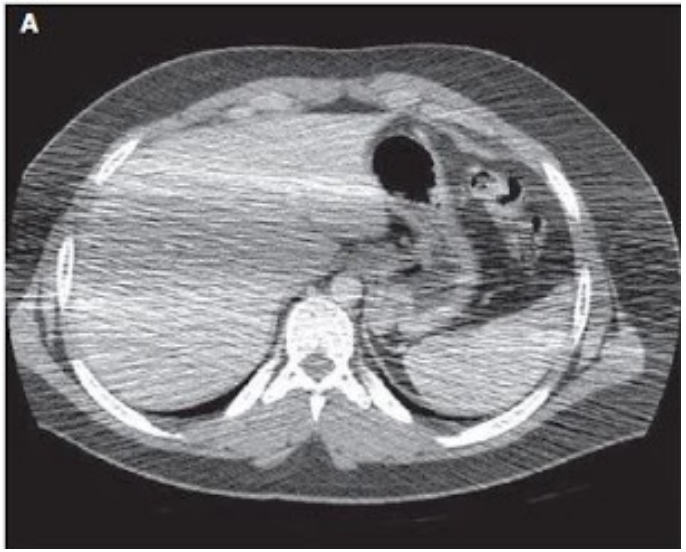
CT image

Medical Imaging: X-ray computed tomography (CT)



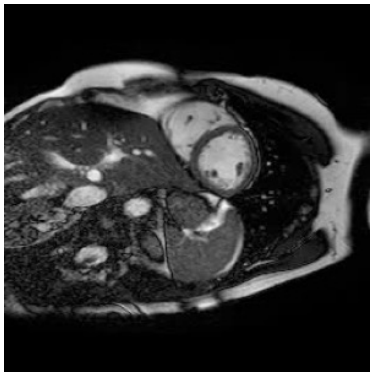
Sinogram

Under-sampled reconstruction

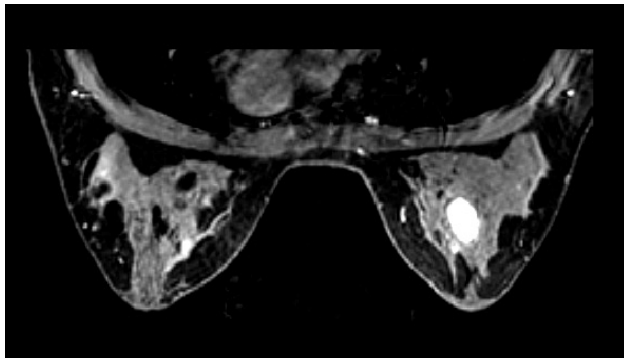


CT image

Medical Imaging: Magnetic Resonance (MR)

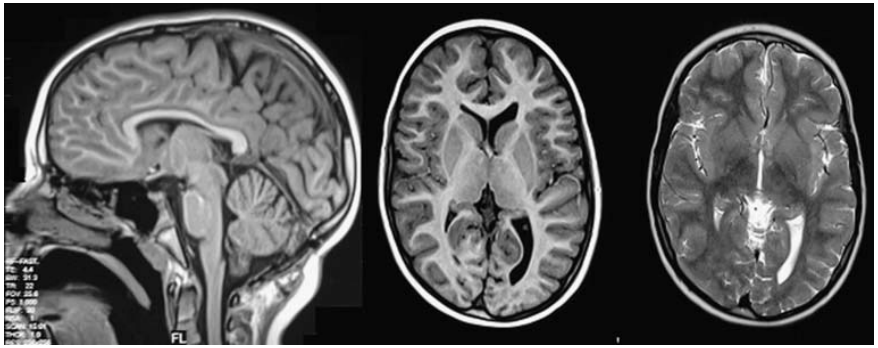


heart



cancer

Whole body



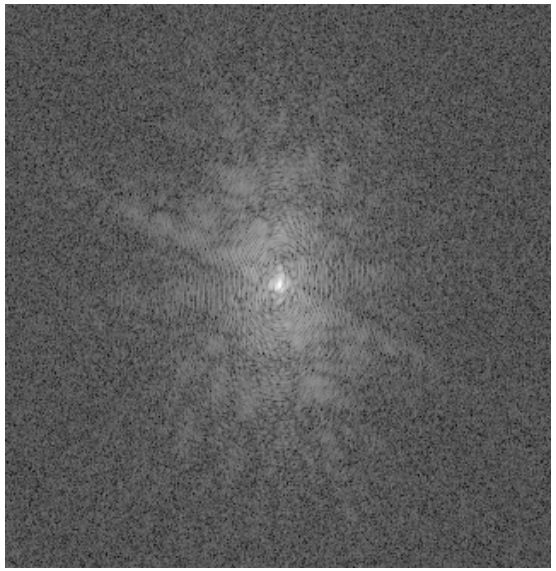
brain

lung

Medical Imaging: Magnetic Resonance (MR)

- MR imaging:
  - high contrast ✓
  - high spatial resolution ✓
  - no ionising radiation ✓
  - but slow acquisition process 👎
- Slow acquisition is
  - ok for static objects (e.g. brain, bones, etc)
  - problematic for moving objects (e.g. heart, liver, fetus)
- Options for MR acquisition:
  - real-time MR: fast, but 2D and relatively poor image quality
  - gated MRI fine for period motion, e.g. respiration or cardiac motion but requires gating (ECG or navigators) leading to long acquisition times (30-90 min).

Medical Imaging: Magnetic Resonance (MR)

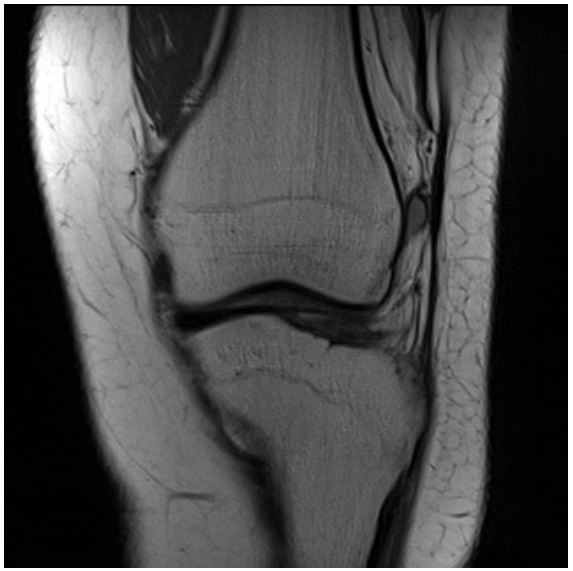


K-Space

Forward problem
(Fourier transform)

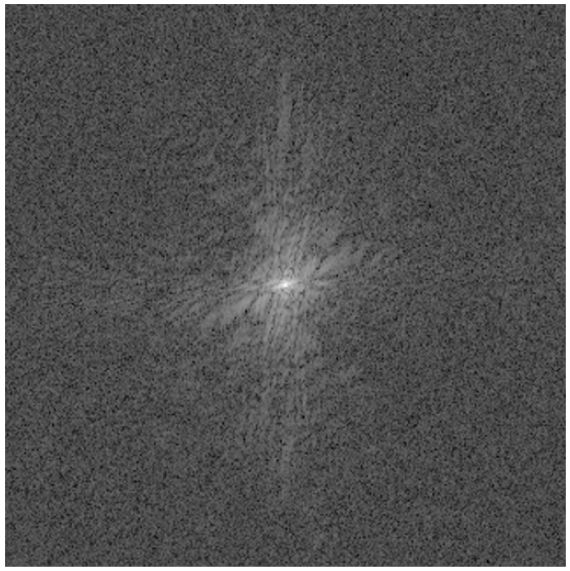
Inverse problem:

Fully sampled
reconstruction



MR image

Medical Imaging: Magnetic Resonance (MR)



Forward problem
(Fourier transform)

Inverse problem:

