Chapter 4: CNNs for Other Supervised Learning Tasks

Mathematical Foundations of Deep Neural Networks

Fall 2022

Department of Mathematical Sciences

Ernest K. Ryu

Seoul National University

Inverse problem model

In *inverse problems*, we wish to recover a signal X_{true} given measurements Y. The unknown and the measurements are related through

$$\mathcal{A}[X_{\text{true}}] + \varepsilon = Y,$$

where $\mathcal{A}$ is often, but not always, linear, and ε represents small error.

The *forward model* $\mathcal{A}$ may or may not be known. In other words, the goal of an inverse problem is to find an approximation of $\mathcal{A}^{-1}$.

In many cases, $\mathcal{A}$ is not even be invertible. In such cases, we can still hope to find an mapping that serves as an approximate inverse in practice.

Gaussian denoising

Given $X_{\text{true}} \in \mathbb{R}^{w \times h}$, we measure

$$Y = X_{\text{true}} + \varepsilon$$

where $\varepsilon_{ij} \sim \mathcal{N}(0, \sigma^2)$ is IID Gaussian noise. For the sake of simplicity, assume we know σ. Goal is to recover X_{true} from Y.

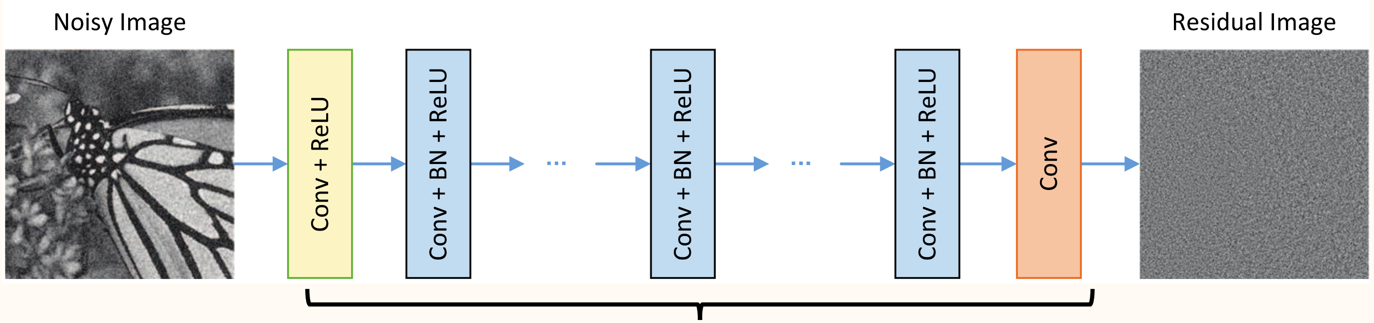
Guassian denoising is the simplest setup in which the goal is to remove noise from the image. In more realistic setups, the noise model will be more complicated and the noise level σ will be unknown.

DnCNN

In 2017, Zhang et al. presented the denoising convolutional neural networks (DnCNNs). They trained a 17-layer CNN f_θ to learn the noise with the loss

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \|f_\theta(Y_i) - (Y_i - X_i)\|^2$$

so that the clean recovery can be obtained with $Y_i - f_\theta(Y_i)$. (This is equivalent to using a residual connection from beginning to end.)



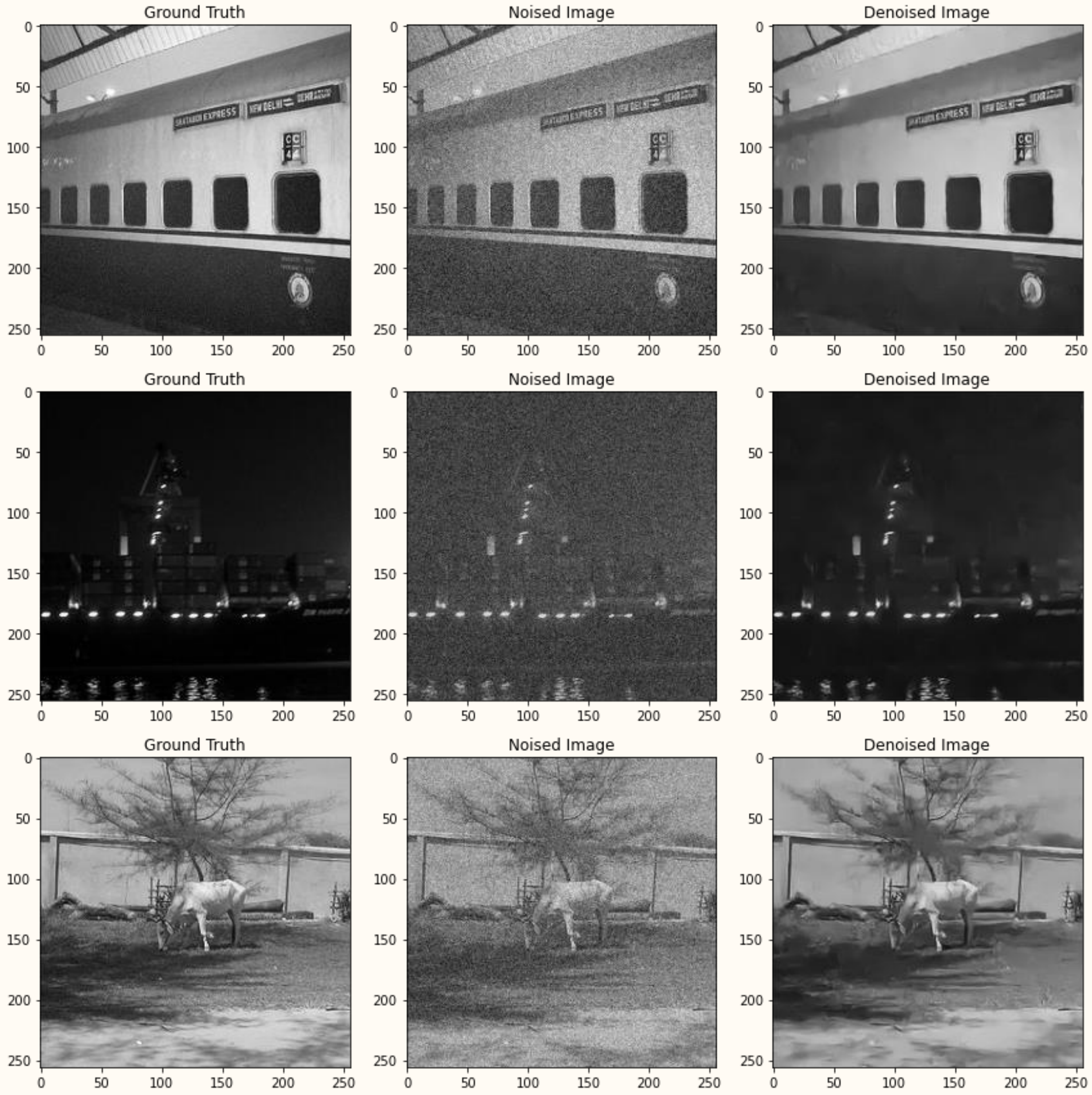
17 layers of
3x3 conv.
p=1

K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising, *IEEE TIP*, 2017.

DnCNN

Image denoising is was an area with a large body of prior work. DnCNN dominated all prior approaches that were not based on deep learning.

Nowadays, all state-of-the-art denoising algorithms are based on deep learning.



K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising, *IEEE TIP*, 2017.

Inverse problems via deep learning

In deep learning, we use a neural network to approximate the inverse mapping

$$f_\theta \approx \mathcal{A}^{-1}$$

i.e., we want $f_\theta(Y) \approx X_{\text{true}}$ for the measurements X that we care about.

If we have $X_1, \ldots, X_N$ and $Y_1, \ldots, Y_N$ (but no direct knowledge of $\mathcal{A}$), we can solve

$$\underset{\theta \in \mathbb{R}^p}{\text{minimize}} \quad \sum_{i=1}^{N} \|f_\theta(Y_i) - X_i\|$$

If we have $X_1, \ldots, X_N$ and knowledge of $\mathcal{A}$, we can solve

$$\underset{\theta \in \mathbb{R}^p}{\text{minimize}} \quad \sum_{i=1}^{N} \|f_\theta[\mathcal{A}(X_i)] - X_i\|$$

If we have $Y_1, \ldots, Y_N$ and knowledge of $\mathcal{A}$, we can solve

$$\underset{\theta \in \mathbb{R}^p}{\text{minimize}} \quad \sum_{i=1}^{N} \|\mathcal{A}[f_\theta(Y_i)] - Y_i\|$$

Image super-resolution

Given $X_{\text{true}} \in \mathbb{R}^{w \times h}$, we measure

$$Y = A(X_{\text{true}})$$

where A is a “downsampling” operator. So $Y \in \mathbb{R}^{w_2 \times h_2}$ with $w_2 < w$ and $h_2 < h$. Goal is to recover X_{true} from Y.

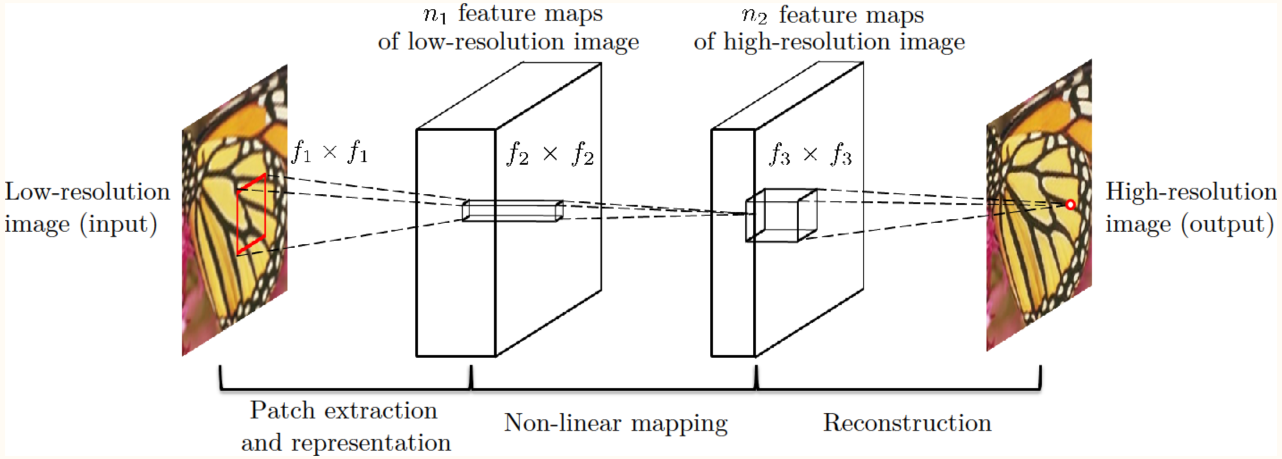
In the simplest setup, A is an average pool operator with $r \times r$ kernel and a stride r.

