Lecture 5, Deep Unfolding

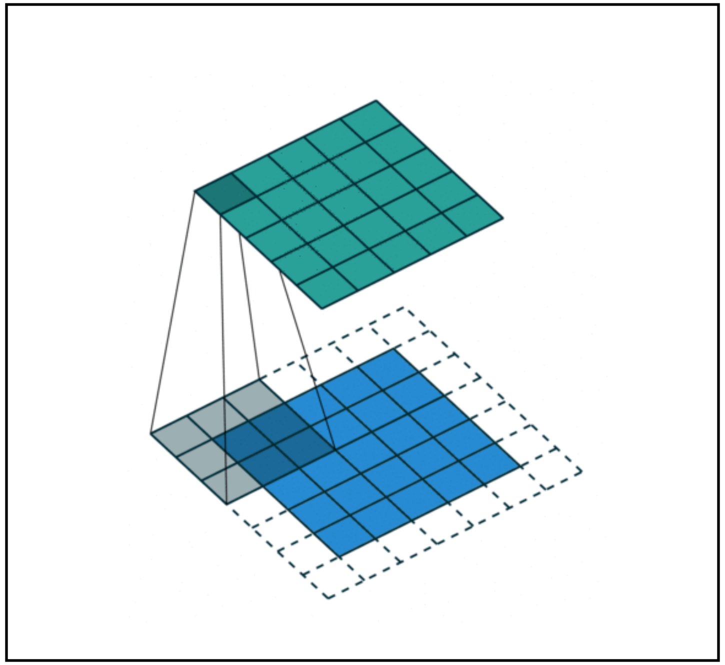
Change of notation:
Let $m \in M$ be a model choice
$p(\theta, m) = p(\theta)p(m)$

Do we have any prior preferences for the model choice?

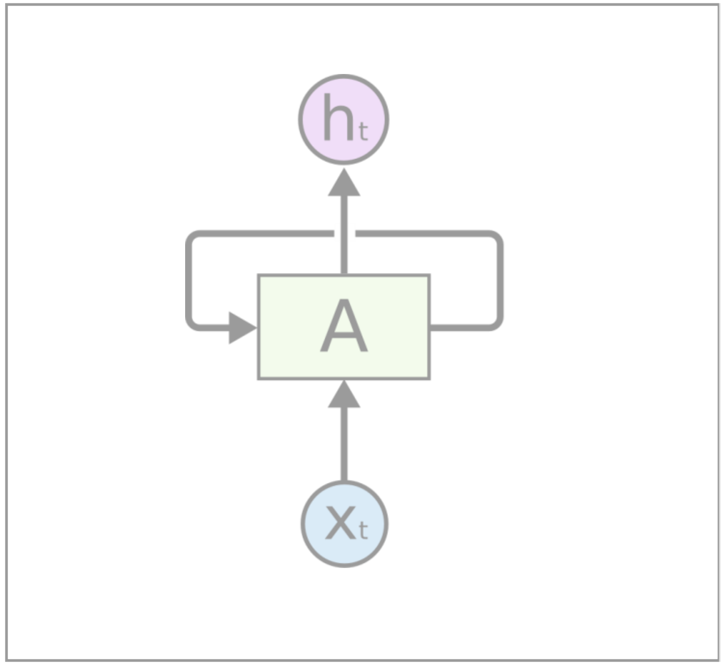
$$p(\theta, m|D) = \frac{p(D|\theta, m)}{p(D)} p(\theta) p(m)$$

What are the best model parameters?
And what is the best model choice?