Fully sampled
reconstruction

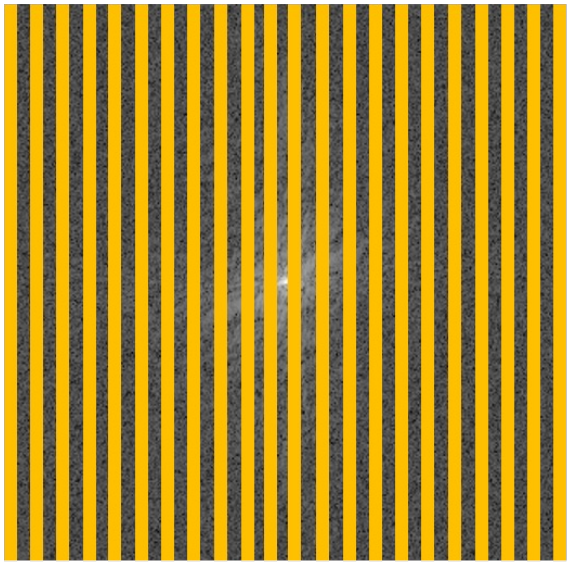
K-Space

MR image

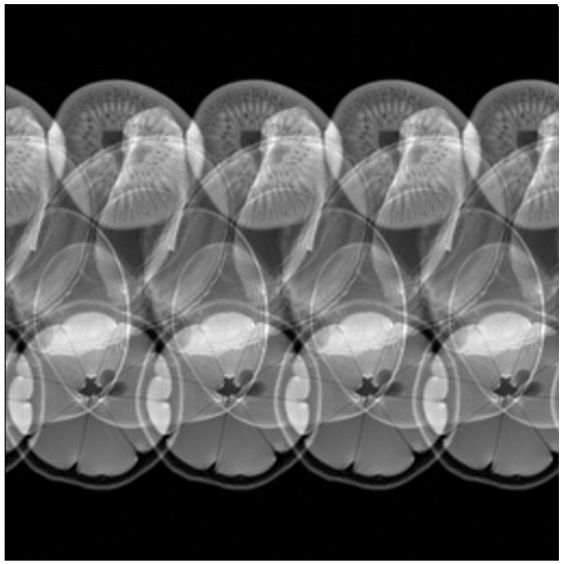
Recover an image $x \in \mathbb{K}^{N_x}$ from a set of observations $y \in \mathbb{K}^{N_y}$ which are corrupted by noise $n \in \mathbb{K}^{N_y}$, and A: $\mathbb{K}^{N_x} \to \mathbb{K}^{N_y}$ is a linear operator

$$y = Ax + n$$

Medical Imaging: Magnetic Resonance (MR)



Inverse problem



Undersampled K-Space

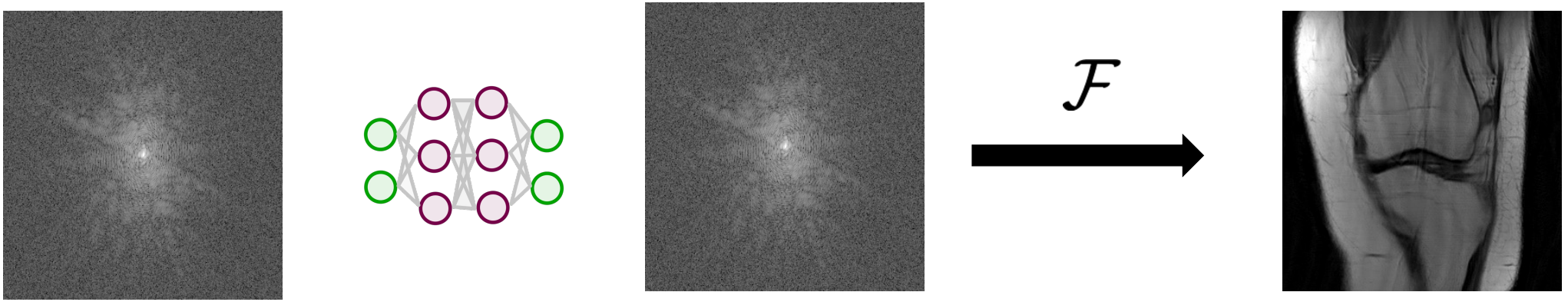
Aliased MR image

Recover an image $x \in \mathbb{K}^{N_x}$ from a set of observations $y \in \mathbb{K}^{N_y}$ which are corrupted by noise $n \in \mathbb{K}^{N_y}$, and A: $\mathbb{K}^{N_x} \rightarrow \mathbb{K}^{N_y}$ is a linear operator

$$y = Ax + n$$

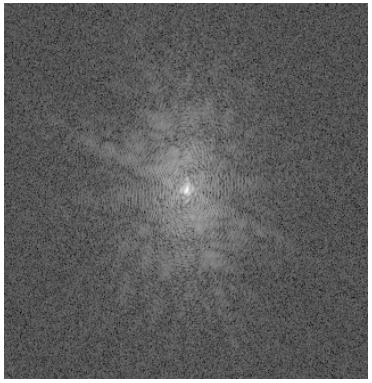
Possible reconstruction approaches: Image domain

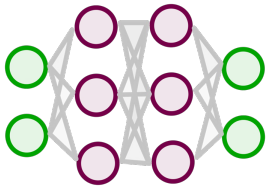
Possible reconstruction approaches: k-space domain