SRCNN

In 2015, Dong et al. presented super-resolution convolutional neural network (SRCNN). They trained a 3-layer CNN f_θ to learn the high-resolution reconstruction with the loss

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \left\| f_\theta(\tilde{Y}_i) - X_i \right\|^2$$

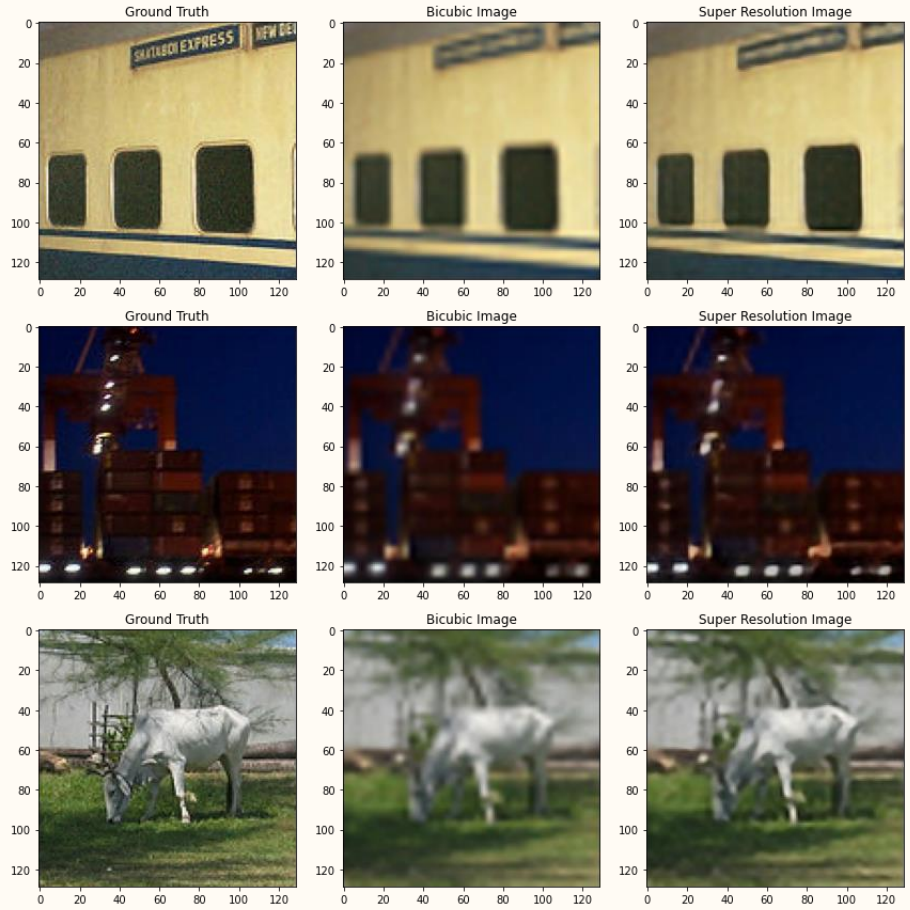
where $\tilde{Y}_i \in \mathbb{R}^{w\times h}$ is an upsampled version of $Y_i \in \mathbb{R}^{(w/r)\times(h/r)}$, i.e., $\tilde{Y}_i$ has the same number of pixels as X_i, but the image is pixelated or blurry. The goal is to have $f_\theta(\tilde{Y}_i)$ be a sharp reconstruction.



C. Dong, C. C. Loy, K. He, and X. Tang, Image super-resolution using deep convolutional networks, *IEEE TPAMI*, 2015.

SRCNN

SRCNN showed that simple learning based approaches can match the state-of the-art performances of super-resolution task.



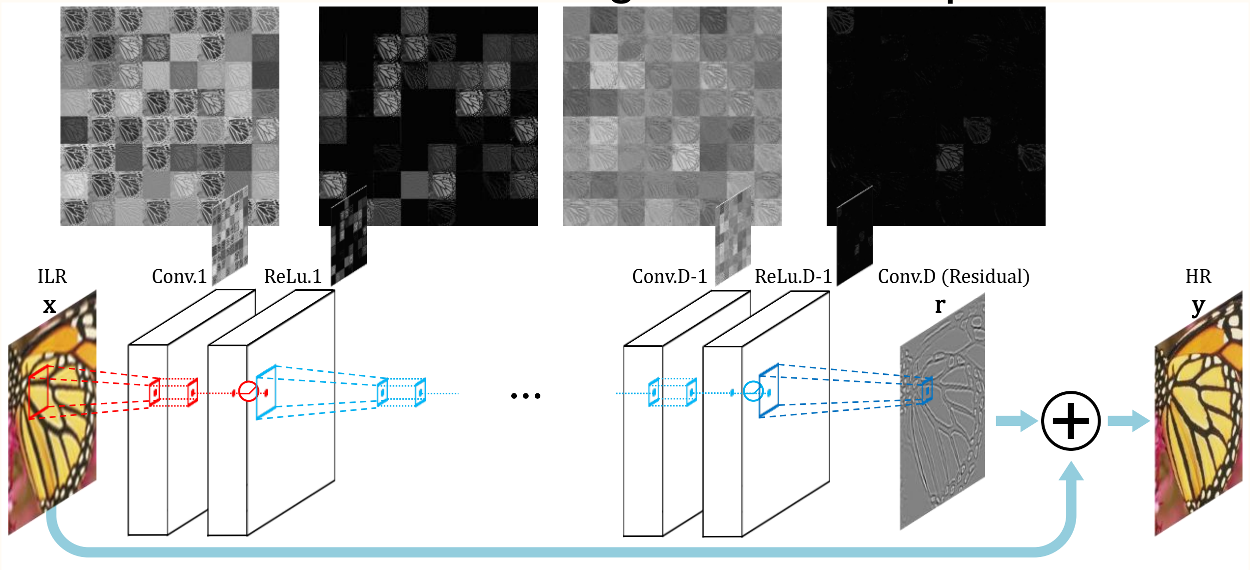
C. Dong, C. C. Loy, K. He, and X. Tang, Image super-resolution using deep convolutional networks, *IEEE TPAMI*, 2015.

VDSR

In 2016, Kim et al. presented VDSR. They trained a 20-layer CNN with a residual connection f_θ to learn the high-resolution reconstruction with the loss

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} \left\| f_\theta(\tilde{Y}_i) - X_i \right\|^2$$

The residual connection was the key insight that enabled the training of much deeper CNNs.

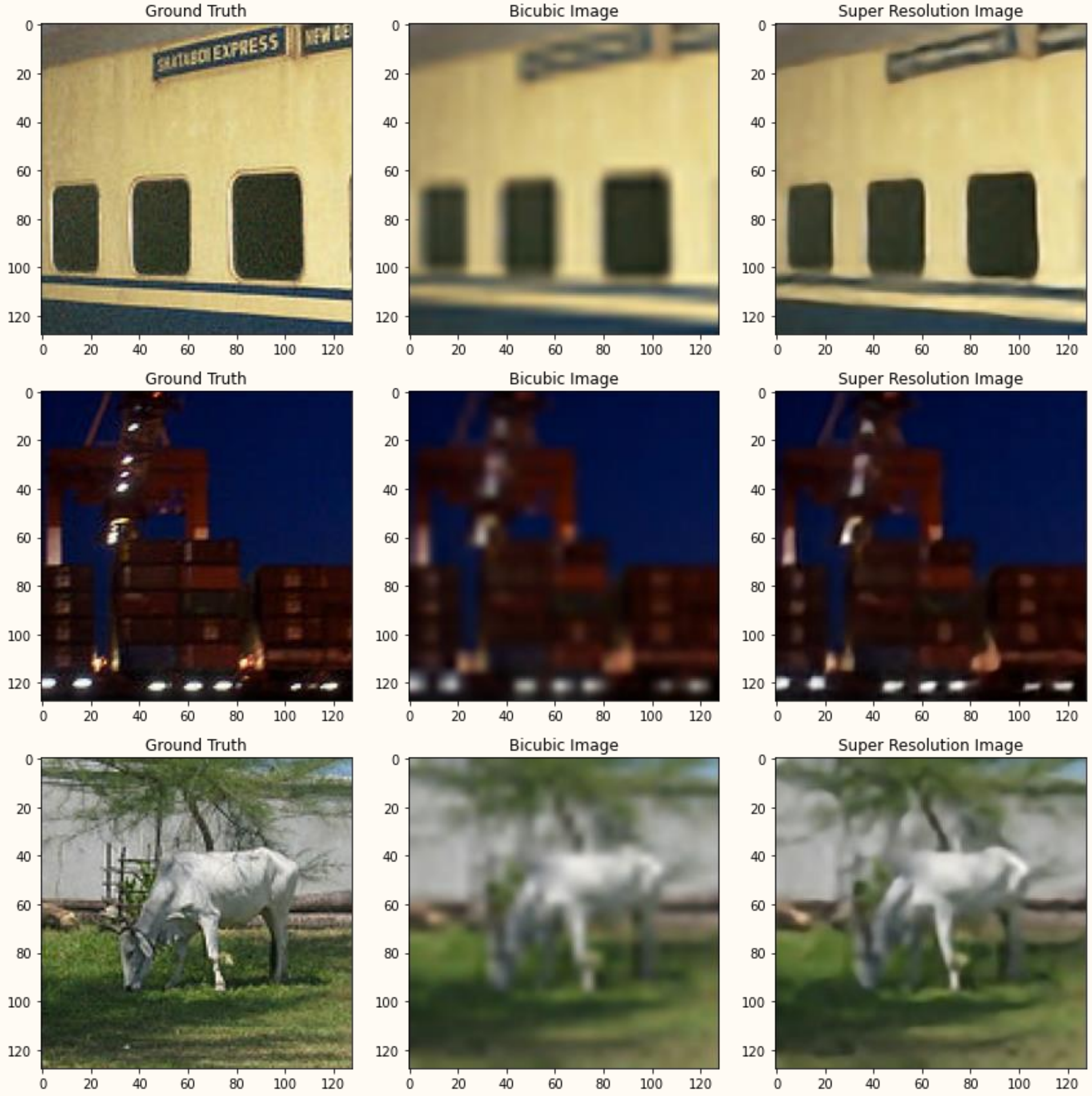


J. Kim, J. K. Lee, and K. M. Lee, Accurate image super-resolution using very deep convolutional networks, *CVPR*, 2016.

VDSR

VDSR dominated all prior approaches not based on deep learning.

showed that simple learning based approaches can batch the state-of the-art performances of super-resolution task.