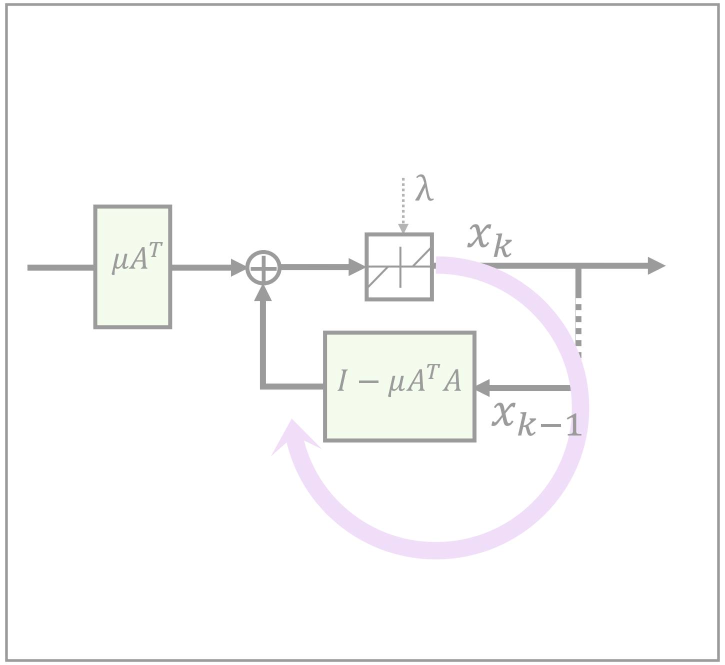
Architectural priors



(Group) convolutional neural networks

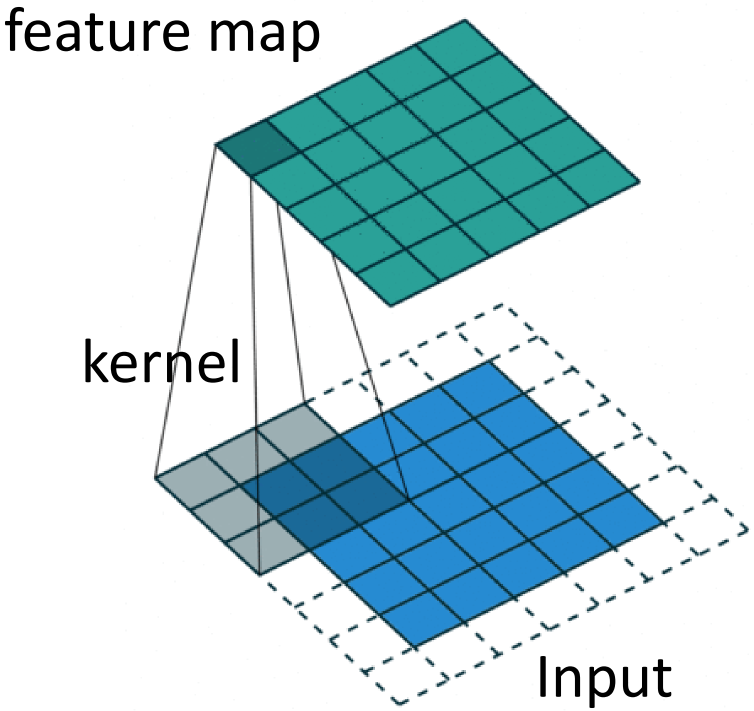


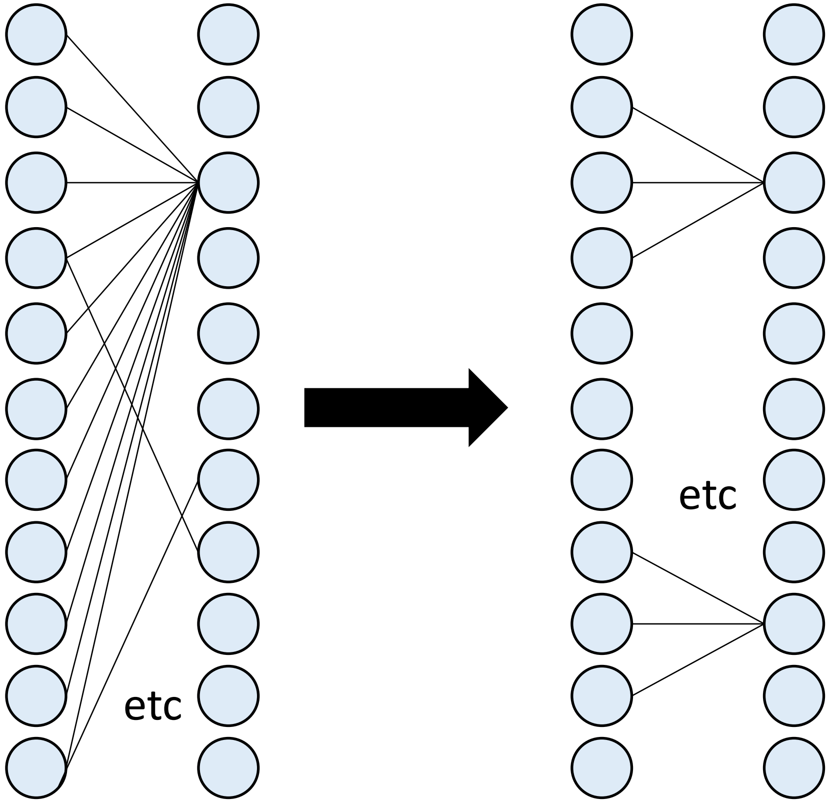
Recurrent neural networks



Deep unfolding

Convolutional networks





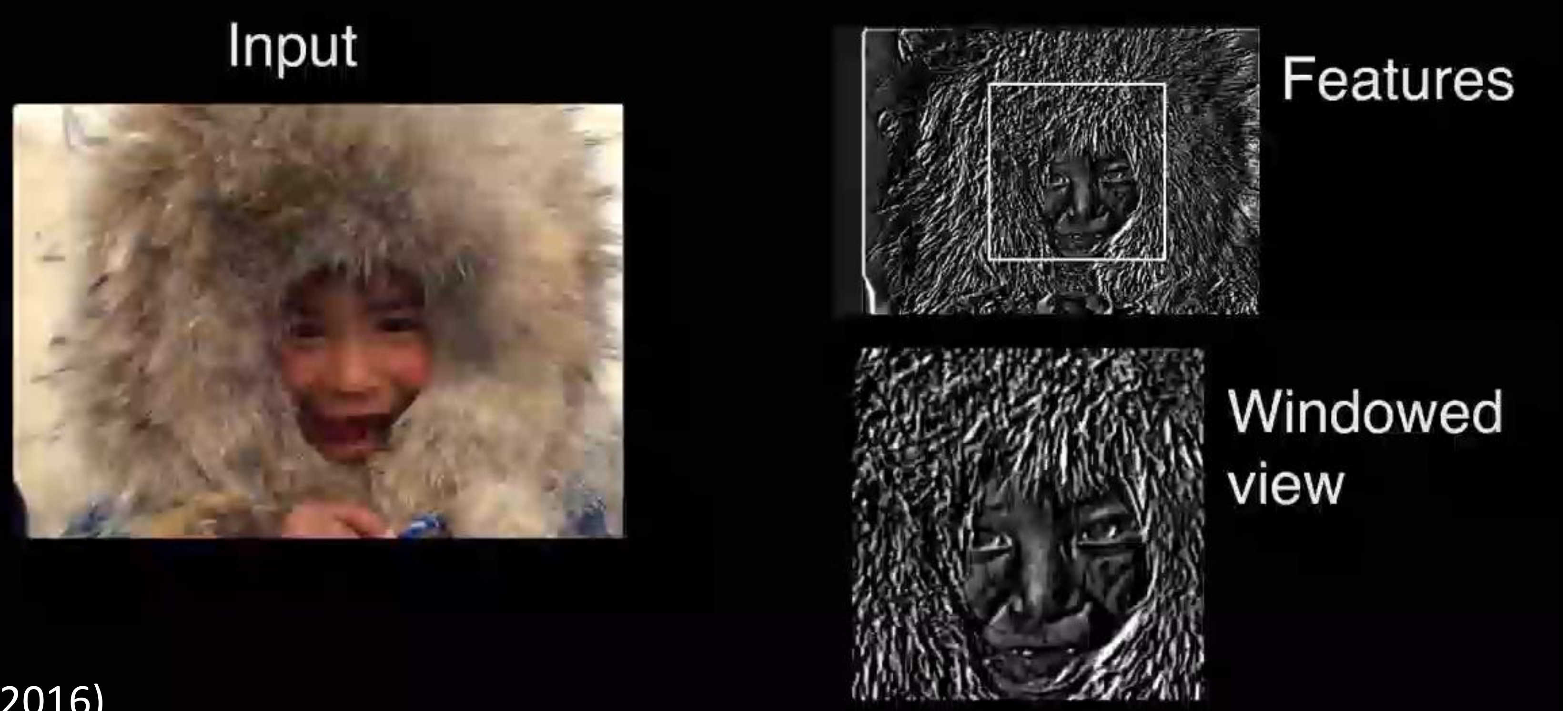
Fully connected
independent weights

1D convolutional
(kernel 3)

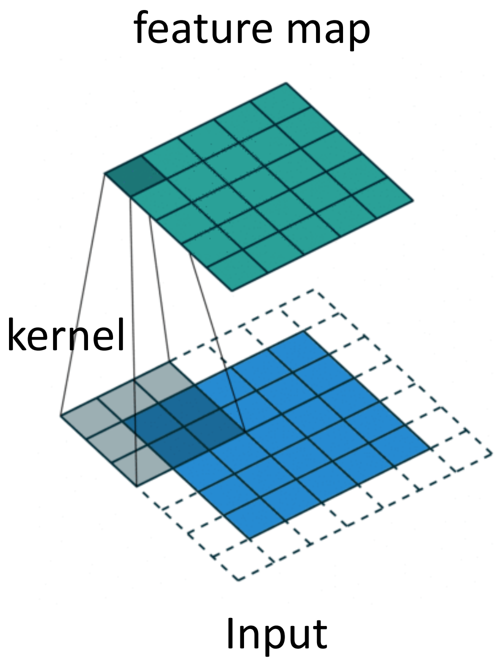
Locally connected + shared weights

Convolutional networks

Strong prior: translational invariance



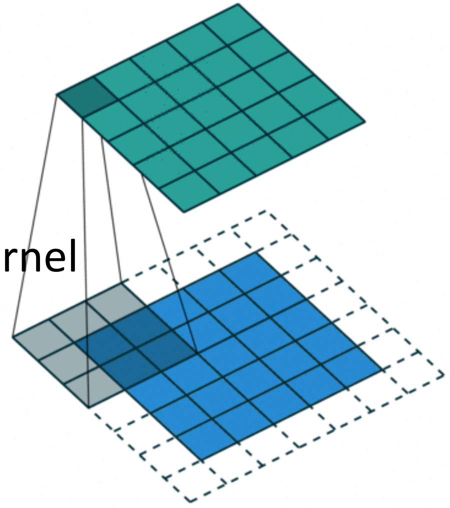
(Cohen, 2016)



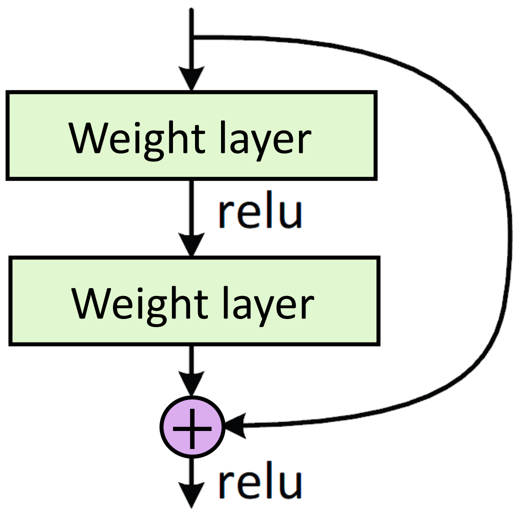
Convolutional networks

Not invariant to rotation! …data augmentation/spherical harmonics

(Cohen, 2016)