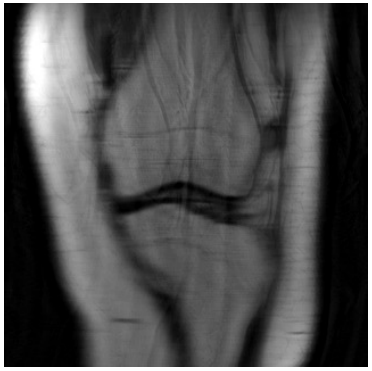


Training Phase

- Optimize θ of the model f_θ using the training set

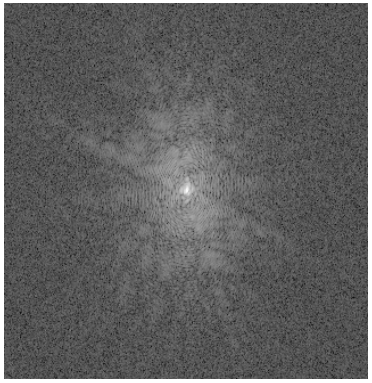


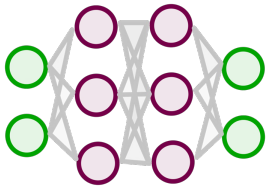


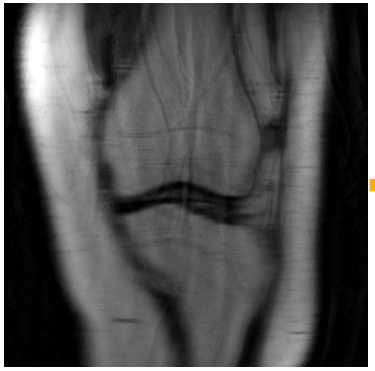


Training Phase

- Optimize θ of the model f_θ using the training set



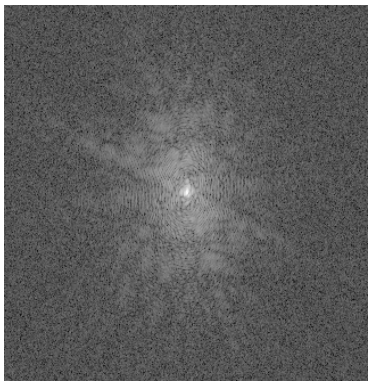


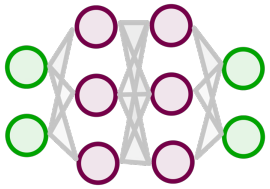


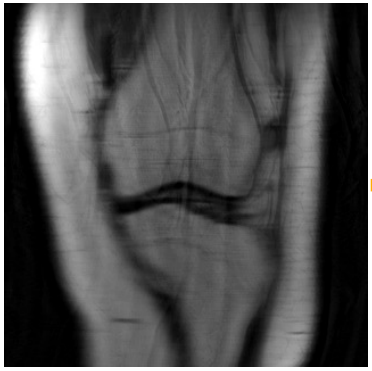
Cost Function

Training Phase

- Optimize θ of the model f_θ using the training set





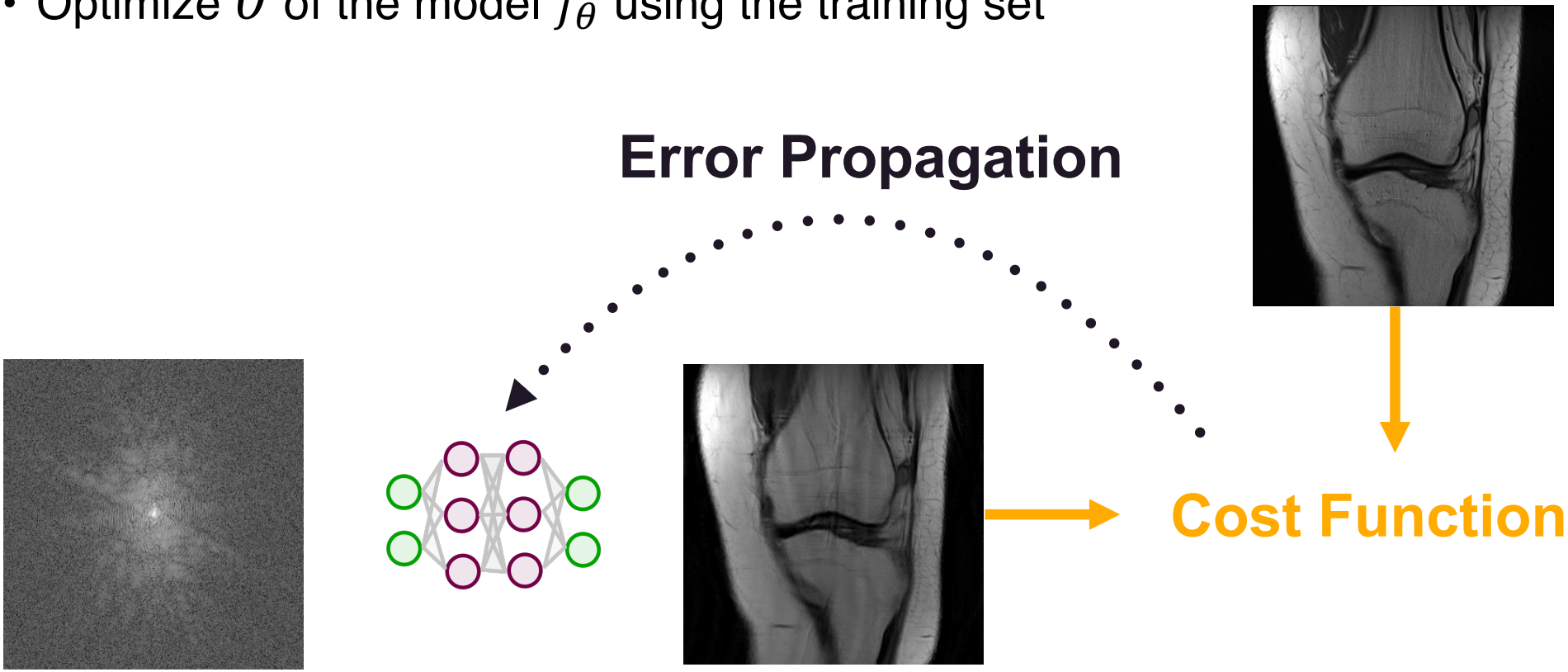


Error Propagation

Cost Function

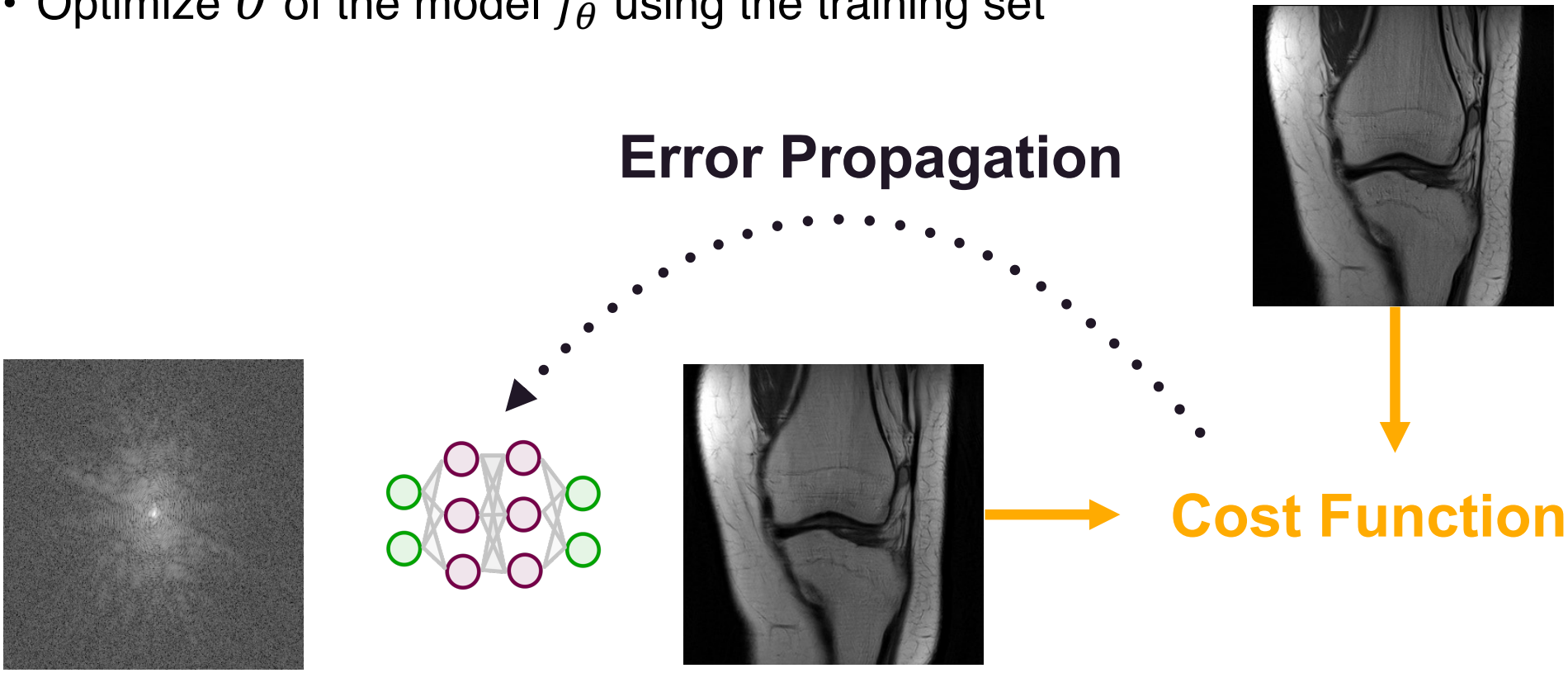
Training Phase

- Optimize θ of the model f_θ using the training set



Training Phase

- Optimize θ of the model f_θ using the training set



Cost Function

Synonym: Loss function, error function, similarity measure

- Quantifies how well the model prediction matches targets
- Needs to be selected according to the underlying task
- Is optimized during training → needs to be differentiable!

Cost Function

Synonym: Loss function, error function, similarity measure

- Quantifies how well the model prediction matches targets
- Needs to be selected according to the underlying task
- Is optimized during training → needs to be differentiable!