J. Kim, J. K. Lee, and K. M. Lee, Accurate image super-resolution using very deep convolutional networks, *CVPR*, 2016.

Other inverse problem tasks and results

There are many other inverse problems. Almost all of them now require deep neural networks to achieve state-of-the-art results.

We won’t spend more time on inverse problems in this course, but let’s have fun and see a few other tasks and results. (These results are based on much more complex architectures and loss functions.)

SRGAN


bicubic interpolation


SRGAN


ground truth

C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, Photo-realistic single image super-resolution using a generative adversarial network, *CVPR*, 2017.

SRGAN

bicubic interpolation

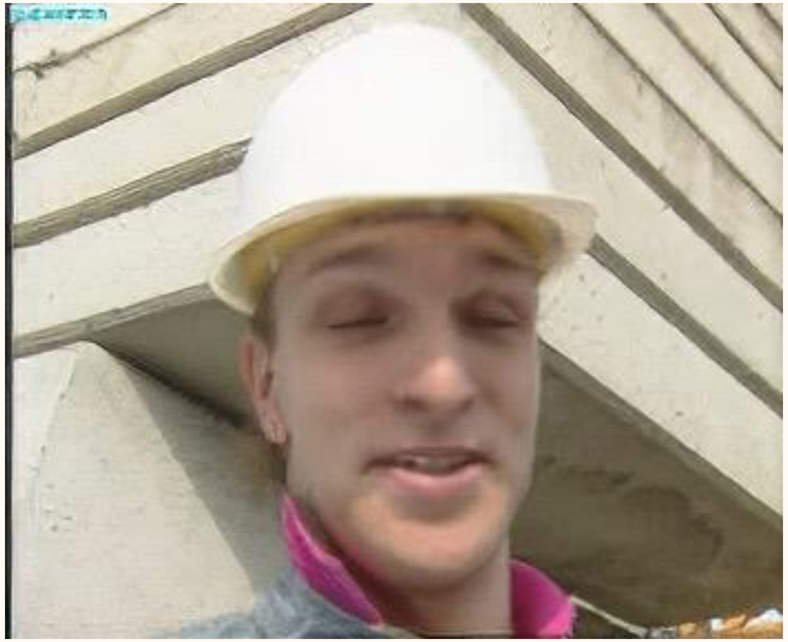
SRGAN

ground truth

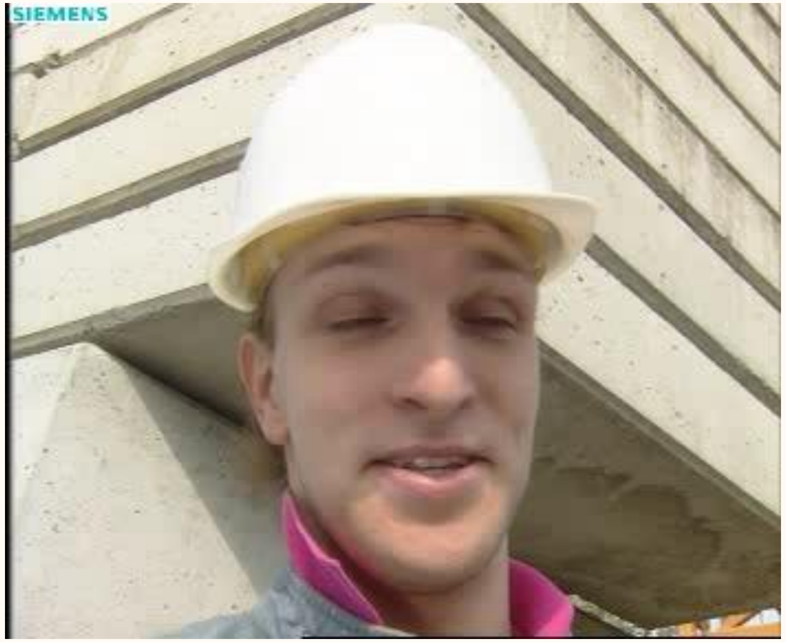
C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, Photo-realistic single image super-resolution using a generative adversarial network, *CVPR*, 2017.

SRGAN

bicubic interpolation



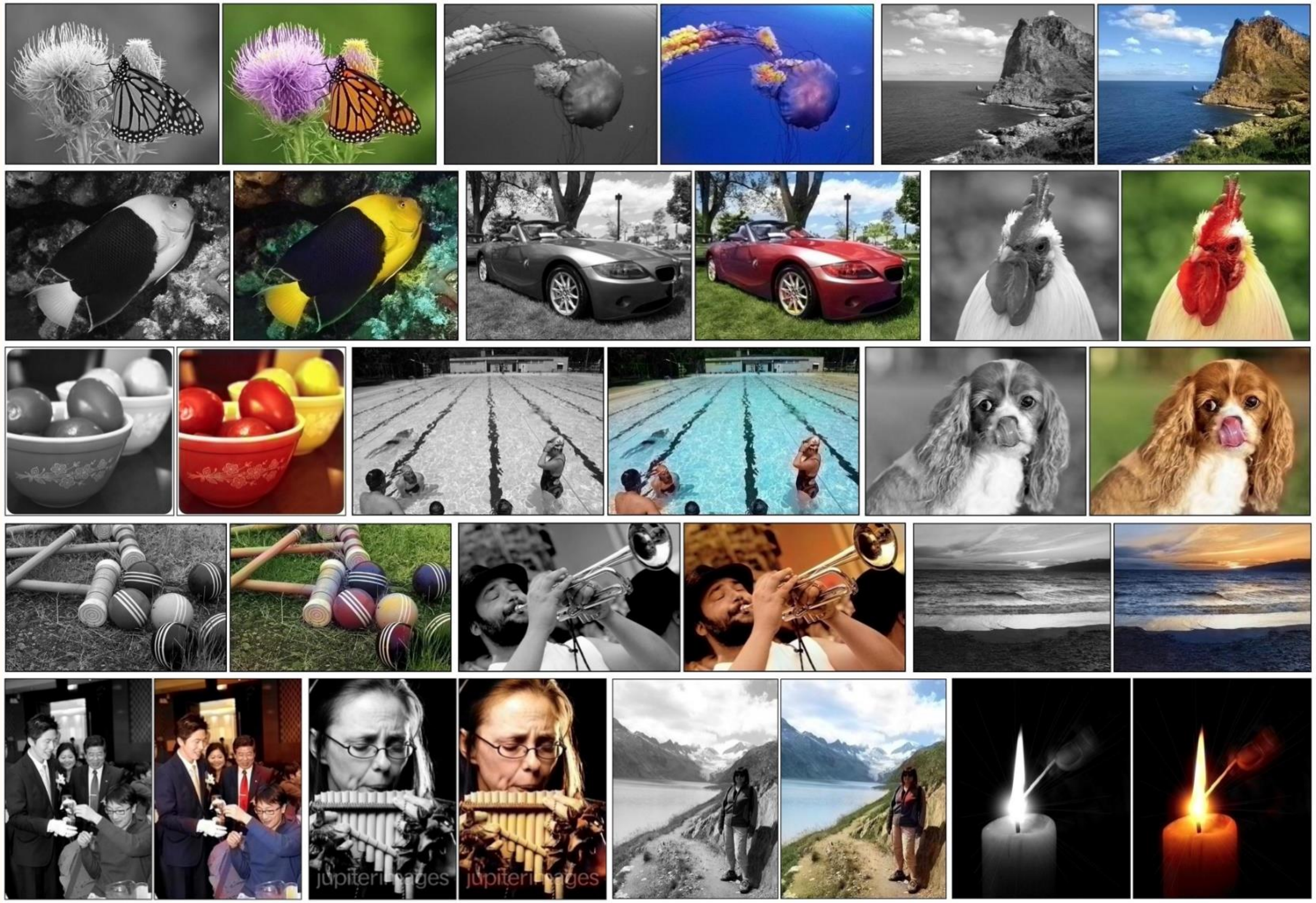
SRGAN



ground truth

C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, Photo-realistic single image super-resolution using a generative adversarial network, *CVPR*, 2017.

Image colorization



R. Zhang, P. Isola, and A. A. Efros, Colorful image colorization, *ECCV*, 2016.

Image inpainting

J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, Generative image inpainting with contextual attention, *CVPR*, 2018.

Image inpainting

input output ground truth

J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, Generative image inpainting with contextual attention, *CVPR*, 2018.

Linear operator ≅ matrix

Core tenet of linear algebra: matrices are linear operators and linear operators are matrices.

Let $f : \mathbb{R}^n \to \mathbb{R}^m$ be linear, i.e.,

$$f(x+y) = f(x) + f(y) \quad \text{and} \quad f(\alpha x) = \alpha f(x)$$

for all $x, y \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$.

There exists a matrix $A \in \mathbb{R}^{m \times n}$ that *represents* $f : \mathbb{R}^n \to \mathbb{R}^m$, i.e.,

$$f(x) = Ax$$

for all $x \in \mathbb{R}^n$.

Linear operator ≅ matrix

Let e_i be the i-th unit vector, i.e., e_i has all zeros elements except entry 1 in the i-th coordinate.

Given a linear $f : \mathbb{R}^n \to \mathbb{R}^m$, we can find the matrix

$$A = [A_{:,1} \quad A_{:,2} \quad \cdots \quad A_{:,n}] \in \mathbb{R}^{m \times n}$$

representing f with

$$f(e_j) = Ae_j = A_{:,j}$$

for all $j = 1, \dots, n$, or with

$$e_i^\top f(e_j) = e_i^\top A e_j = A_{i,j}$$

for all $i = 1, \dots, m$ and $j = 1, \dots, n$.

Linear operator $\ncong$ matrix

In applied mathematics and machine learning, there are many setups where explicitly forming the matrix representation $A \in \mathbb{R}^{m \times n}$ is costly, even though the matrix-vector products Ax and $A^\top y$ are efficient to evaluate.

In machine learning, convolutions are the primary example. Other areas, linear operators based on FFTs are the primary example.

In such setups, the matrix representation is still a useful conceptual tool, even if we never intend to form the matrix.

Transpose (adjoint) of a linear operator

Given a matrix A, the transpose $A^\top$ is obtained by flipping the row and column dimensions, i.e., $(A^\top)_{ij} = (A)_{ji}$. However, using this definition is not always the most effective when understanding the action of $A^\top$.