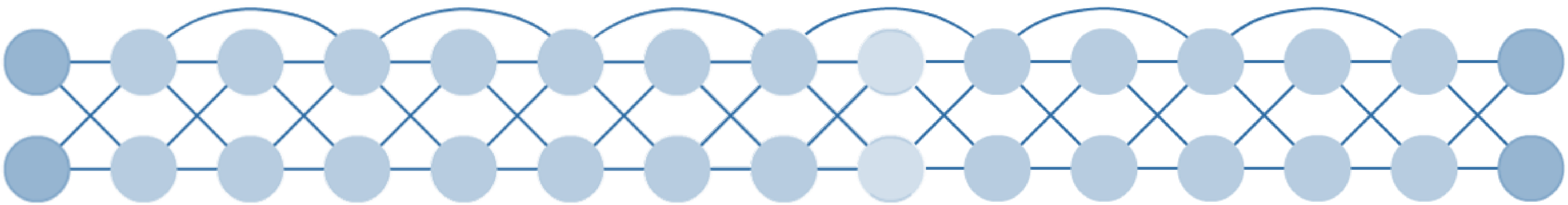


Residual networks



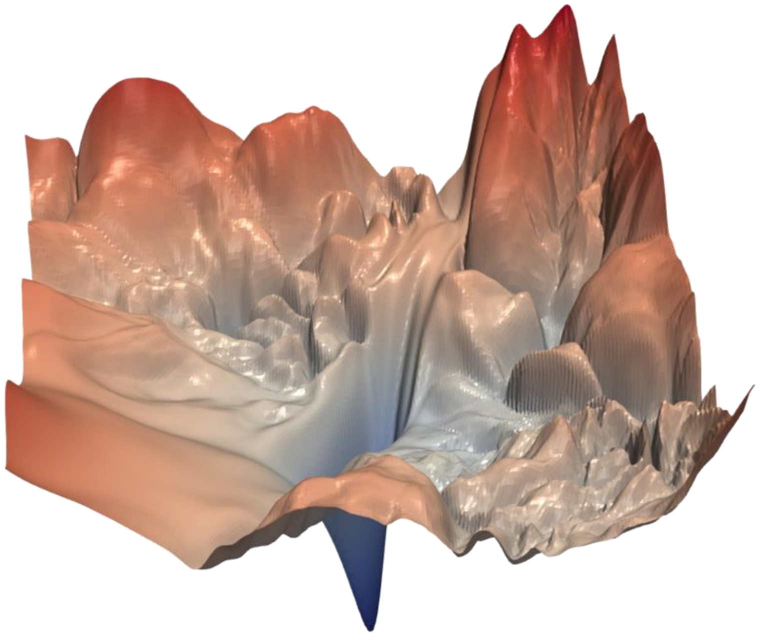
Residual connections

Much more efficient training with faster convergence

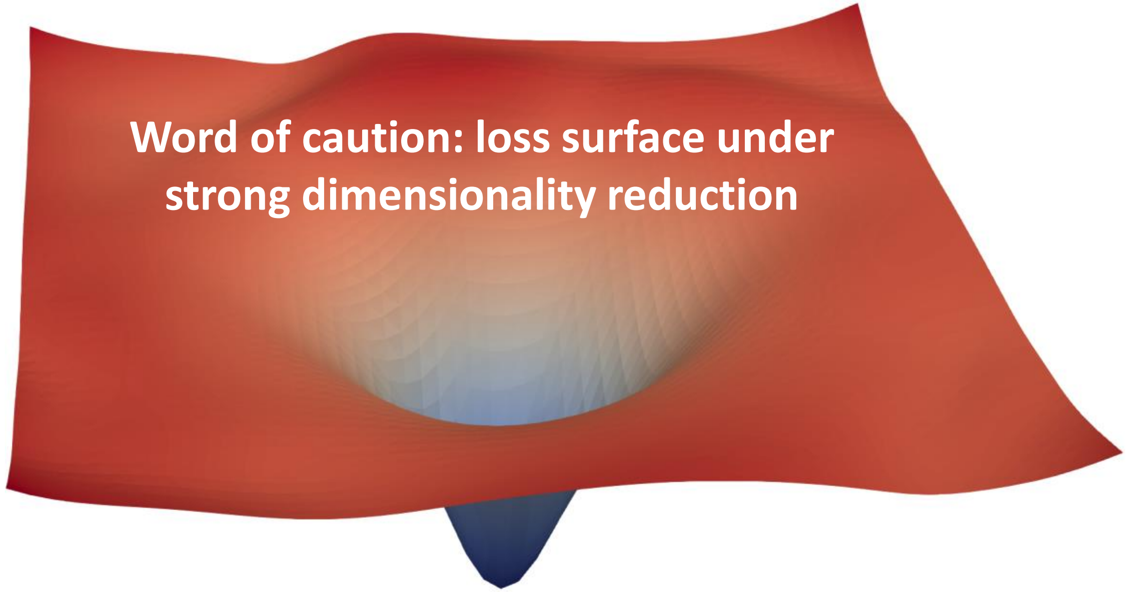


Residual networks

ResNet-56
Without residual connections



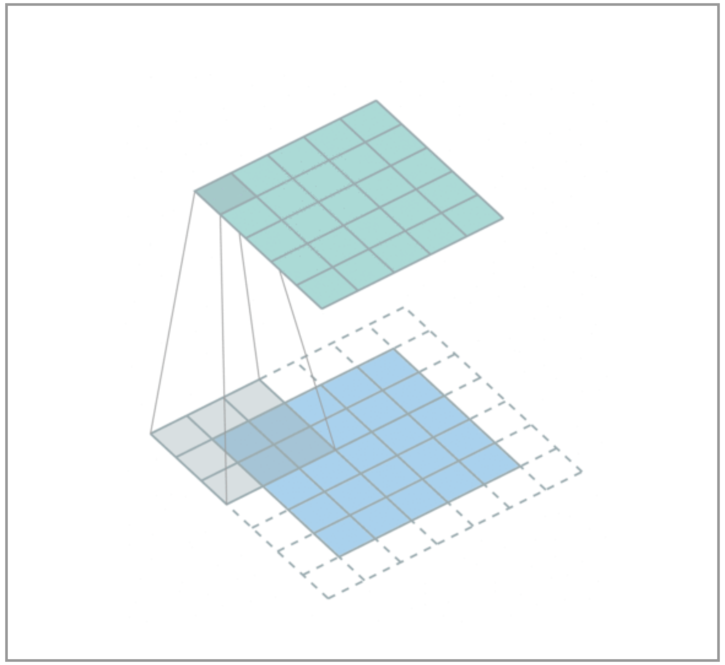
ResNet-56
With residual connections



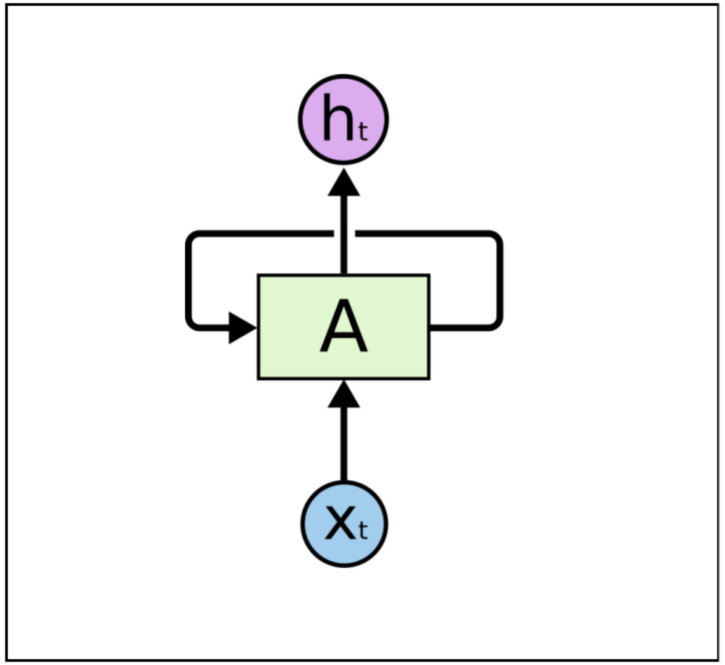
Priors matter! Some neural architectures are easier to minimize than others