Example for image reconstruction: Mean Squared Error

$$MSE(y, \hat{y}) = \frac{1}{N_S} \sum_{n=1}^{N_S} \| y_n - \hat{y}_n \|_2^2$$

Training Phase

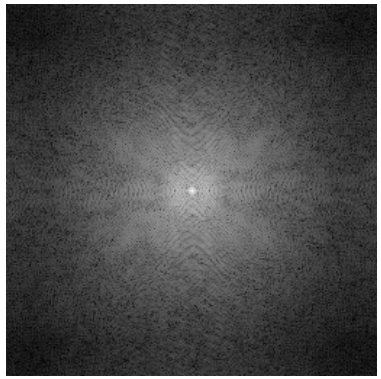
Learning model: $\widehat{\boldsymbol{y}}_{\boldsymbol{n}} = \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_{\boldsymbol{n}})$

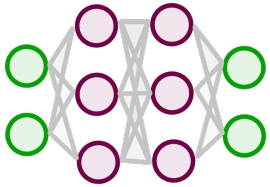
- The parameters θ of the mapping function f_θ are optimized under a cost function
- The cost function quantifies how well $\hat{y}_n = f_\theta(x_n)$ is predicted given x_n
- The parameters θ by minimizing the cost function $\mathcal{L}$ with learning rate τ

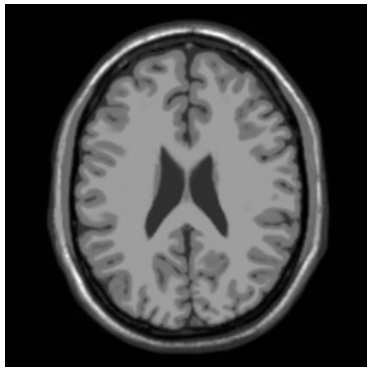
$$\theta^{(k+1)} = \theta^{(k)} - \tau \frac{\partial \mathcal{L}(\theta, x)}{\partial \theta}\Big|_{\theta=\theta^{(k)}}$$

Testing Phase (Inference)

- Apply f_θ using the optimized θ to the test set

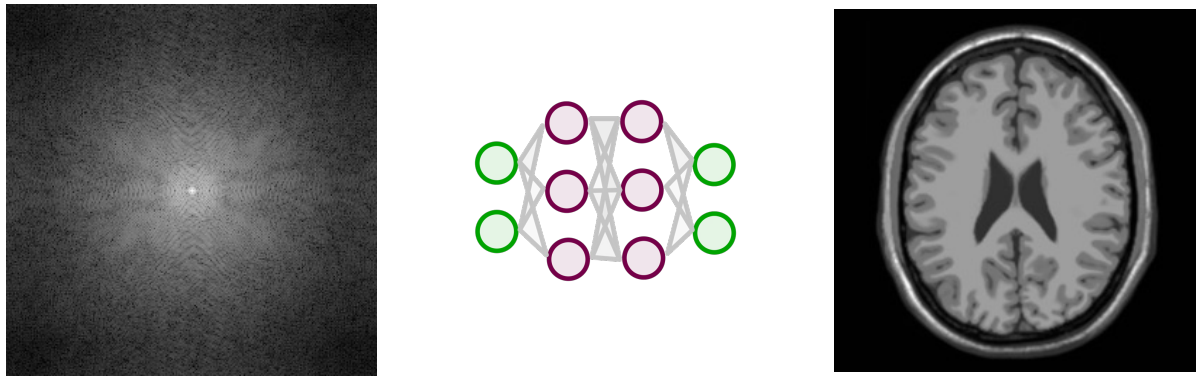






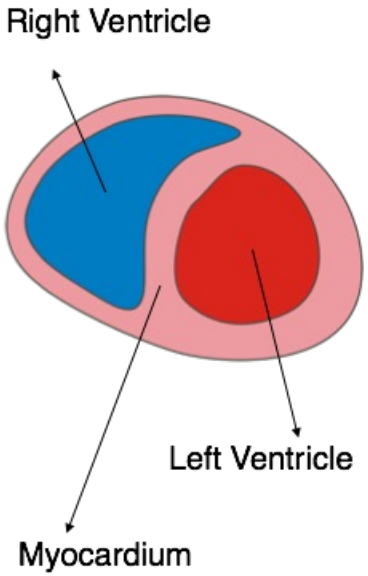
Testing Phase (Inference)

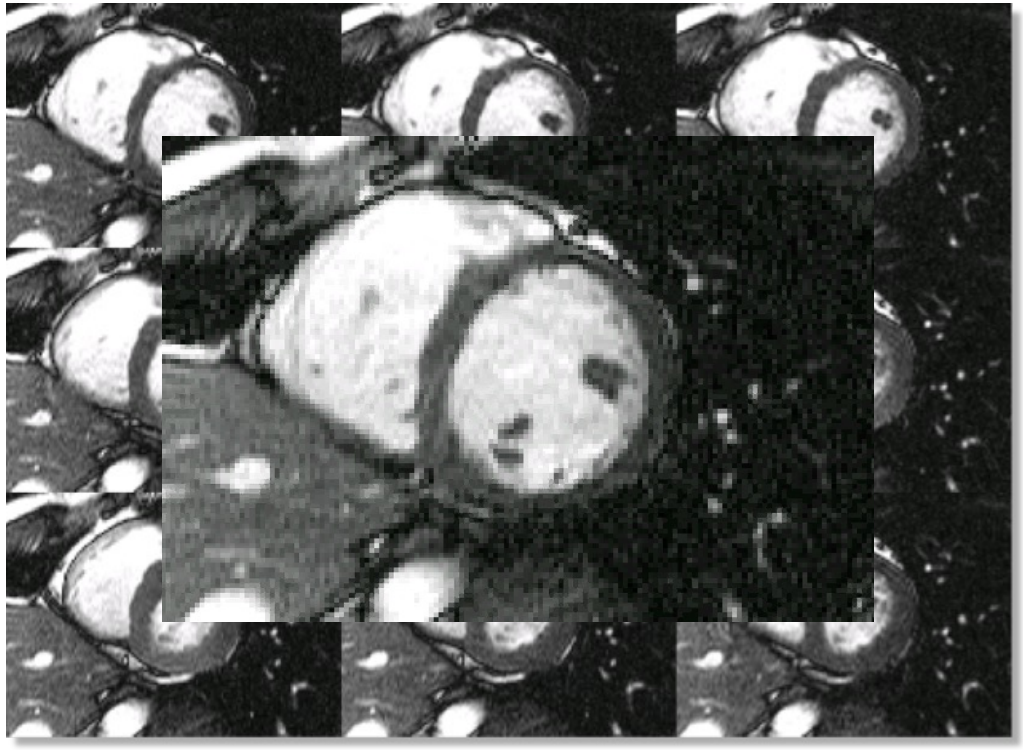
- Apply f_θ using the optimized θ to the test set



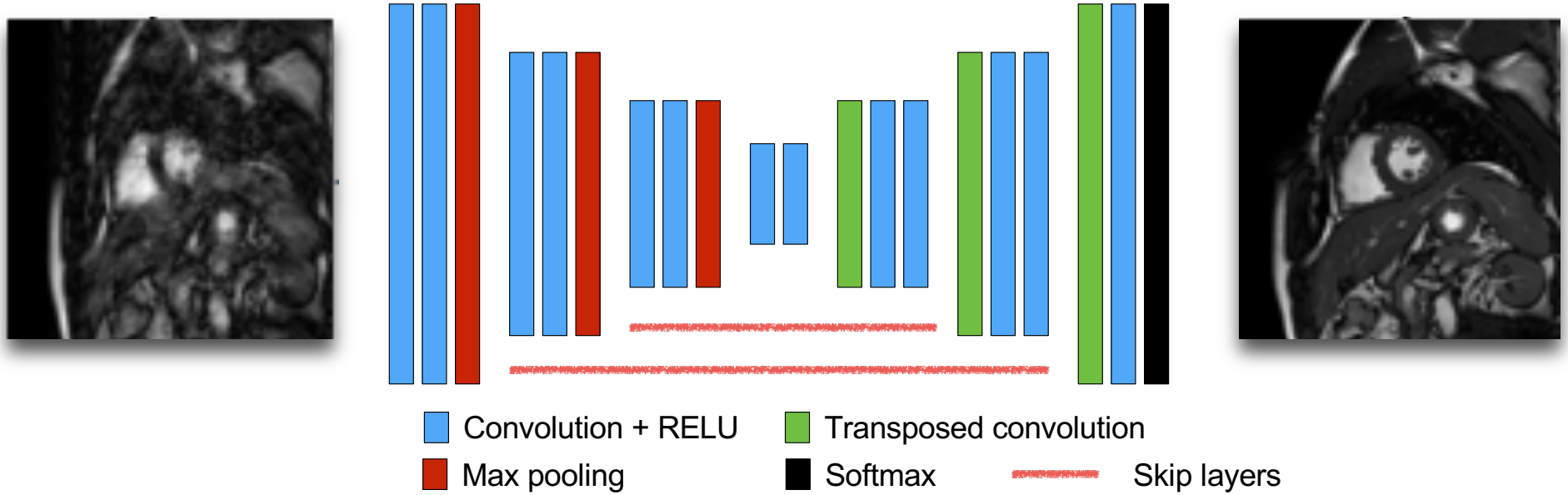
Generalization: Ability to correctly predict unseen examples