Another approach is to use the adjoint view. Since

$$y^\top(Ax) = (A^\top y)^\top x$$

for any $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, understand the action of $A^\top$ by finding an expression of the form

$$y^\top Ax = \sum_{j=1}^{n} (\text{something})_j \ x_j = (A^\top y)^\top x$$

Example: 1D transpose convolution

Consider the 1D convolution represented by $A \in \mathbb{R}^{(n-f+1)\times n}$ defined with a given $w \in \mathbb{R}^f$ and

$$A = \begin{bmatrix} w_1 & \cdots & w_f & 0 & \cdots & & & 0 \\ 0 & w_1 & \cdots & w_f & 0 & \cdots & & 0 \\ 0 & 0 & w_1 & \cdots & w_f & 0 & \cdots & 0 \\ \vdots & & & \ddots & & \ddots & & \vdots \\ 0 & & \cdots & 0 & w_1 & \cdots & w_f & 0 \\ 0 & & \cdots & 0 & 0 & w_1 & \cdots & w_f \end{bmatrix}.$$

Then we have

$$(Ax)_j = \sum_{i=1}^{f} w_i x_{j+i-1}$$

Example: 1D transpose convolution

and we have the following formula which coincides with transposing the matrix A.

For more complicated linear operators, this is how you understand the transpose operation.

$$
\begin{aligned}
y^\intercal A x &= \sum_{j=1}^{n-f+1} y_j \sum_{i=1}^{f} w_i x_{j+i-1} \\
&= \sum_{j=1}^{n-f+1} \sum_{i=1}^{f} y_j w_i x_{j+i-1} \sum_{k=1}^{n} \mathbf{1}_{\{k=j+i-1\}} \\
&= \sum_{k=1}^{n} \sum_{j=1}^{n-f+1} \sum_{i=1}^{f} y_j w_i x_k \mathbf{1}_{\{k-j+1=i\}} \\
&= \sum_{k=1}^{n} x_k \sum_{j=1}^{n-f+1} \sum_{i=1}^{f} w_{k-j+1} y_j \mathbf{1}_{\{k-j+1=i\}} \\
&= \sum_{k=1}^{n} x_k \sum_{j=1}^{n-f+1} w_{k-j+1} y_j \sum_{i=1}^{f} \mathbf{1}_{\{k-j+1=i\}} \\
&= \sum_{k=1}^{n} x_k \sum_{j=1}^{n-f+1} w_{k-j+1} y_j \mathbf{1}_{\{1 \le k-j+1 \le f\}} \\
&= \sum_{k=1}^{n} x_k \sum_{j=1}^{n-f+1} w_{k-j+1} y_j \mathbf{1}_{\{j \le k\}} \mathbf{1}_{\{k-f+1 \le j\}} \\
&= \sum_{k=1}^{n} x_k \sum_{j=\max(k-f+1,1)}^{\min(n-f+1,k)} w_{k-j+1} y_j = (A^\intercal y)^\intercal x
\end{aligned}
$$

Operations increasing spatial dimensions

In image classification tasks, the spatial dimensions of neural networks often decrease as the depth progresses.

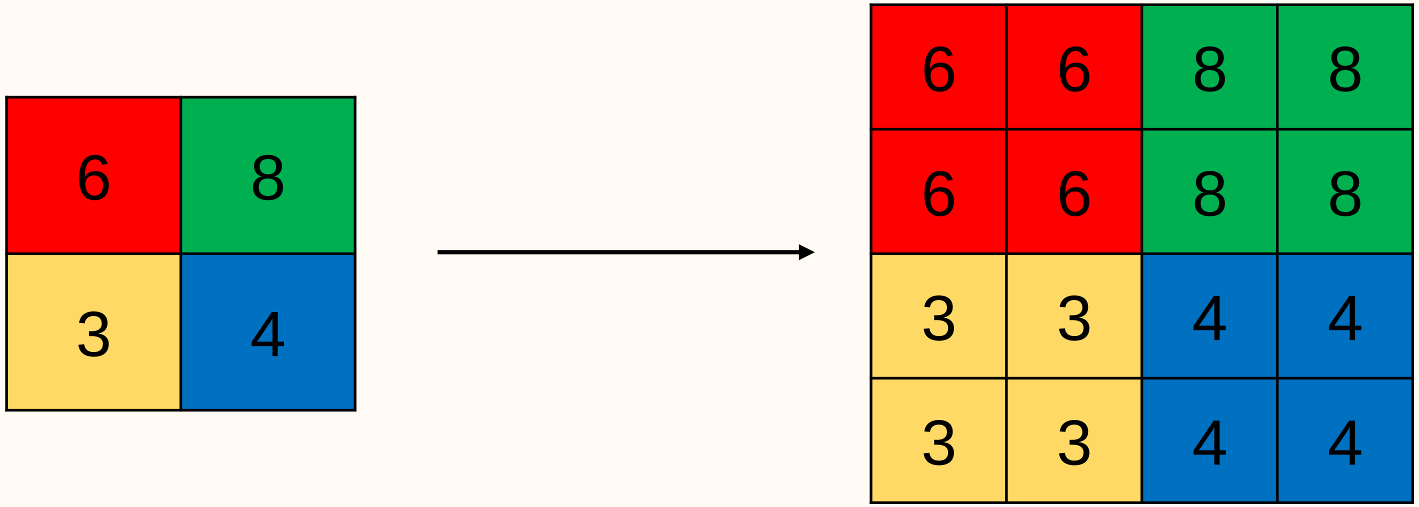
This is because we are trying to forget location information. (In classification, we care about *what* is in the image, but we do not *where* it is in the image.)

However, there are many networks for which we want to increase the spatial dimension:

- Linear layers
- Upsampling
- Transposed convolution

Upsampling: Nearest neighbor

`torch.nn.Upsample` with `mode='nearest'`



Upsampling: Bilinear interpolation

`Torch.nn.Upsample` with `mode='bilinear'`

(We won’t pay attention to the interpolation formula.)

6	8
3	4

→

6.0000	6.5000	7.5000	8.0000
5.2500	5.6875	6.5625	7.0000
3.7500	4.0625	4.6875	5.0000
3.0000	3.2500	3.7500	4.0000

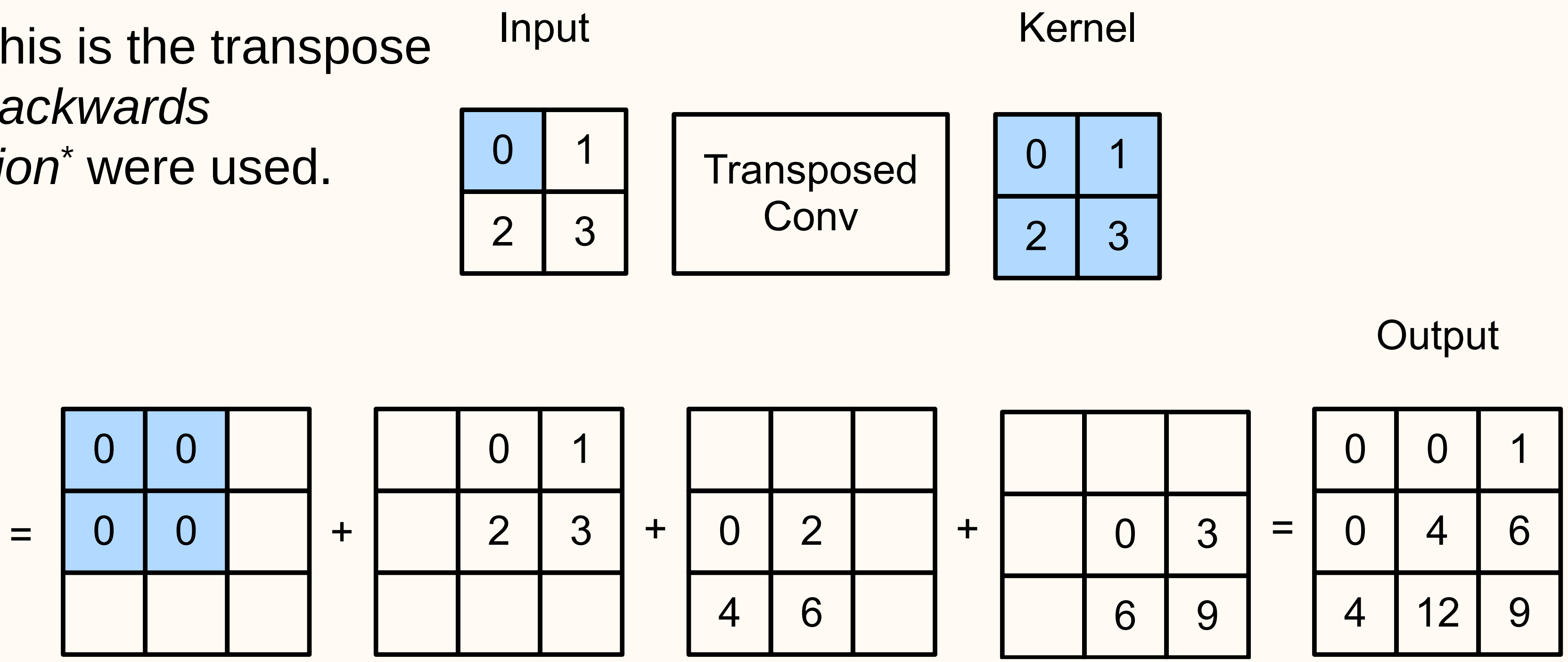
`'linear'` interpolation is available for 1D data

`'trilinear'` interpolation is available for 3D data

Transposed convolution

In *transposed convolution*, Input neurons additively distribute values to the output via the kernel.

Before people noticed that this is the transpose of convolution, the names *backwards convolution* and *deconvolution*[*] were used.