Source: Goldstein et al., NeurIPS 2018

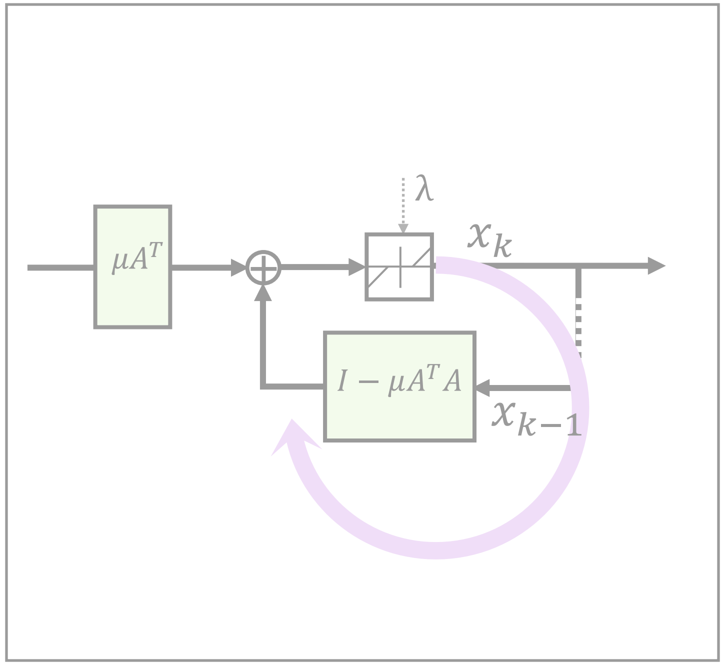
Architectural priors



(Group) convolutional neural networks

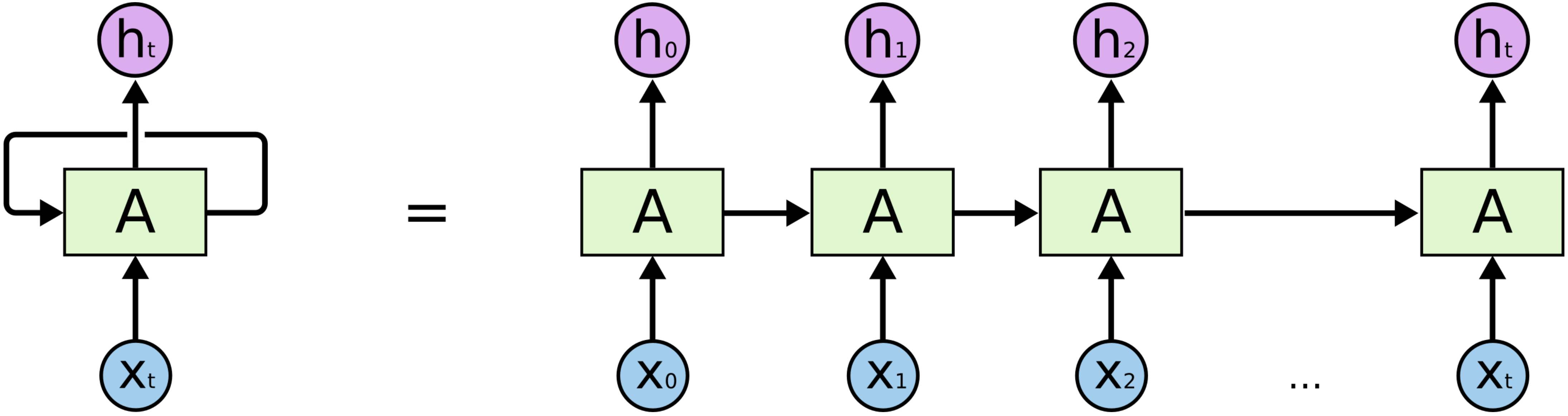


Recurrent neural networks



Deep unfolding

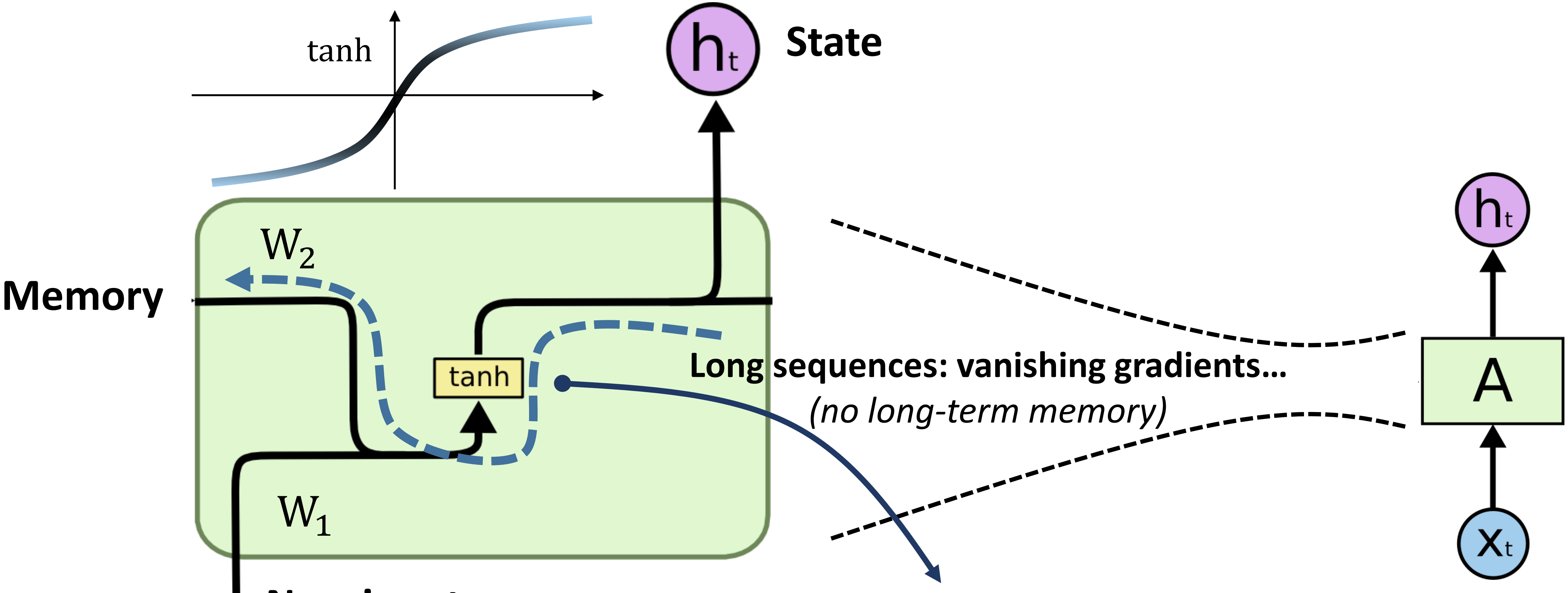
Recurrent networks



An unrolled recurrent neural network

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent networks



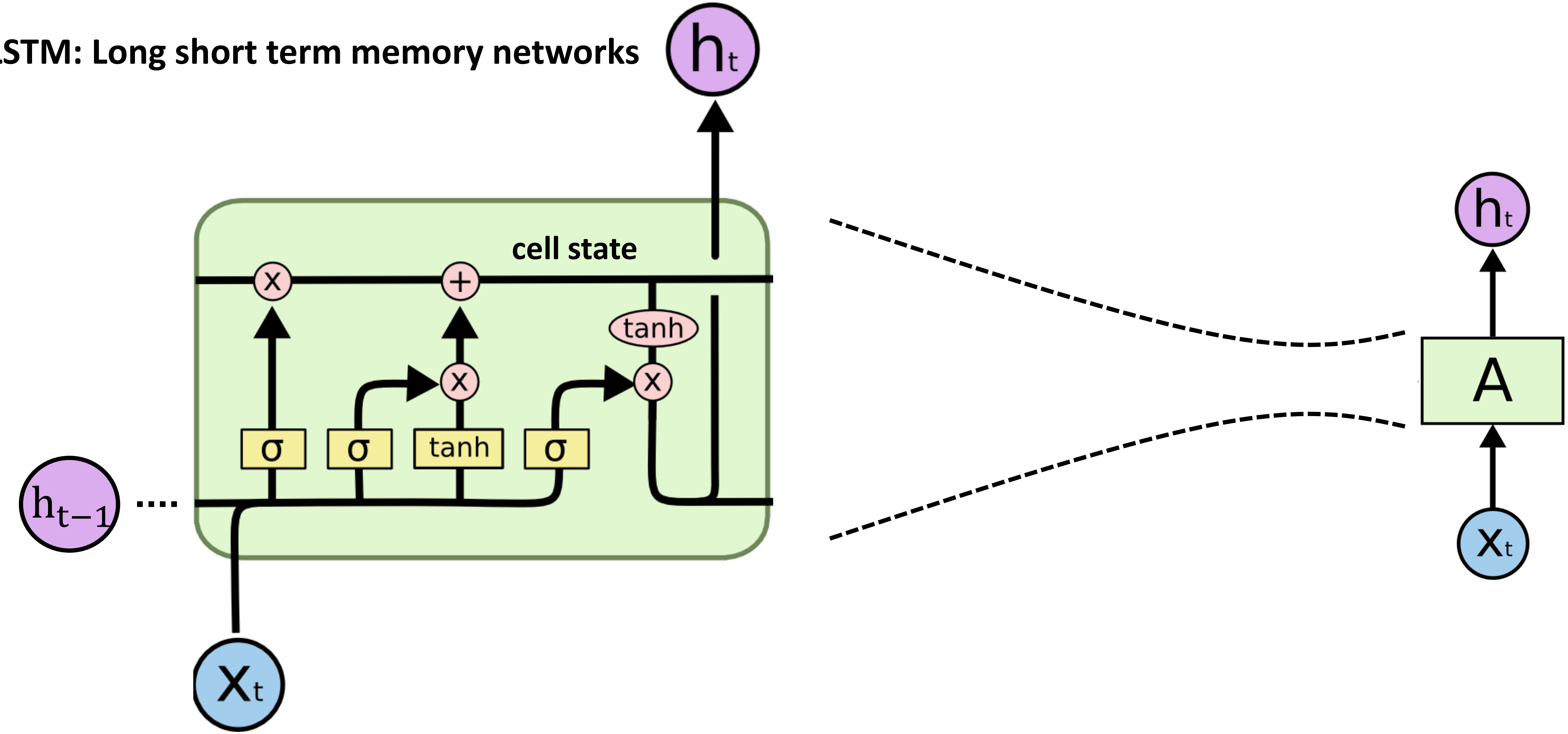
Neural network layer with memory:

$$h_t = \tanh(W_1 x_t + W_2 h_{t-1})$$

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent networks

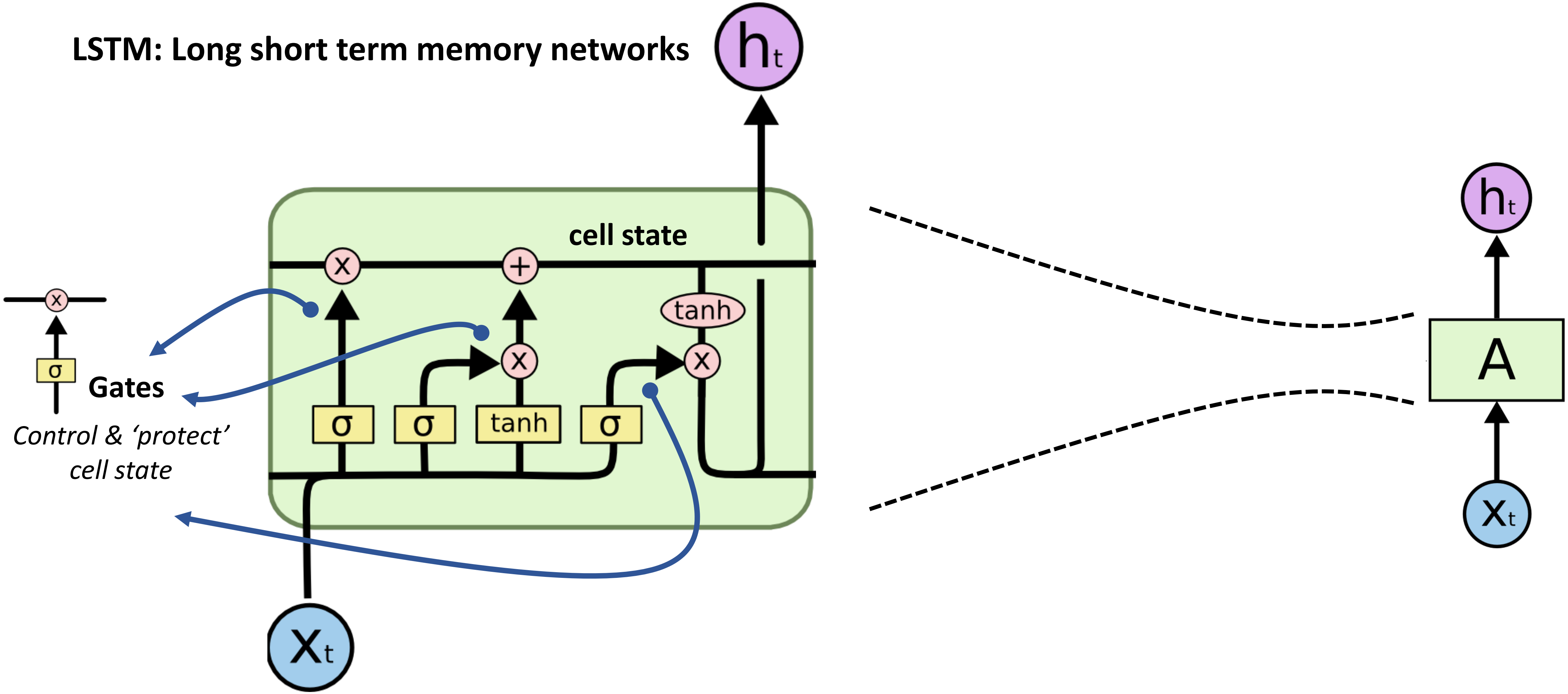
LSTM: Long short term memory networks



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent networks

LSTM: Long short term memory networks

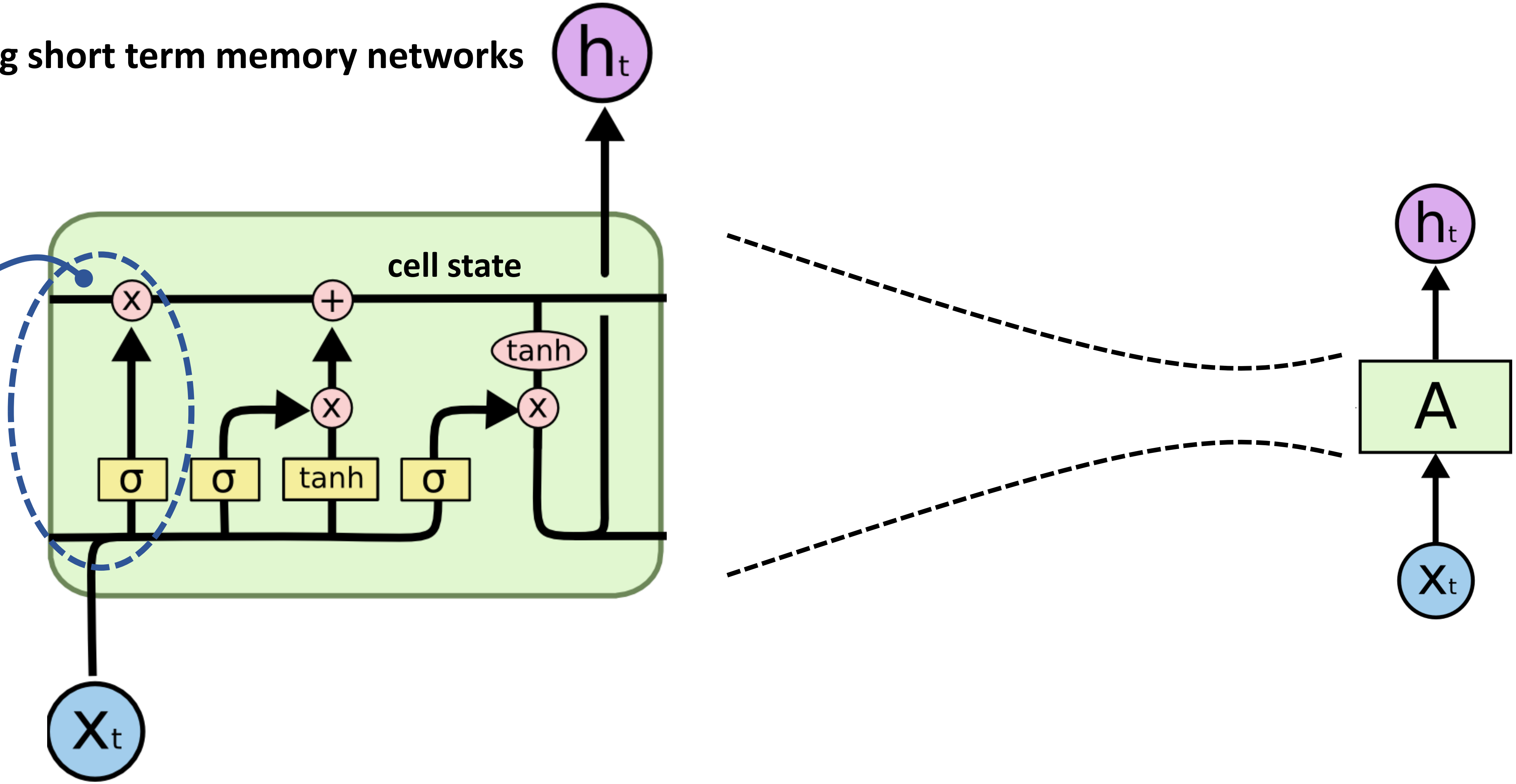


http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent networks

LSTM: Long short term memory networks

Which cell state information is preserved?