Magnetic Resonance (MR)

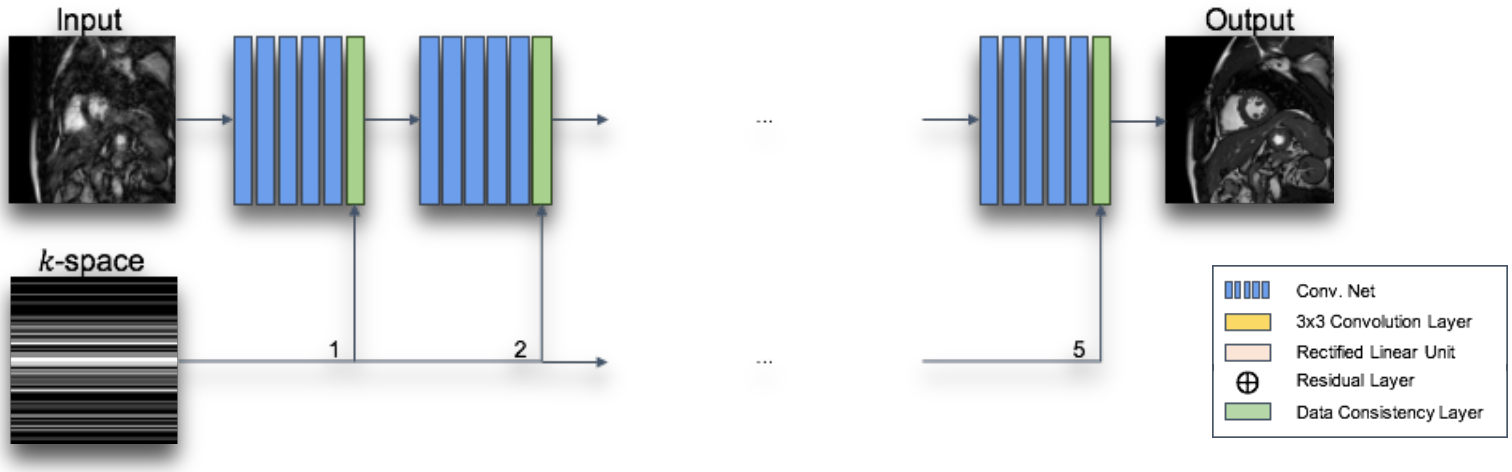




Deep learning for image reconstruction

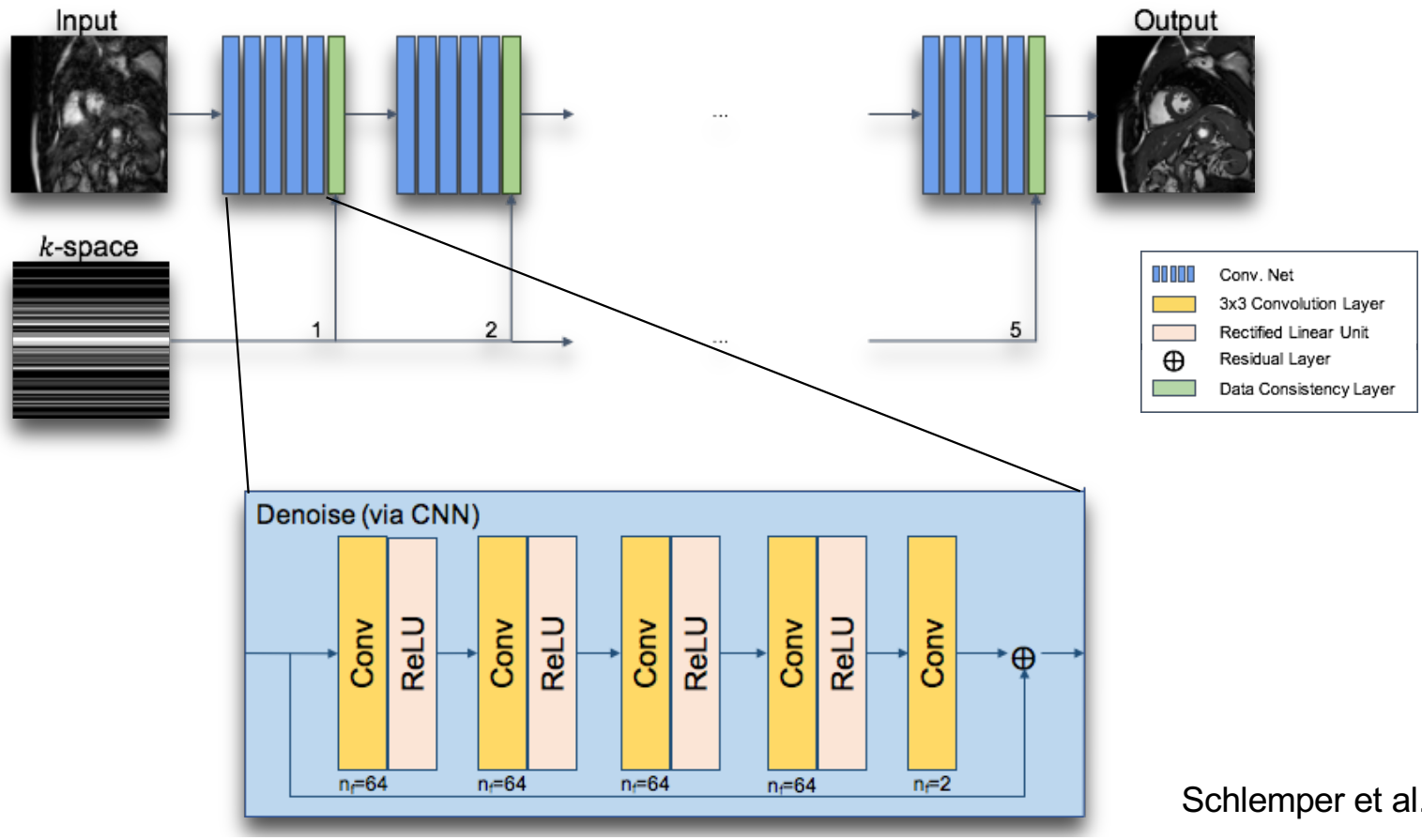


Deep learning for image reconstruction



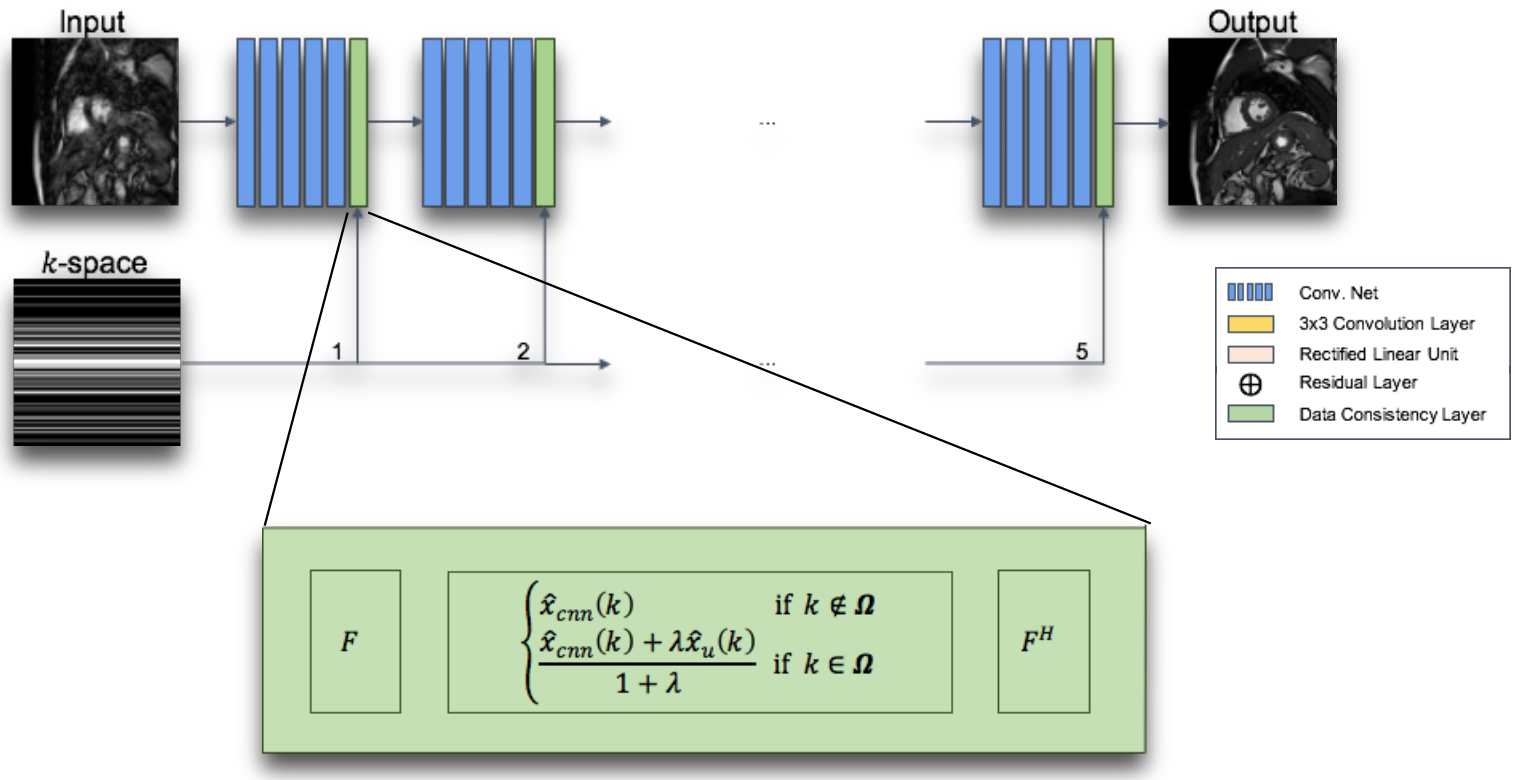
Schlemper et al. IEEE TMI 2017

Deep learning for image reconstruction