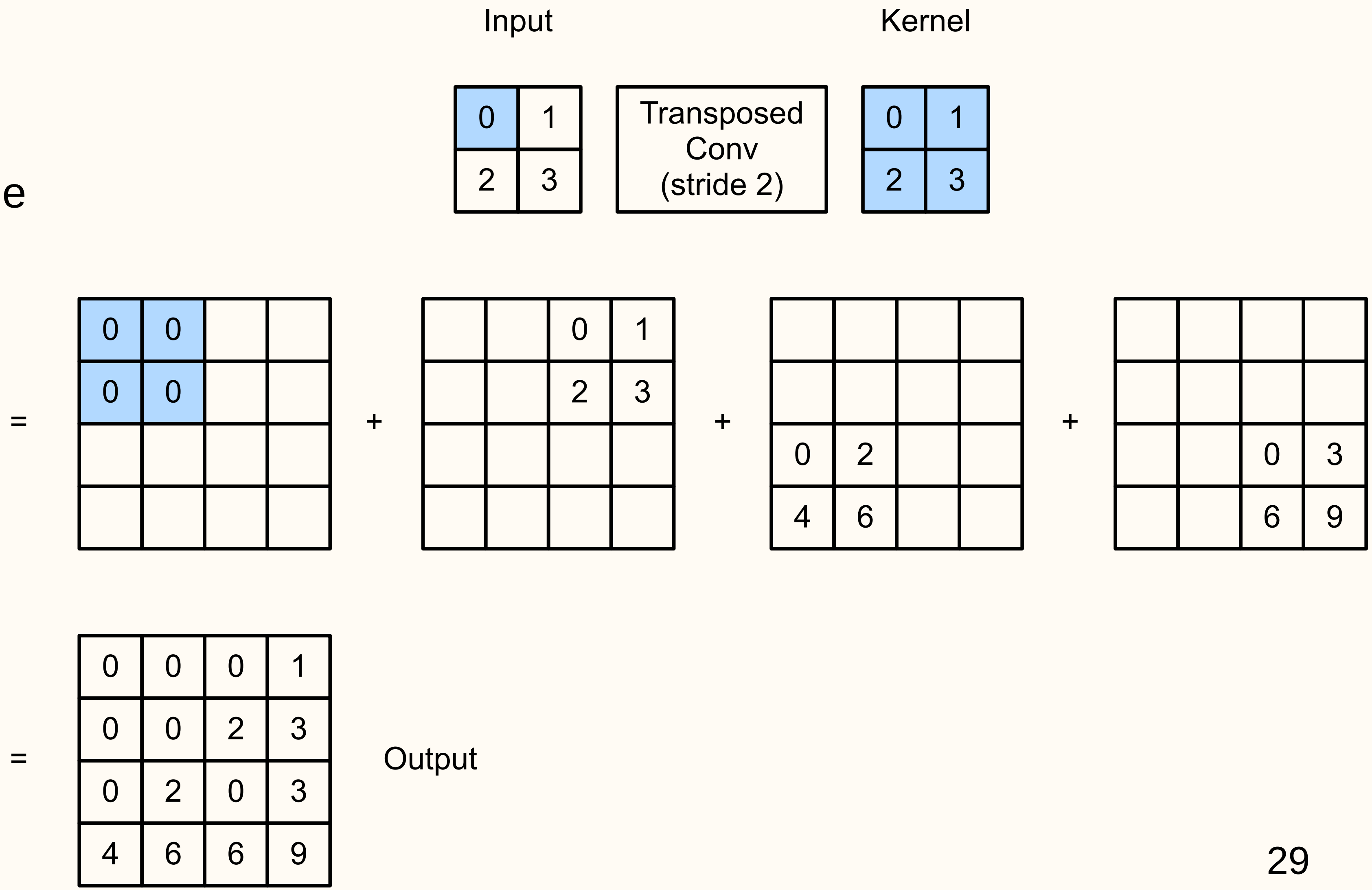


[*]This is a particularly bad name as deconvolution refers to the inverse, rather than transpose, of the convolution in signal processing.

Transposed convolution

For each input neuron, multiply the kernel and add (accumulate) the value in the output.

Can accommodate strides, padding, and multiple channels.



Convolution visualized

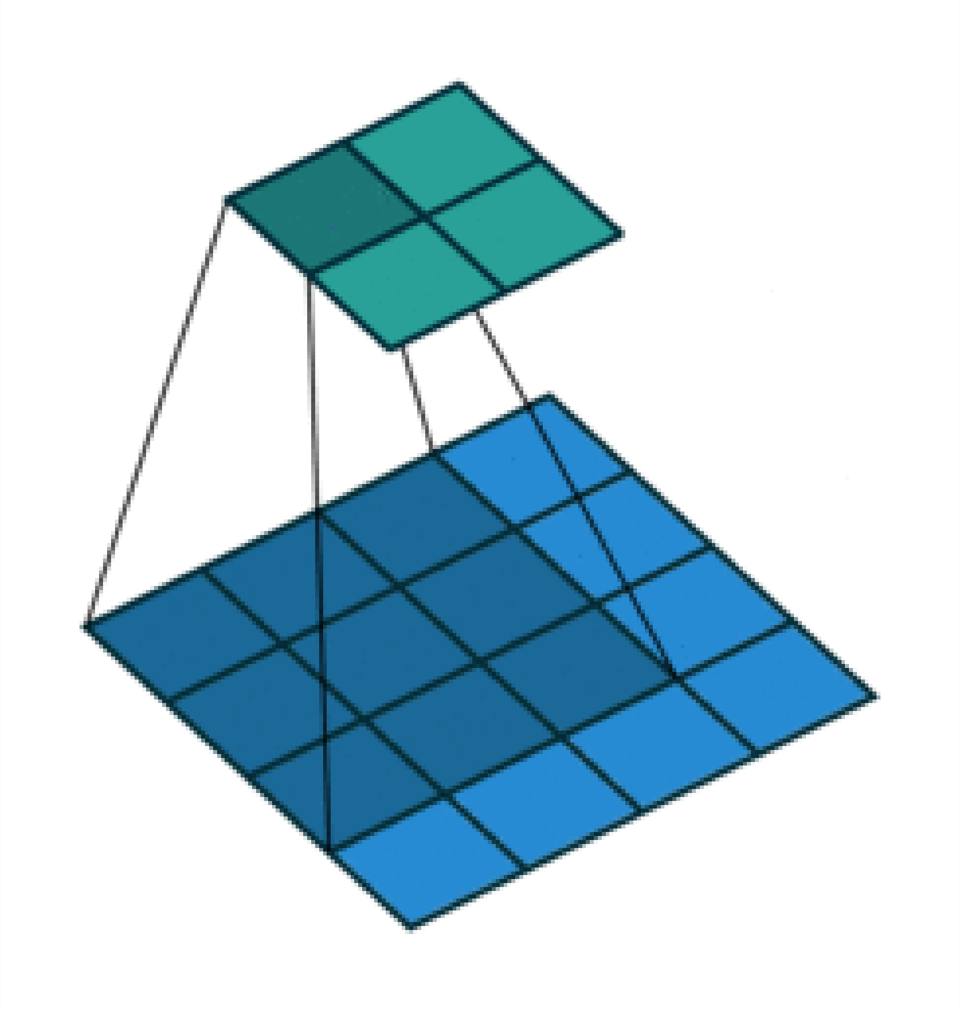


Illustration due to: https://medium.com/apache-mxnet/convolutions-explained-with-ms-excel-465d6649831c
D. Mishra, Convolutions explained with… MS Excel!, *Medium*, 2018.

Transpose convolution visualized

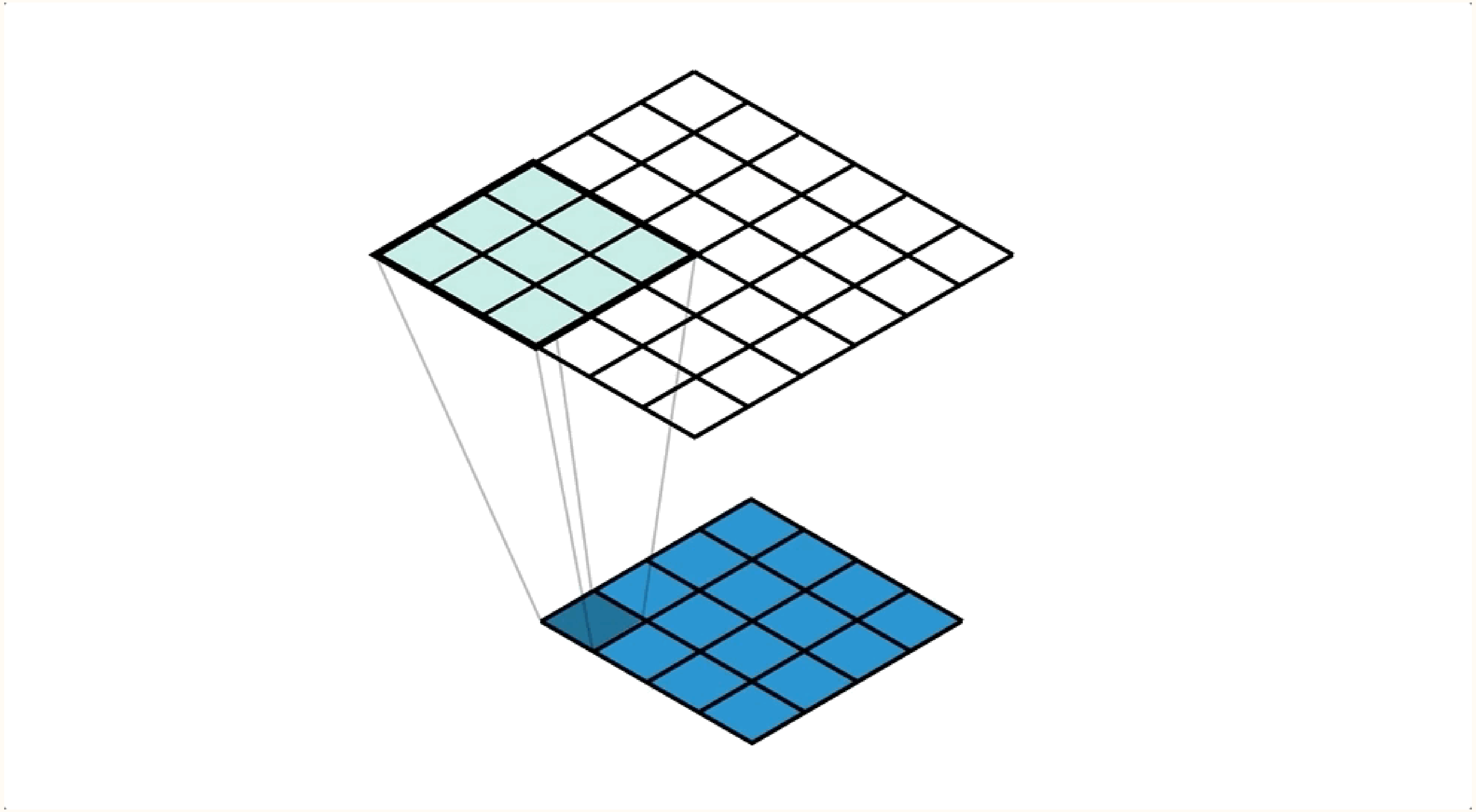


Illustration due to: https://medium.com/apache-mxnet/transposed-convolutions-explained-with-ms-excel-52d13030c7e8
D. Mishra, Transposed convolutions explained with… MS Excel!, *Medium*, 2018.

2D trans. Conv. layer: Formal definition

Input tensor: $Y \in \mathbb{R}^{B \times C_{\text{in}} \times m \times n}$, B batch size, C_{in} # of input channels.

Output tensor: $X \in \mathbb{R}^{B \times C_{\text{out}} \times (m+f_1-1) \times (n+f_2-1)}$, B batch size, C_{out} # of output channels, m, n # of vertical and horizontal indices.

Filter $w \in \mathbb{R}^{C_{\text{in}} \times C_{\text{out}} \times f_1 \times f_2}$, bias $b \in \mathbb{R}^{C_{\text{out}}}$. (If `bias=False`, then $b = 0$.)