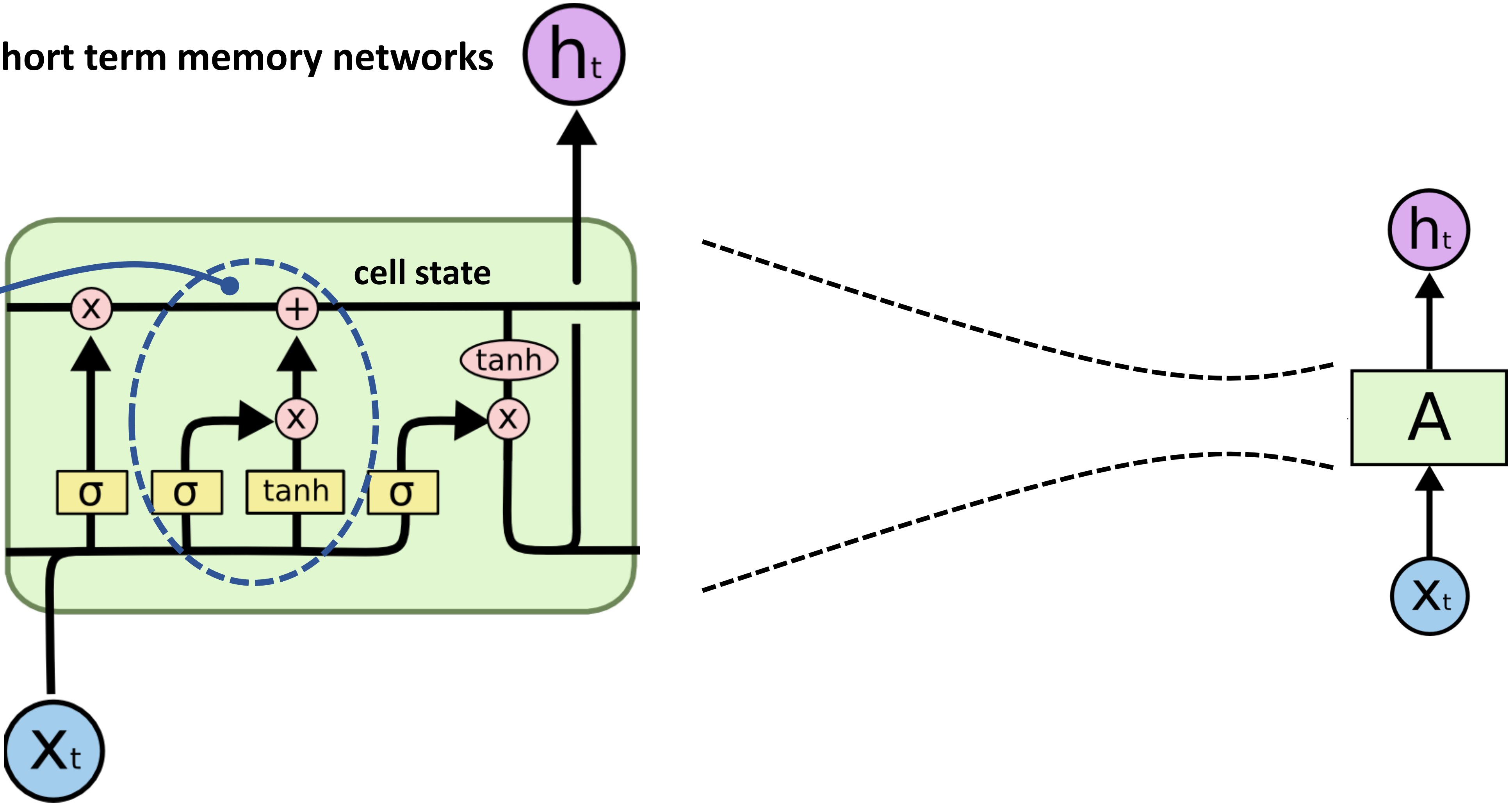


http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent networks

LSTM: Long short term memory networks

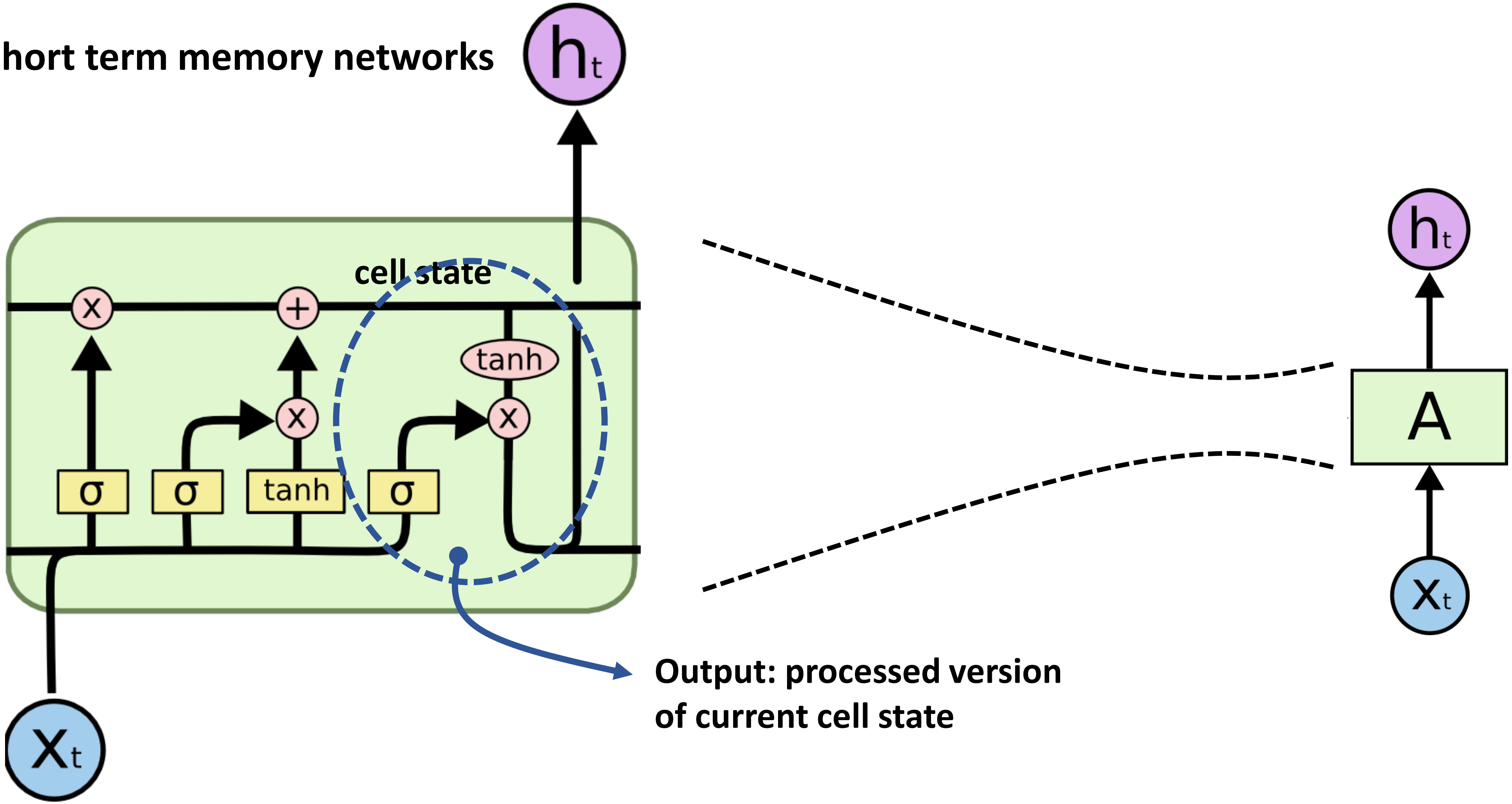
What new information is added?



http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Recurrent networks

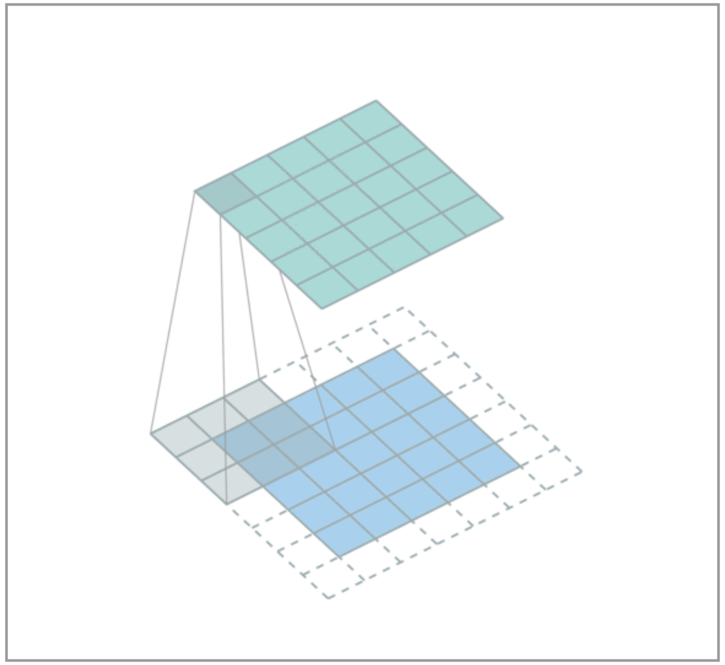
LSTM: Long short term memory networks



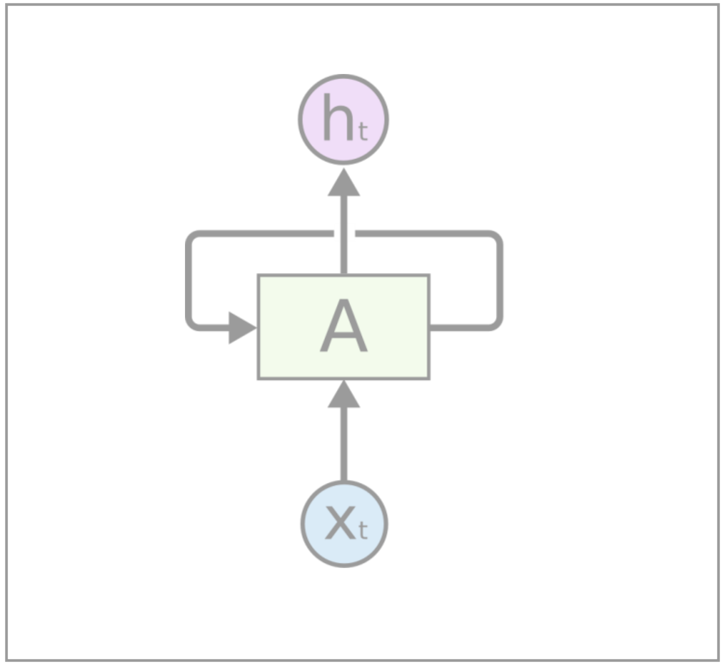
Output: processed version of current cell state