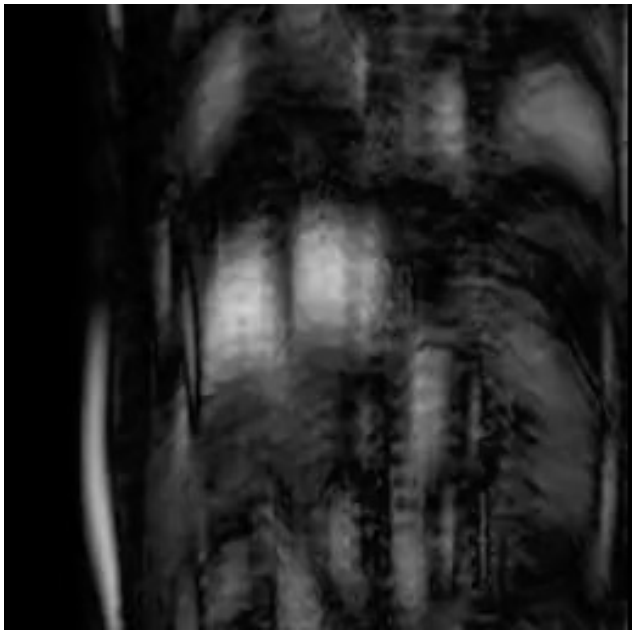
Schlemper et al. IEEE TMI 2017

Deep learning for image reconstruction



Schlemper et al. IEEE TMI 2017

Magnitude reconstruction (6-fold)



(a) 6x Undersampled

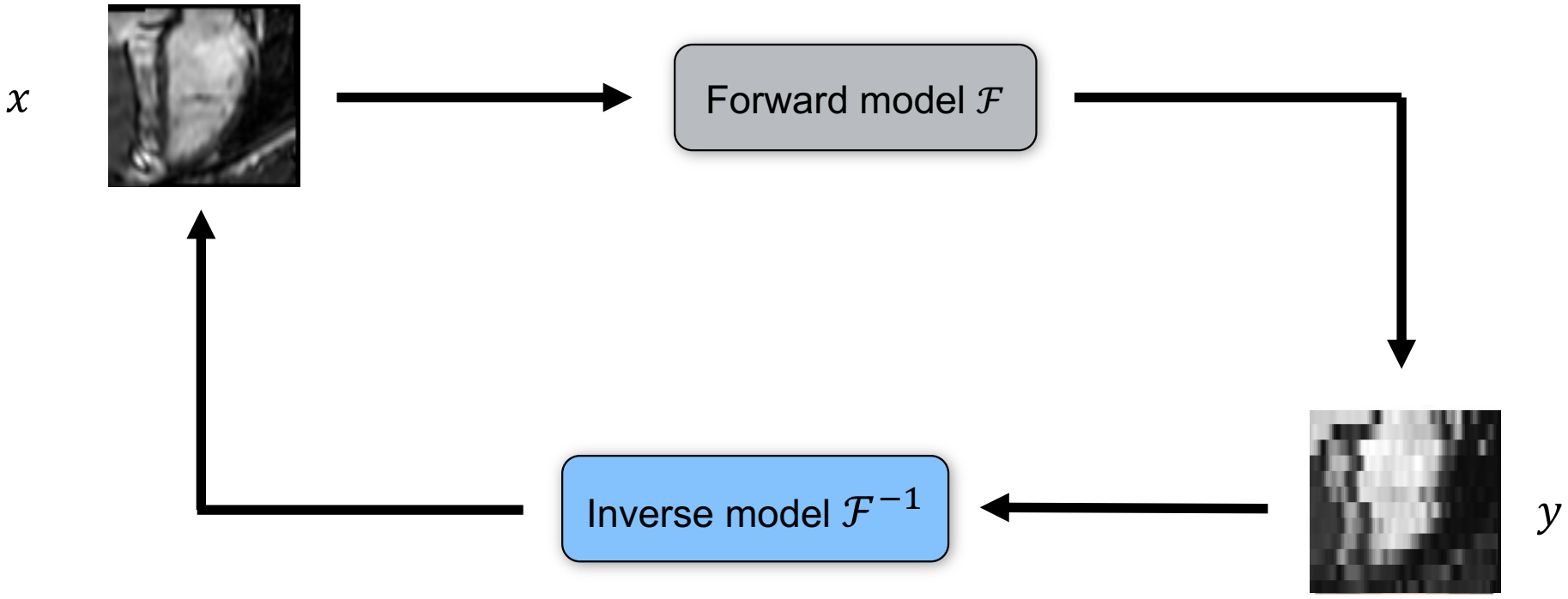
Schlemper et al. IEEE TMI 2017

Magnitude reconstruction (11-fold)