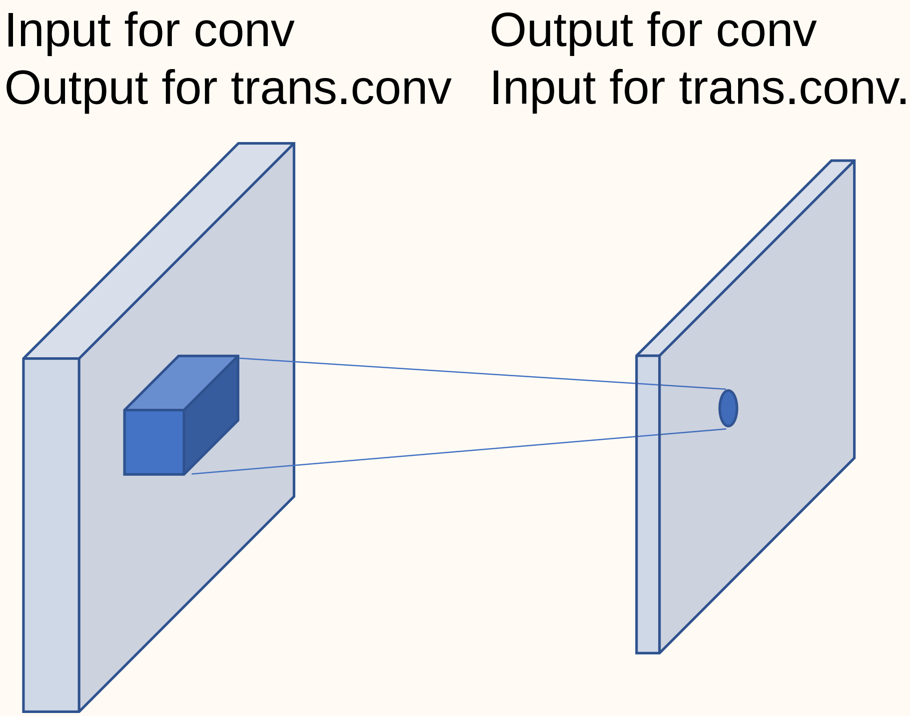
```
def trans_conv(Y, w, b):
  c_in, c_out, f1, f2 = w.shape
  batch, c_in, m, n = Y.shape
  X = torch.zeros(batch, c_out, m + f1 - 1, n + f2 - 1)
  for k in range(c_in):
    for i in range(Y.shape[2]):
      for j in range(Y.shape[3]):
        X[:, :, i:i+f1, j:j+f2] += Y[:, k, i, j].view(-1,1,1,1)*w[k, :, :, :].unsqueeze(0)
  return X + b.view(1,-1,1,1)
```

Dependency by sparsity pattern

In a matrix representation A of convolution. The dependencies of the inputs and outputs are represented by the non-zeros of A, i.e., the sparsity pattern of A.

If $A_{ij} = 0$, then input neuron j does not affect the output neuron i. If $A_{ij} \neq 0$, then $(A^\top)_{ji} \neq 0$. So if input neuron j affects output neuron i in convolution, then input neuron i affects output neuron j in transposed convolution.

We can combine this reasoning with our visual understanding of convolution. The diagram simultaneously illustrates the dependencies for both convolution and transposed convolution.

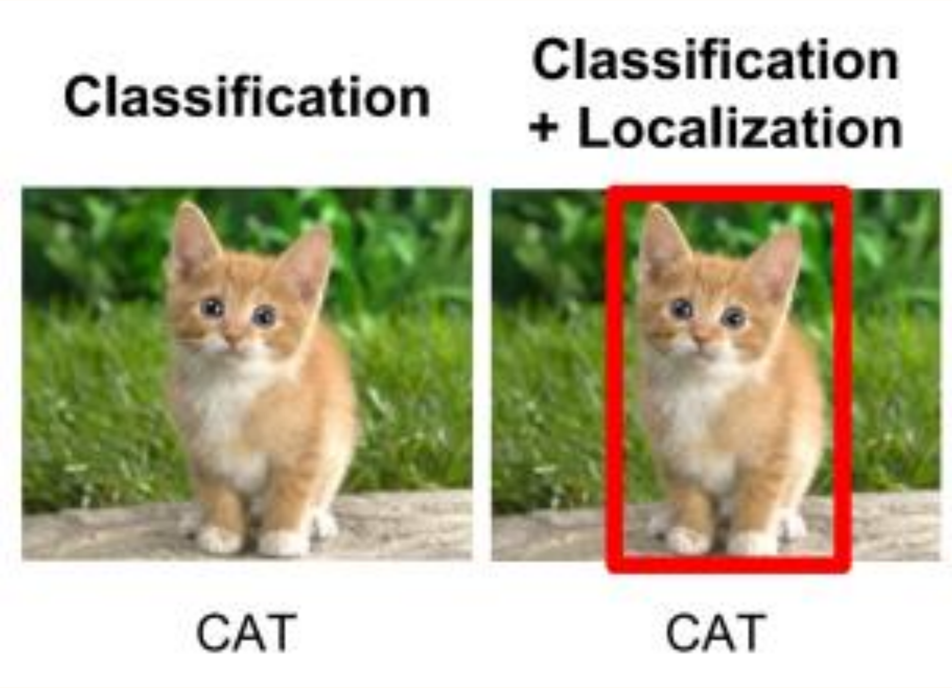


Semantic segmentation

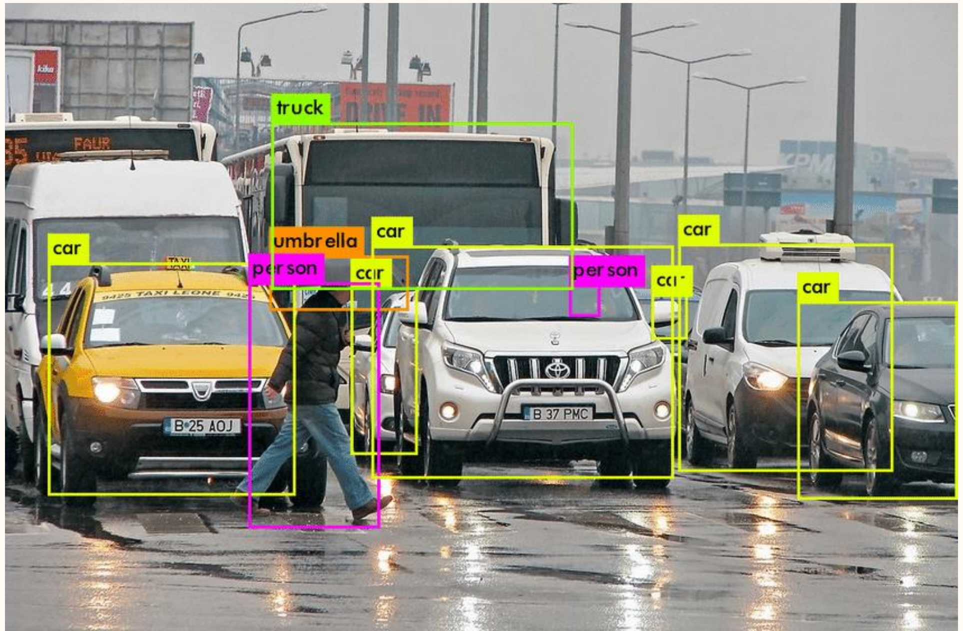
In *semantic segmentation*, the goal is to segment the image into semantically meaningful regions by classifying each pixel.

Other related tasks

Object localization localizes a single object usually via a bounding box.

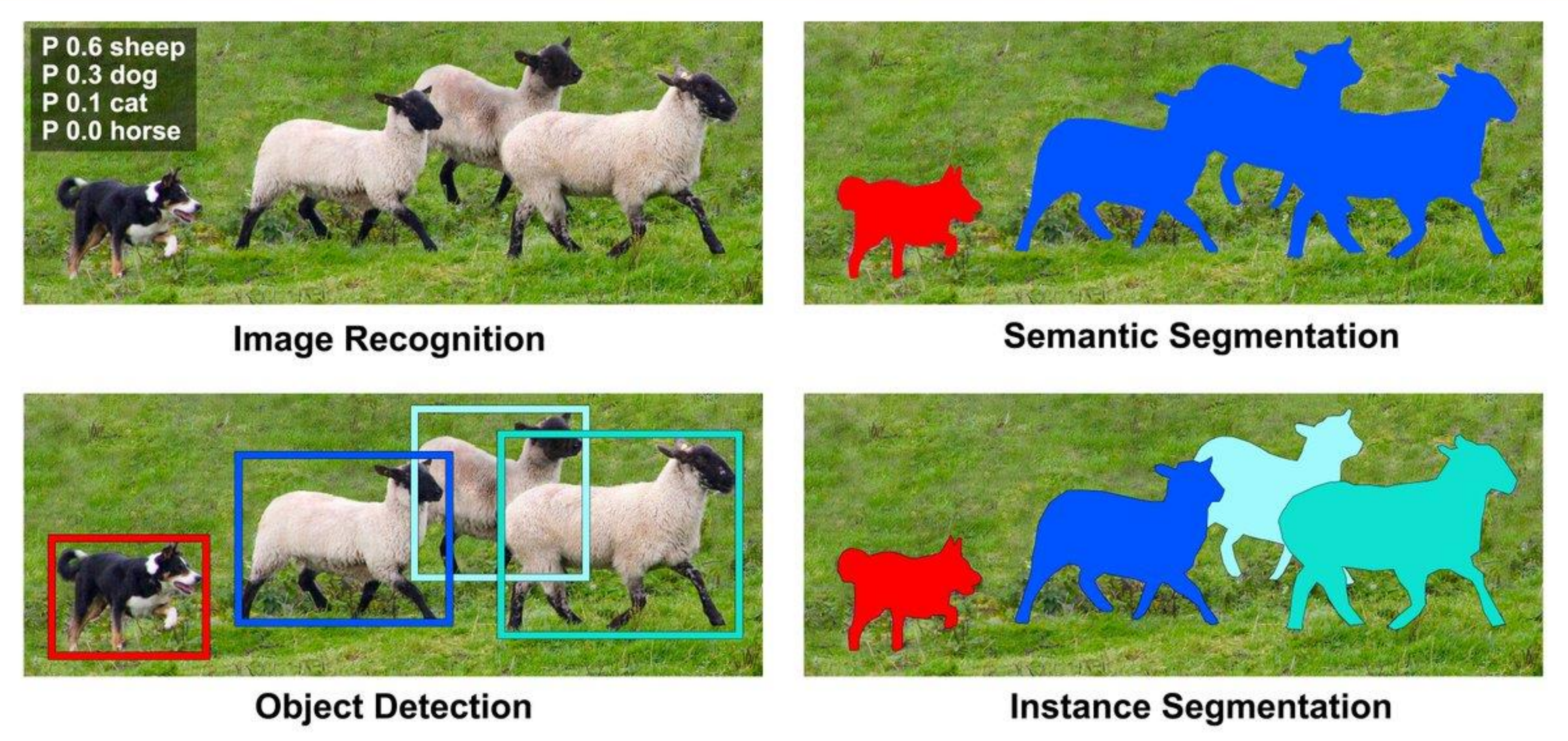


Object detection detects many objects, with the same class often repeated, usually via bounding boxes.