http://colah.github.io/posts/2015-08-Understanding-LSTMs/

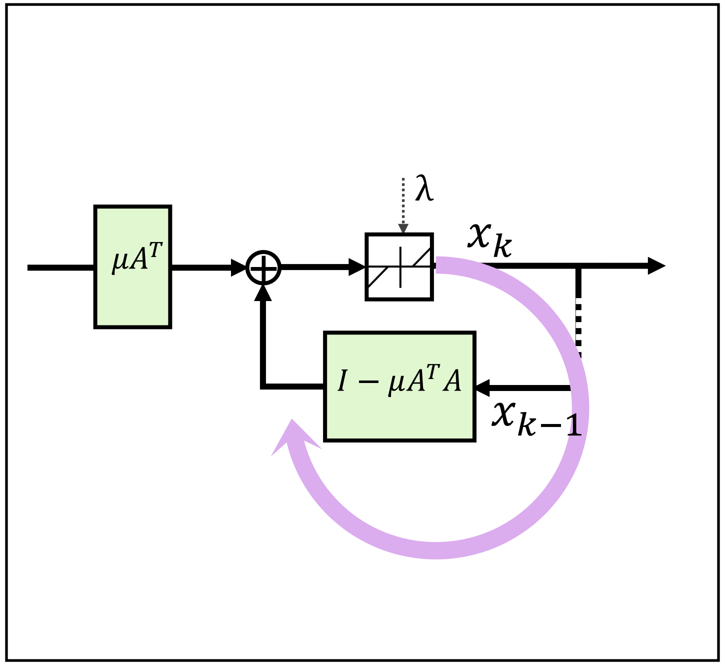
Architectural priors



(Group) convolutional neural networks



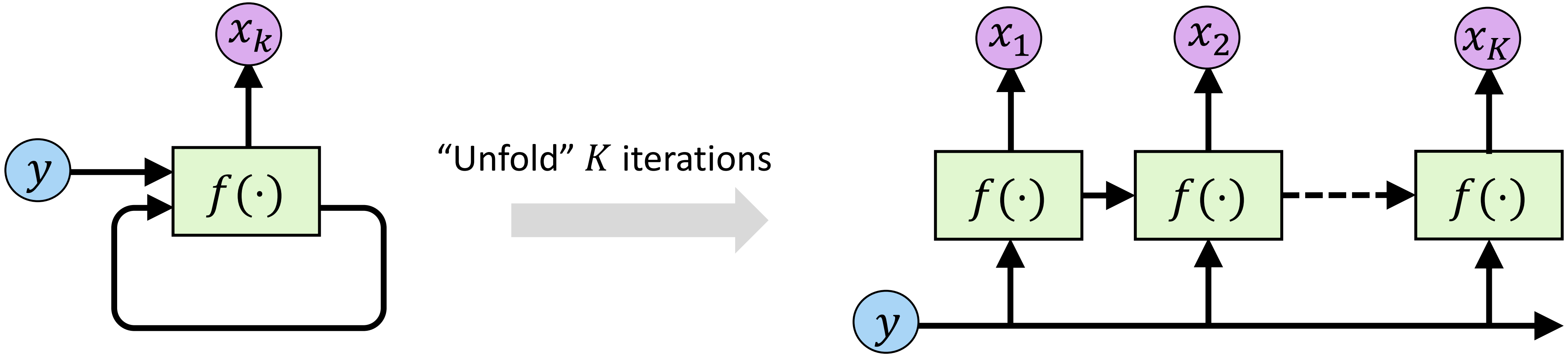
Recurrent neural networks



Deep unfolding

Deep unfolding/unrolling

Explicit embedding of structural signal priors in deep networks

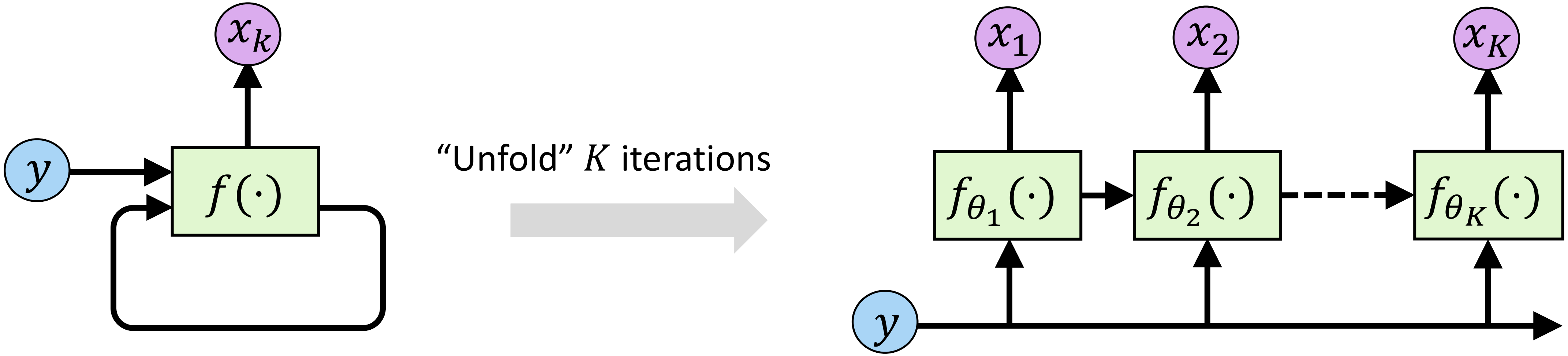


Iterative model-based algorithm with input y and output x that ***leverages some signal structure***

unfolded model-based algorithm with input y and output x that ***leverages some signal structure***

Deep unfolding

Explicit embedding of structural signal priors in deep networks



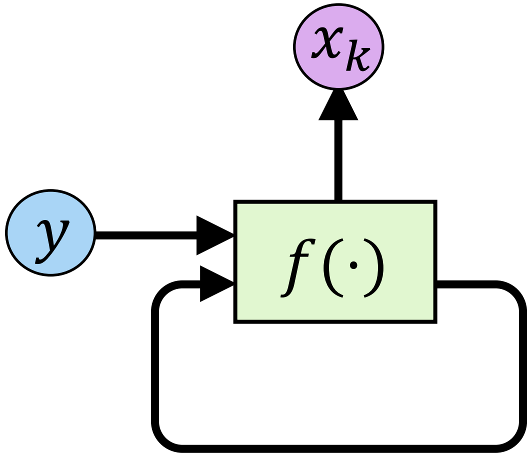
Iterative model-based algorithm with input y and output x that ***leverages some signal structure***

unfolded model-based algorithm ***with learned parameters*** *that* ***leverages some signal structure***

Deep unfolding

Example: sparse coding

Many applications: denoising, compressed sensing, image reconstruction, super-resolution, …



Iterative model-based algorithm with input y and output x that ***leverages signal sparsity***

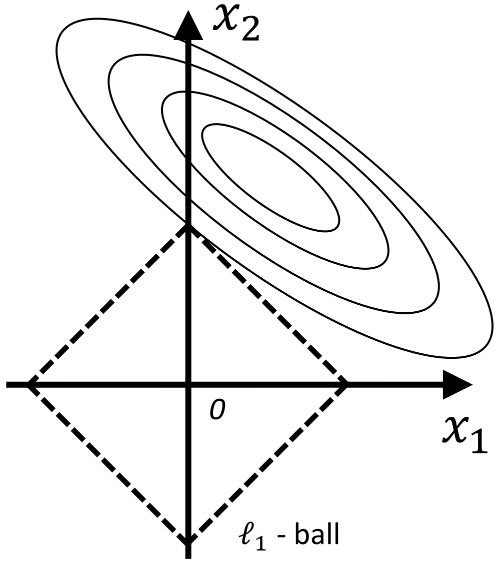
Sparse coding problem

$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$ with $\mathbf{x}$ being sparse

Find $\mathbf{x}$:

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\text{minimize}} \; \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{x}\|_1$$

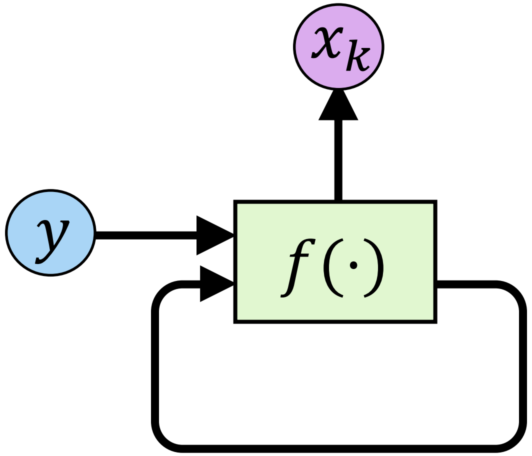
Some intuition:



Deep unfolding

Example: sparse coding

Many applications: denoising, compressed sensing, image reconstruction, super-resolution, …



Iterative model-based algorithm with input y and output x that ***leverages signal sparsity***

Sparse coding problem

$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$ with $\mathbf{x}$ being sparse

Find $\mathbf{x}$:

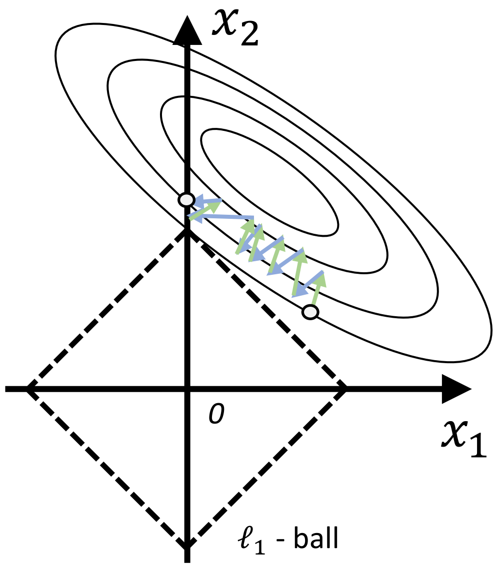
$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\text{minimize}} \; \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{x}\|_1$$

Proximal gradient methods

Iterative shrinkage and thresholding

1. Take a gradient step towards $\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 = 0$
2. Move it in the proximity of $\lambda\|\mathbf{x}\|_1 = 0$