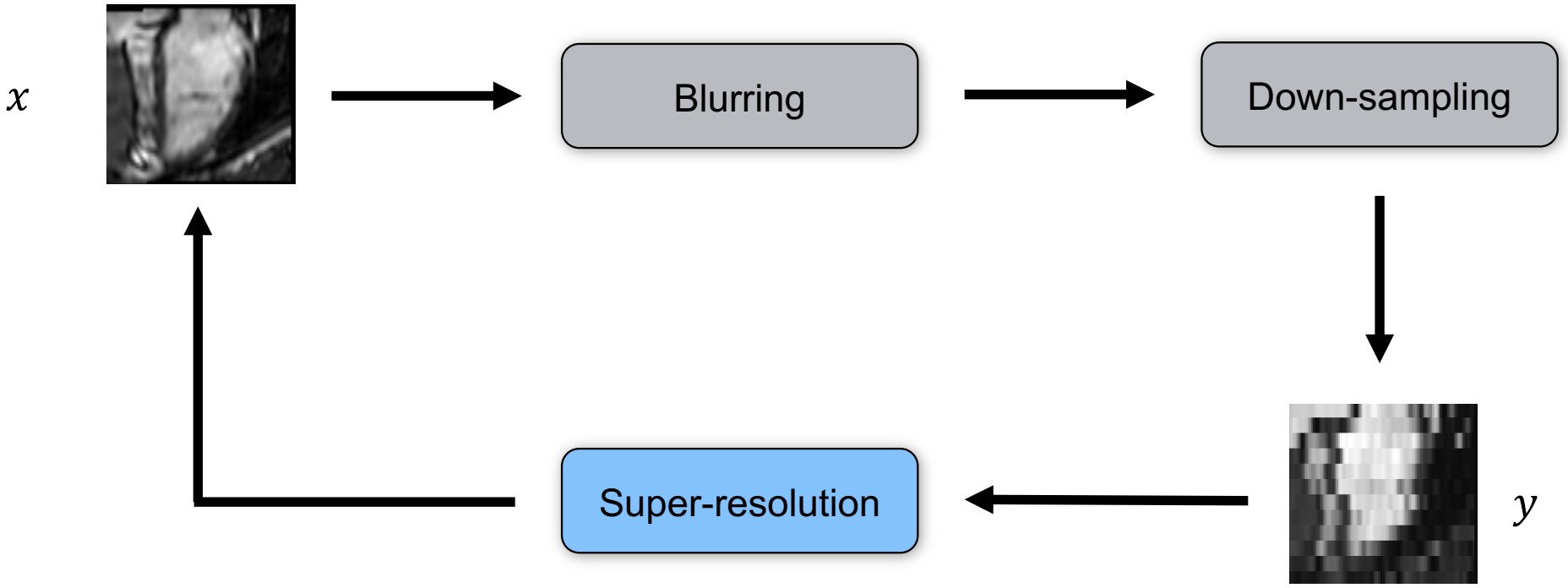
(a) 11x Undersampled

Schlemper et al. IEEE TMI 2017

Forward model/Inverse model

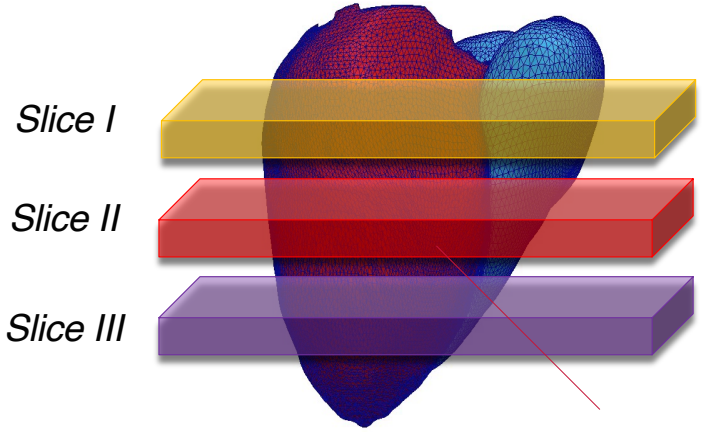


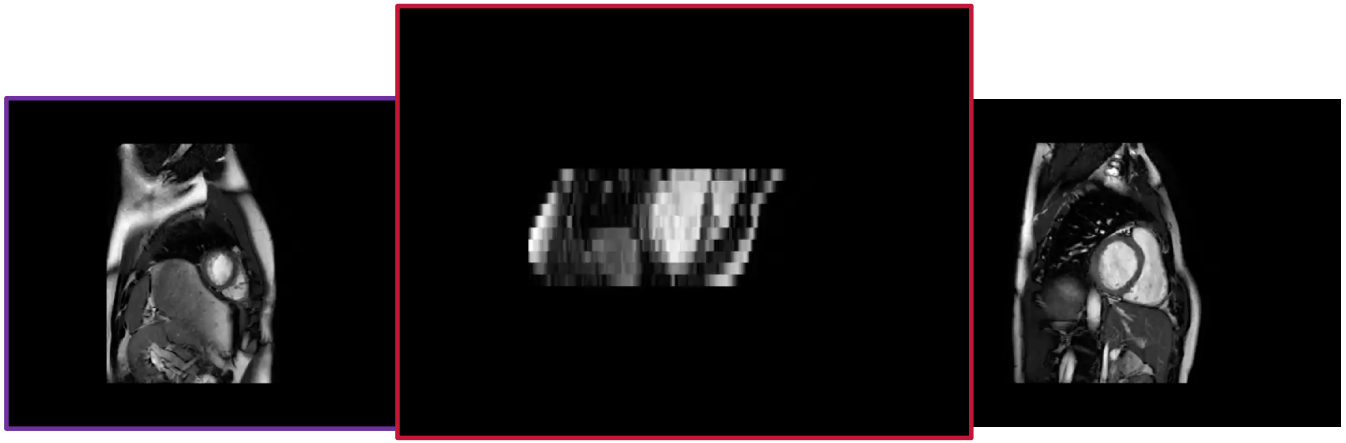
Forward model/Inverse model



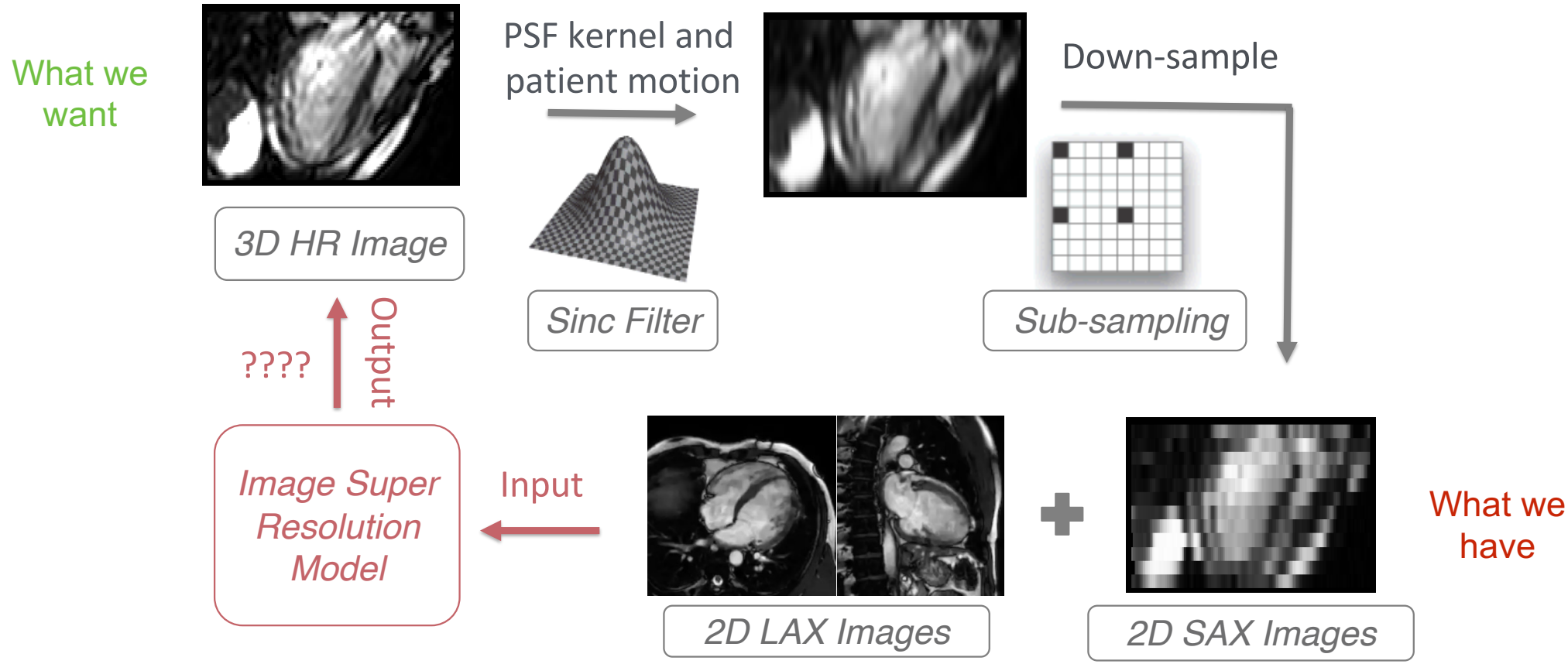
AI-enabled image super-resolution

- Acquisition of cardiac MRI typically consists of 2D multi-slice data due to
  - constraints on SNR
  - breath-hold time
  - total acquisition time
- This leads to thick slice data (thickness 8-10 mm per slice)