Other related tasks

Instance segmentation distinguishes multiple instances of the same object type.



Pascal VOC

We will use PASCAL Visual Object Classes (VOC) dataset for semantic segmentation.

(Dataset also contains labels for object detection.)

There are 21 classes: 20 main classes and 1 “unlabeled” class.

Data $X_1, \dots, X_N \in \mathbb{R}^{3\times m\times n}$ and labels $Y_1, \dots, Y_N \in \{0,1,\dots,20\}^{m\times n}$, i.e., Y_i provides a class label for every pixel of X_i.

Loss for semantic segmentation

Consider the neural network

$$f_\theta : \mathbb{R}^{3\times m\times n} \to \mathbb{R}^{k\times m\times n}$$

such that $\mu\big(f_\theta(X)\big)_{ij} \in \Delta^k$ is the probabilities for the k classes for pixel (i, j).

We minimize the sum of pixel-wise cross-entropy losses

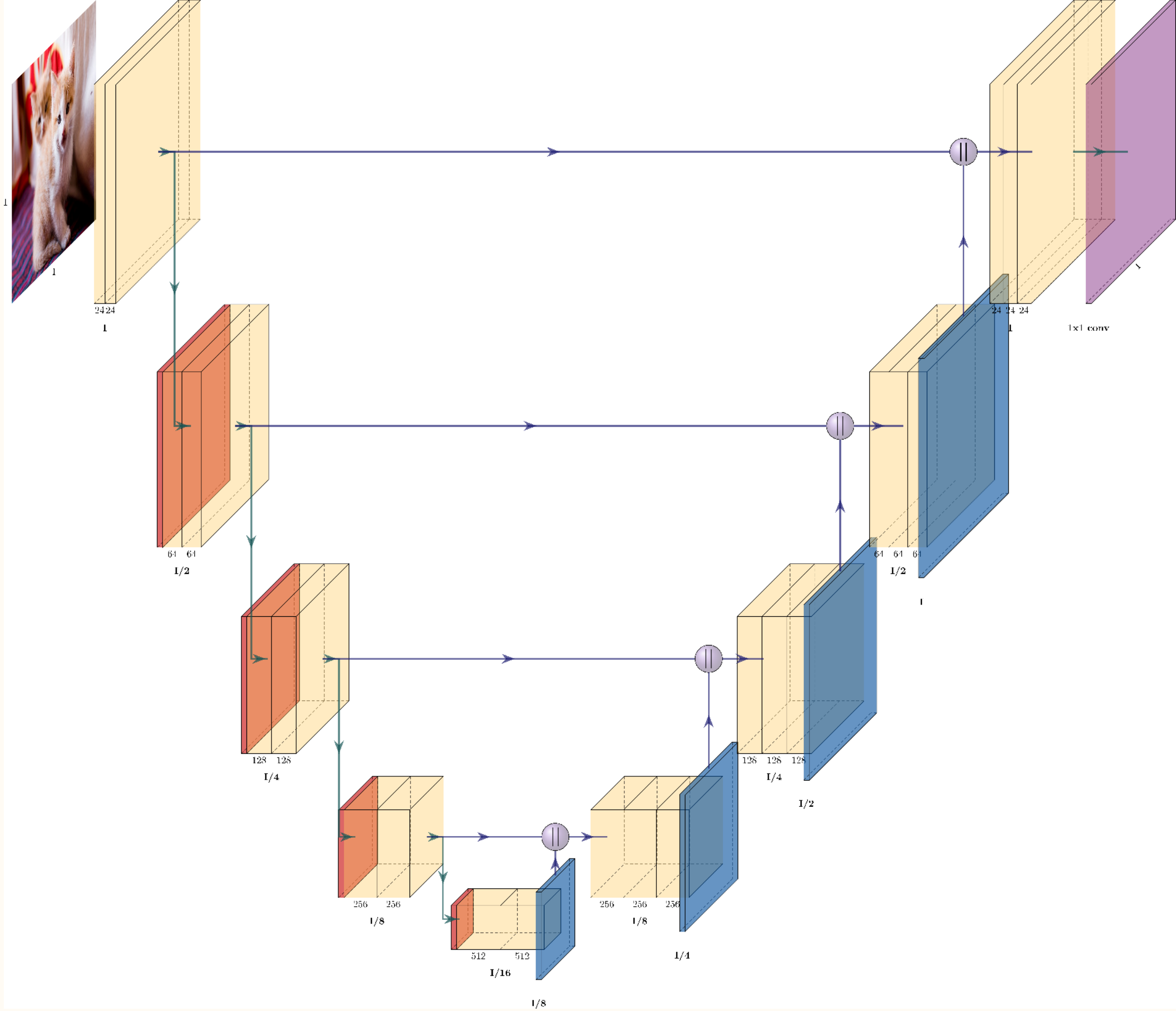
$$\mathcal{L}(\theta) = \sum_{l=1}^{N}\sum_{i=1}^{m}\sum_{j=1}^{n} \ell^{\mathrm{CE}}\big(f_\theta(X_l)_{ij}, (Y_l)_{ij}\big)$$

where ℓ^{CE} is the cross entropy loss.

U-Net

The U-Net architecture:

- Reduce the spatial dimension to obtain high-level (coarse scale) features
- Upsample or transpose convolution to restore spatial dimension.
- Use residual connections across each dimension reduction stage.



O. Ronneberger, P. Fischer, and T. Brox, U-Net: Convolutional networks for biomedical image segmentation, *Medical Image Computing and Computer-Assisted Intervention*, 2015.

Magnetic resonance imaging

Magnetic resonance imaging (MRI) is an inverse problem in which we partially* measure the Fourier transform of the patient and the goal is to reconstruct the patient's image.

So $X_{\text{true}} \in \mathbb{R}^n$ is the true original image (reshaped into a vector) with n pixels or voxels and $\mathcal{A}[X_{\text{true}}] \in \mathbb{C}^k$ with $k \ll n$. (If $k = n$, MRI scan can take hours.)

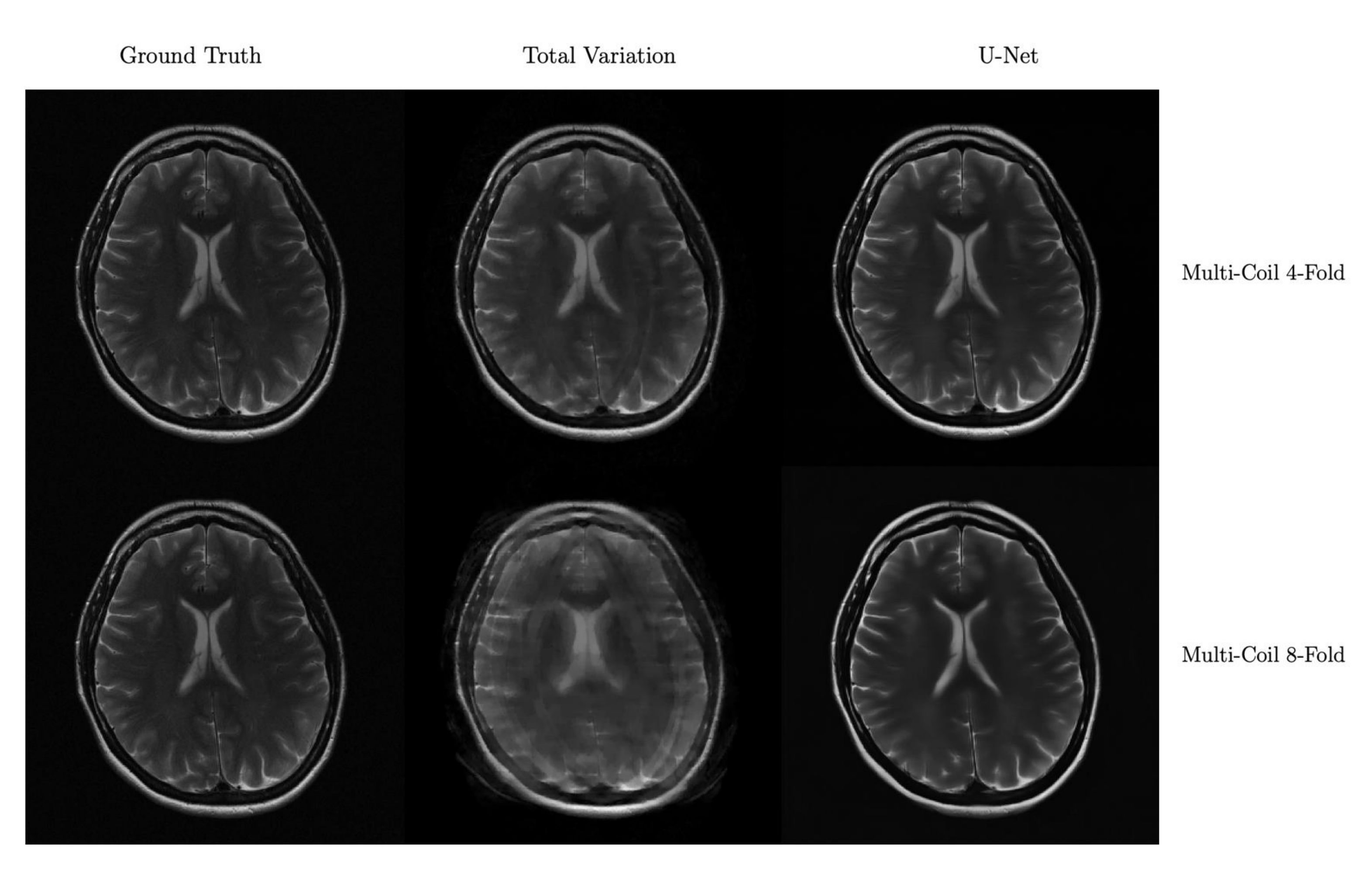
Classical reconstruction algorithms rely on Fourier analysis, total variation regularization, compressed sensing, and optimization.

Recent state-of-the-art use deep neural networks.