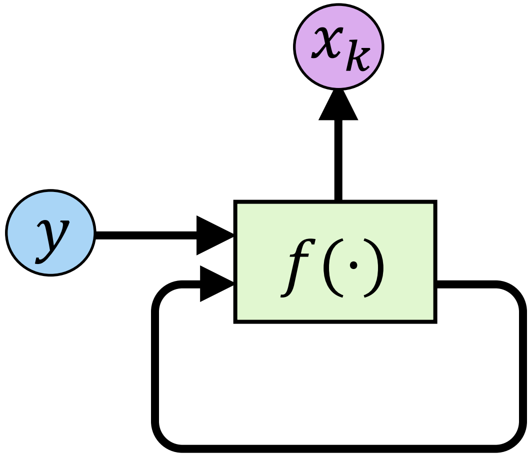
Some intuition:



Deep unfolding

Example: sparse coding

Many applications: denoising, compressed sensing, image reconstruction, super-resolution, …



Iterative model-based algorithm with input y and output x that ***leverages signal sparsity***

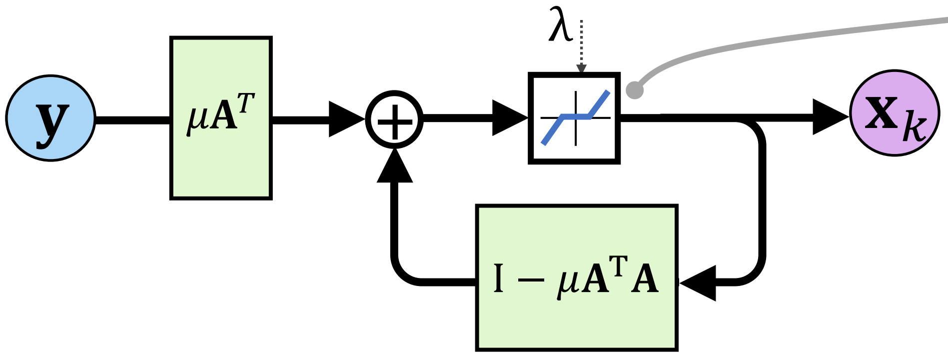
Sparse coding problem

$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$ with $\mathbf{x}$ being sparse

Find $\mathbf{x}$:

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\text{minimize}} \ \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{x}\|_1$$

Iterative shrinkage and thresholding

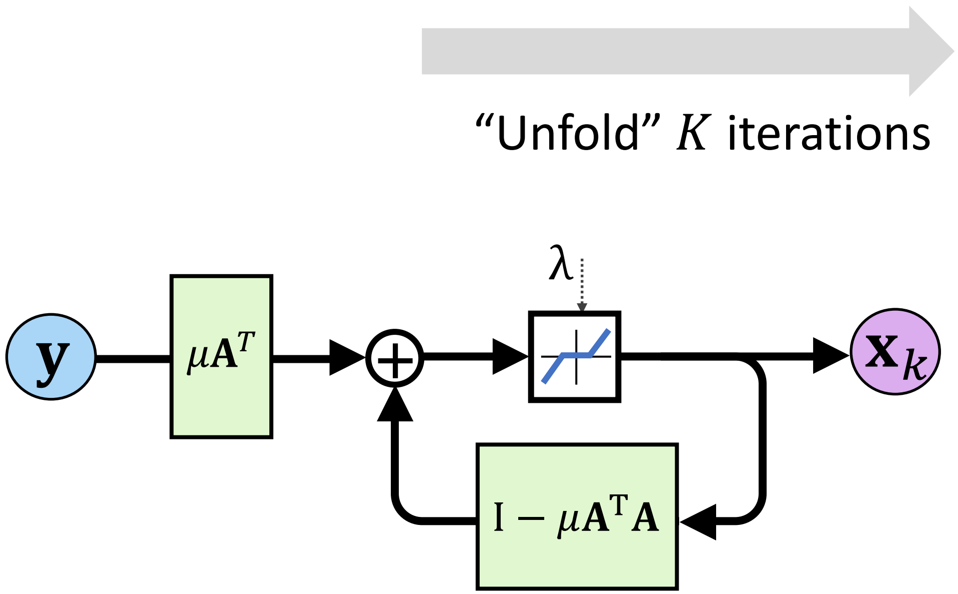


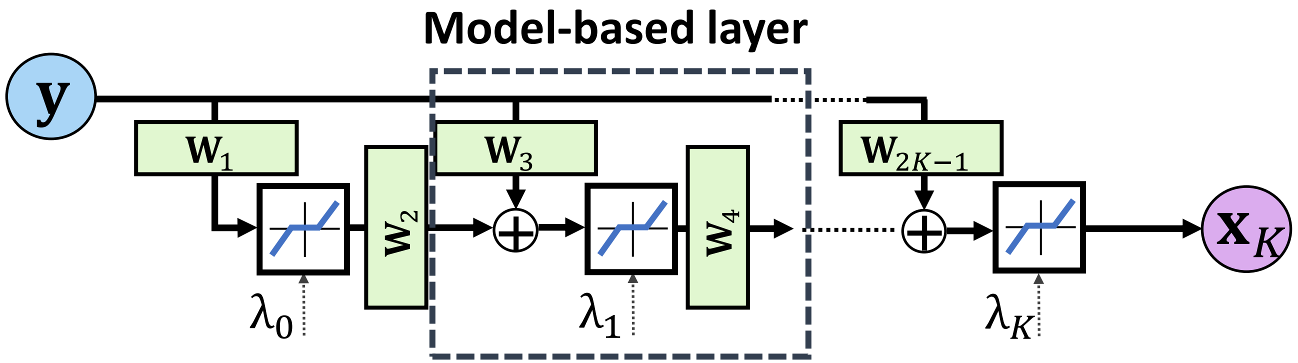
Proximity operator
Soft thresholding

Deep unfolding

Example: sparse coding

Many applications: denoising, compressed sensing, image reconstruction, super-resolution, …