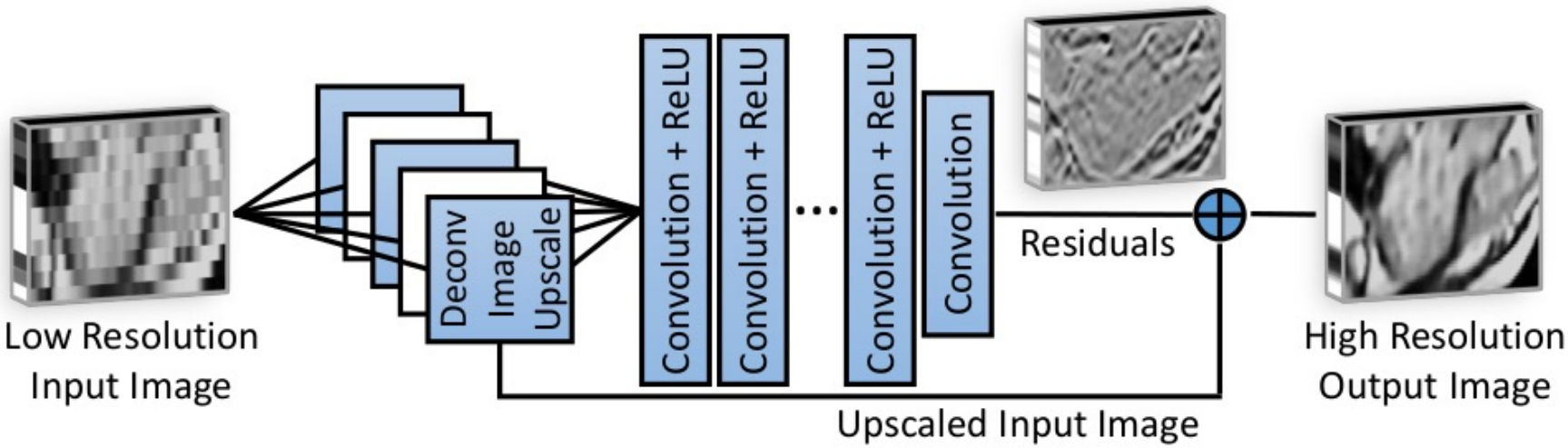




Deep learning for image super-resolution

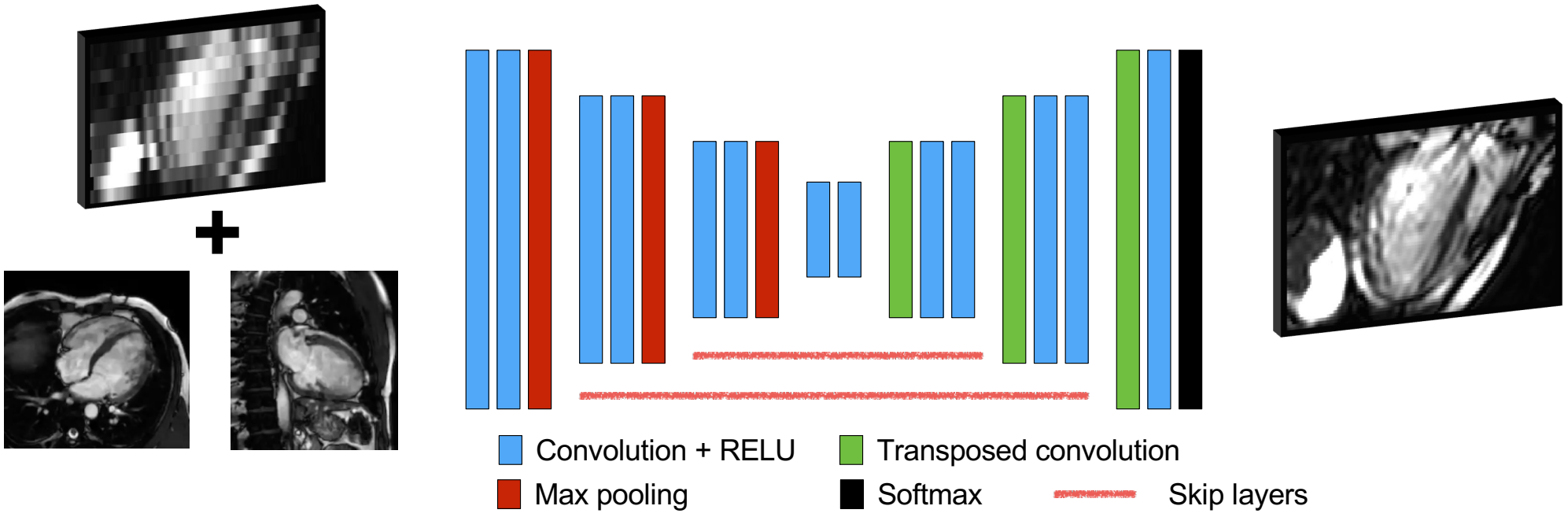


Deep learning for image super-resolution

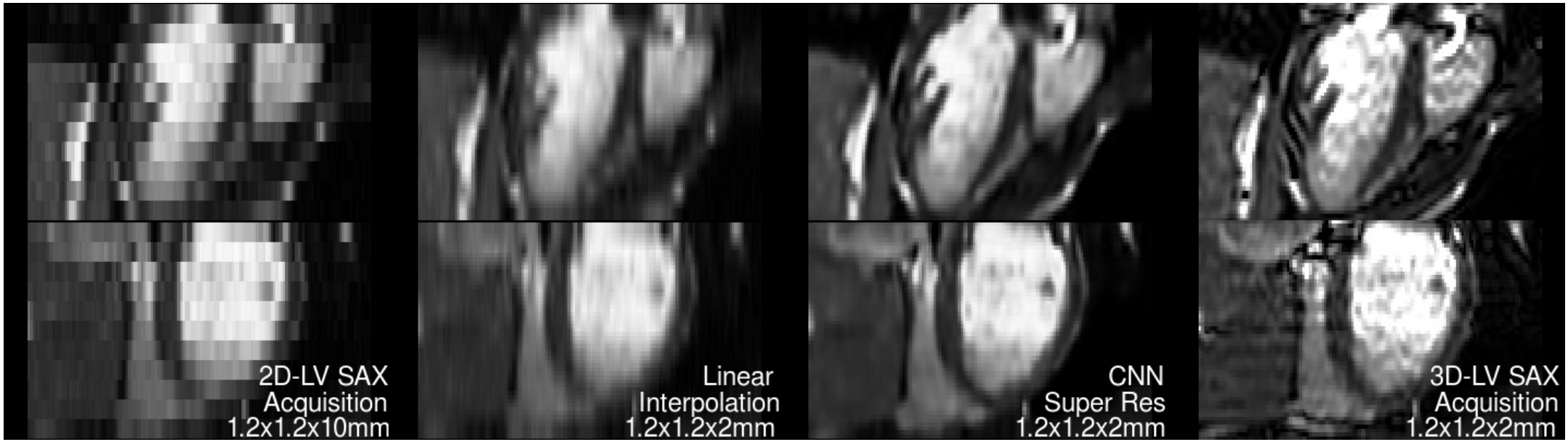


O. Oktay et al. IEEE TMI 2017

AI-enabled image super-resolution



AI-enabled image super-resolution



O. Oktay et al. IEEE TMI 2018

That’s all for now…

References

- Deep Learning for Image Super-resolution: A Survey
  – https://arxiv.org/abs/1902.06068
- Deep Learning Techniques for Inverse Problems in Imaging
  – https://arxiv.org/abs/2005.06001