*We measure fewer points of the Fourier transform than there are pixels or voxels in the 2D or 3D image.

fastMRI dataset

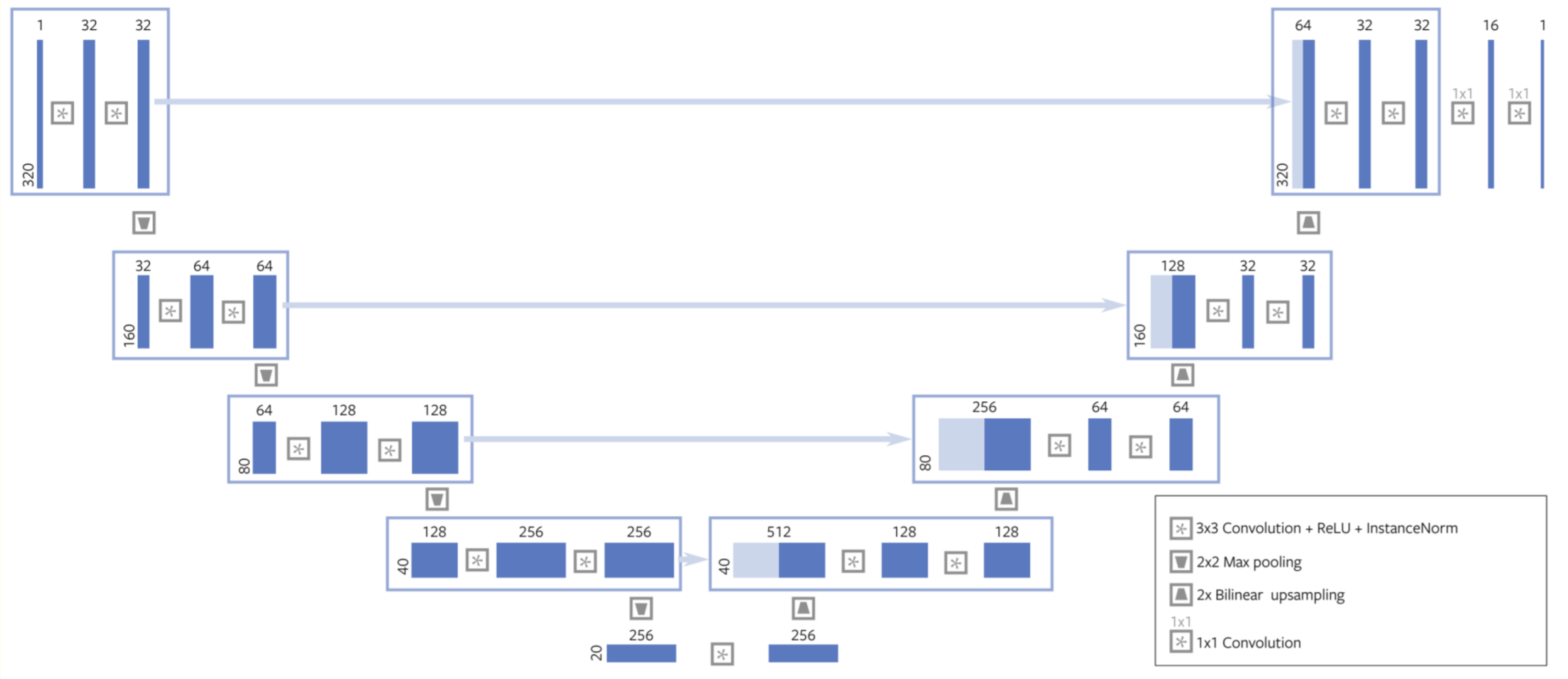
A team of researchers from Facebook AI Research and NYU released a large MRI dataset to stimulate data-driven deep learning research for MRI reconstruction.



J. Zbontar, F. Knoll, A. Sriram, T. Murrell, Z. Huang, M. J. Muckley, A. Defazio, R. Stern, P. Johnson, M. Bruno, M. Parente, K. J. Geras, J. Katsnelson, H. Chandarana, Z. Zhang, M. Drozdzal, A. Romero, M. Rabbat, P. Vincent, N. Yakubova, J. Pinkerton, D. Wang, E. Owens, C. L. Zitnick, M. P. Recht, D. K. Sodickson, and Y. W. Lui, fastMRI: An open dataset and benchmarks for accelerated MRI, *arXiv*, 2019.

U-Net for inverse problems

Although U-Net was originally proposed as an architecture for semantic segmentation, it is also being used widely as one of the default architectures in inverse problems, including MRI reconstruction.



J. Zbontar, F. Knoll, A. Sriram, T. Murrell, Z. Huang, M. J. Muckley, A. Defazio, R. Stern, P. Johnson, M. Bruno, M. Parente, K. J. Geras, J. Katsnelson, H. Chandarana, Z. Zhang, M. Drozdzal, A. Romero, M. Rabbat, P. Vincent, N. Yakubova, J. Pinkerton, D. Wang, E. Owens, C. L. Zitnick, M. P. Recht, D. K. Sodickson, and Y. W. Lui, fastMRI: An open dataset and benchmarks for accelerated MRI, *arXiv*, 2019.

Computational tomography

Computational tomography (CT) is an inverse problem in which we partially* measure the Radon transform of the patient and the goal is to reconstruct the patient's image.

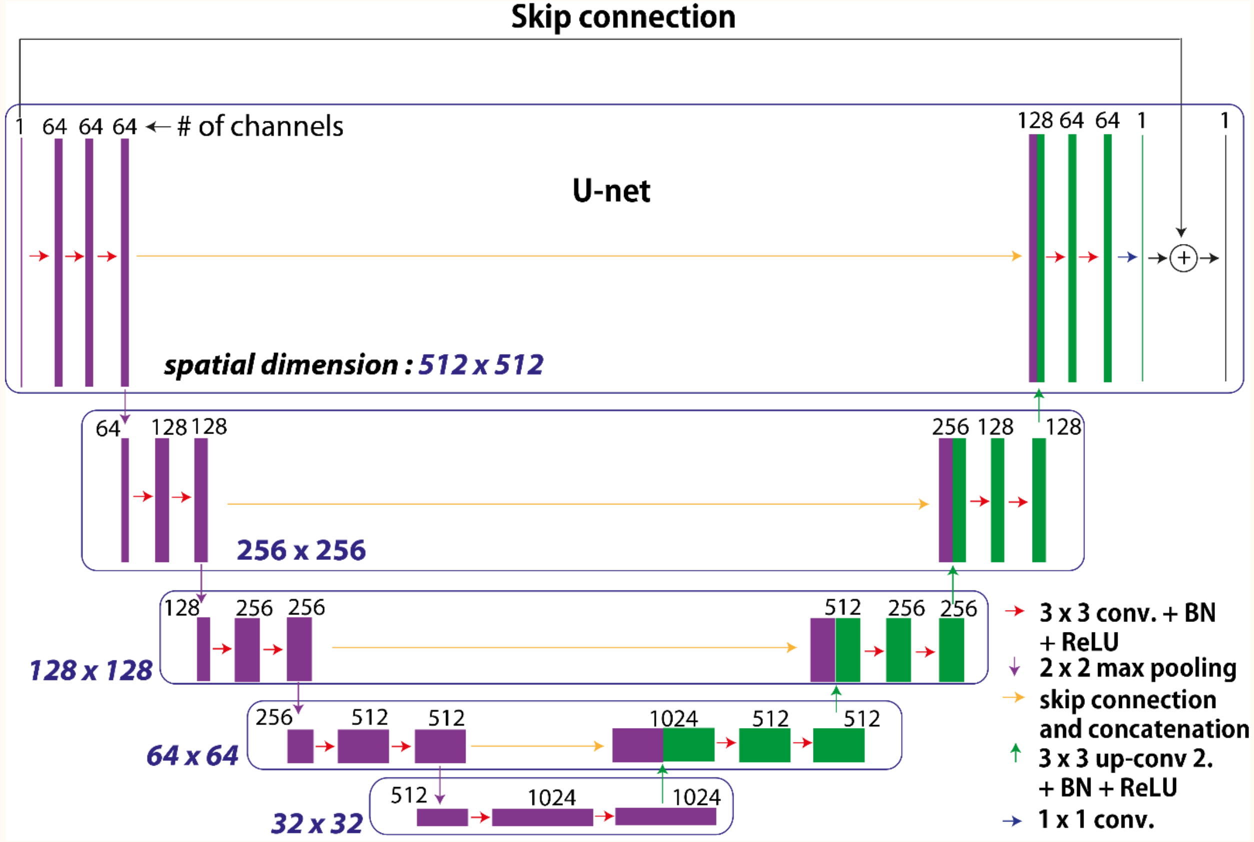
So $X_{\mathrm{true}} \in \mathbb{R}^n$ is the true original image (reshaped into a vector) with n pixels or voxels and $\mathcal{A}[X_{\mathrm{true}}] \in \mathbb{R}^k$ with $k \ll n$. (If $k = n$, the X-ray exposure to perform the CT scan can be harmful.)

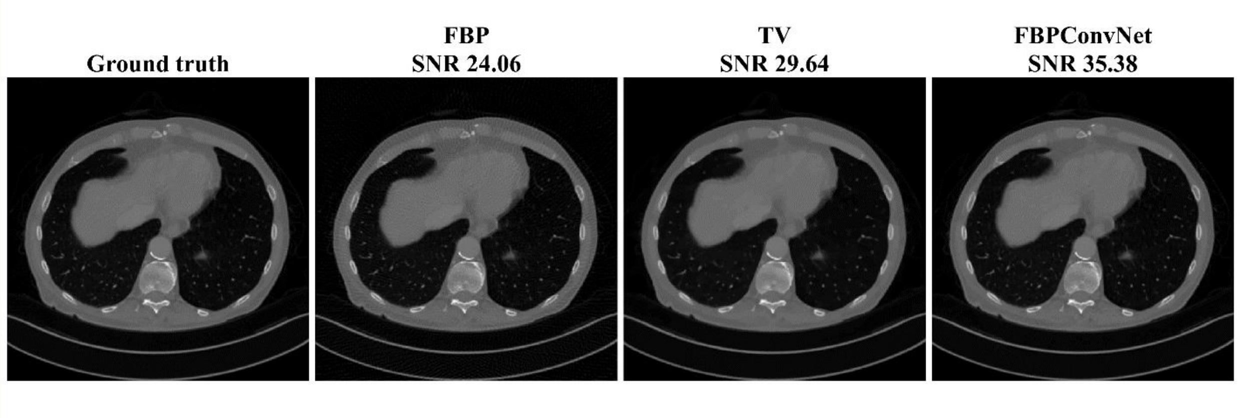
Recent state-of-the-art use deep neural networks.

*We measure fewer points of the Radon transform than there are pixels or voxels in the 2D or 3D image.

U-Net for CT reconstruction

U-Net is also used as one of the default architectures in CT reconstruction





K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, Deep convolutional neural network for inverse problems in imaging, *IEEE TIP*, 2017.