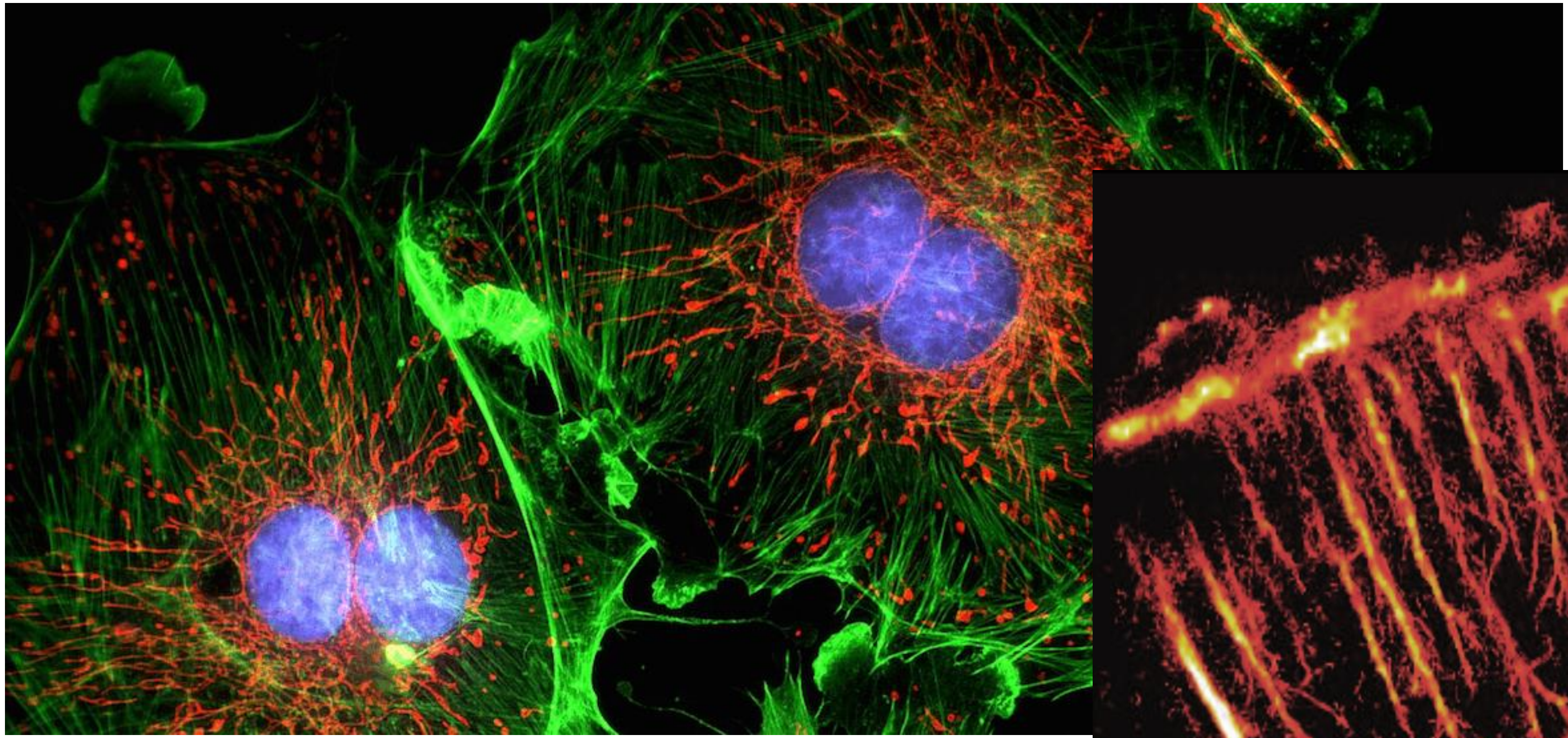




Deep learning with a model-based signal prior

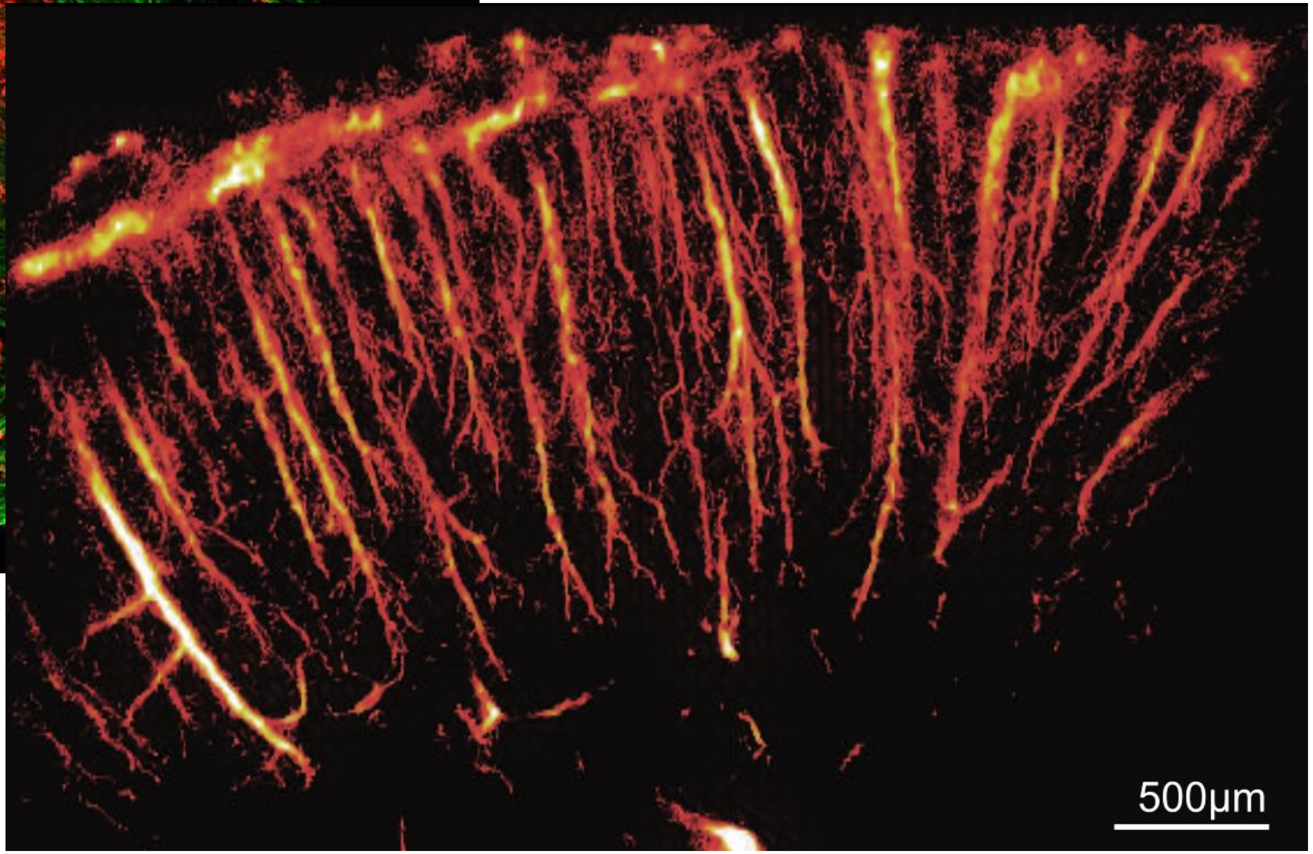
- Robust, intuitive, interpretable
- Fast learning
- Data efficient
- Low complexity and memory footprint

DL for super resolution microscopy



Optical fluorescence microscopy

Ultrasound localization microscopy

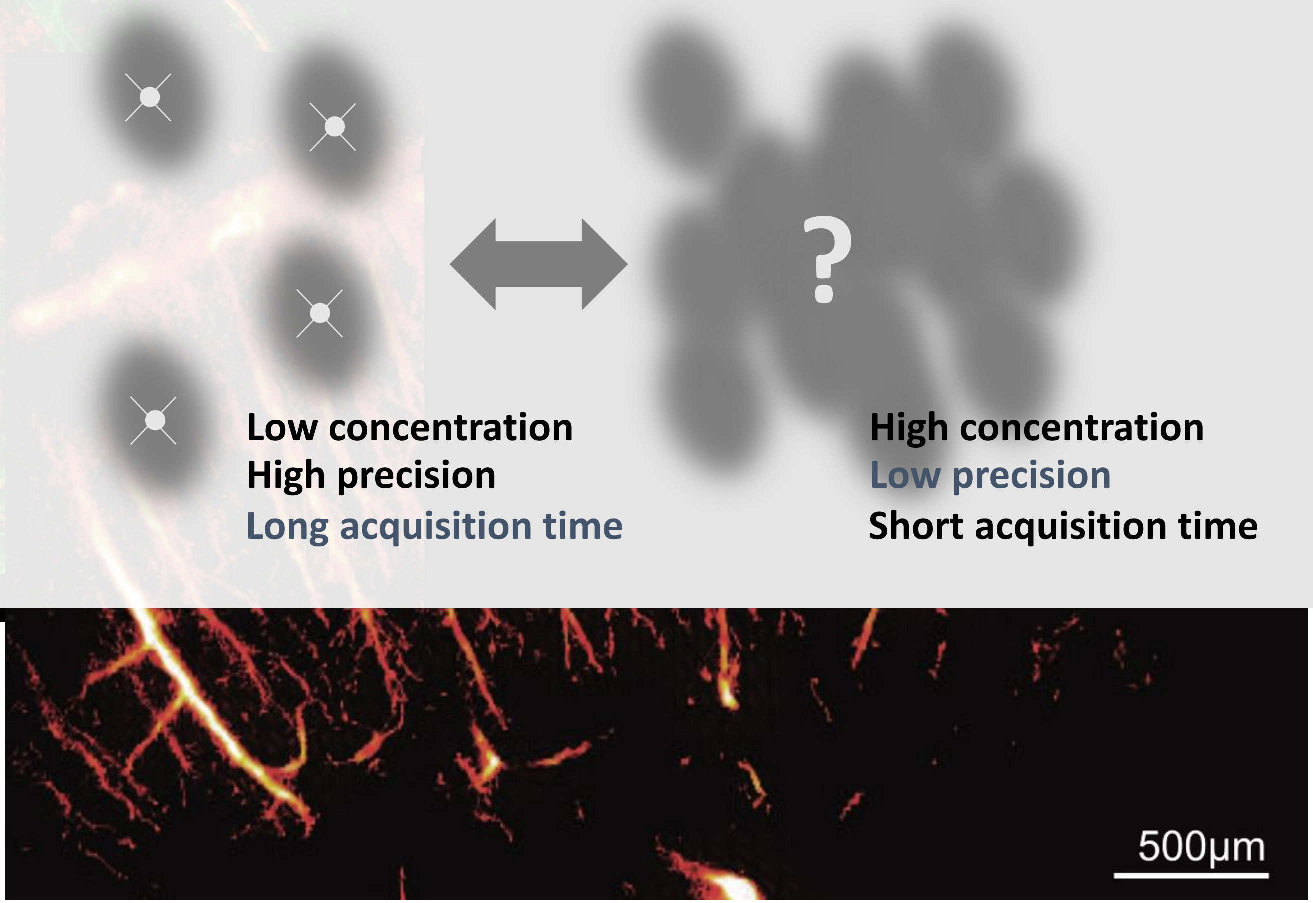


DL for super resolution microscopy

Acquisition time

Indicator concentration

Localization precision



Deep unfolded sparse coding

Learned convolutional ISTA

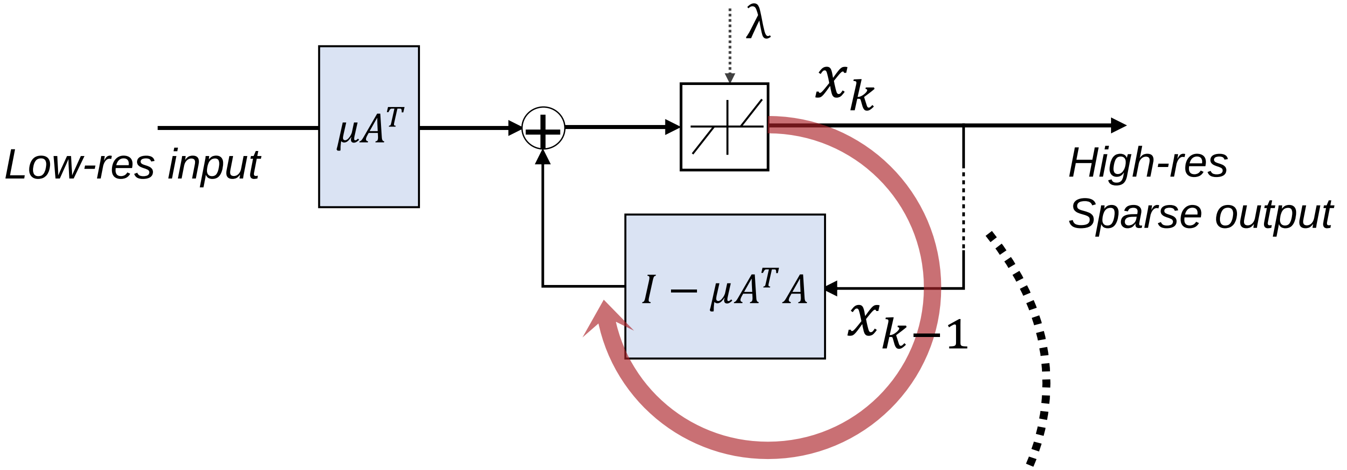
$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\text{minimize}} \; \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{x}\|_1$$

$\mathbf{A}$: Known measurement (PSF) matrix

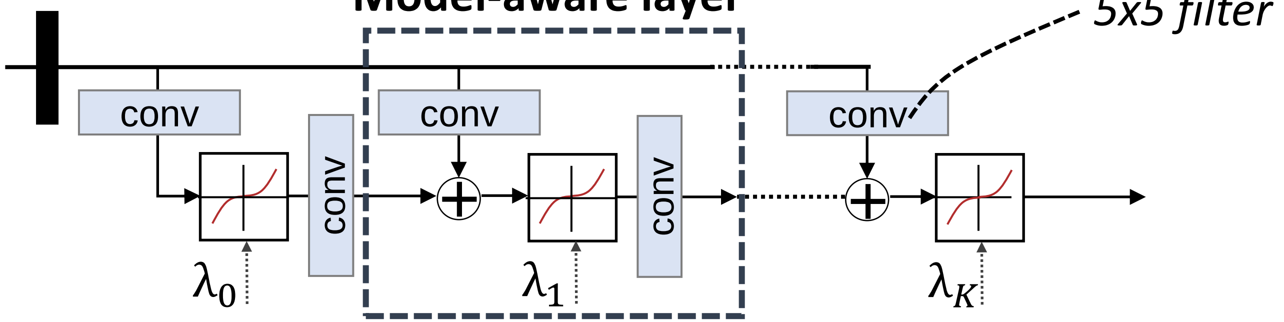
$\mathbf{y}$: Measurement

$\mathbf{x}$: Sparse target location vector

Iterative shrinkage and thresholding (ISTA)



Deep unfolded ISTA



Thousands of iterations

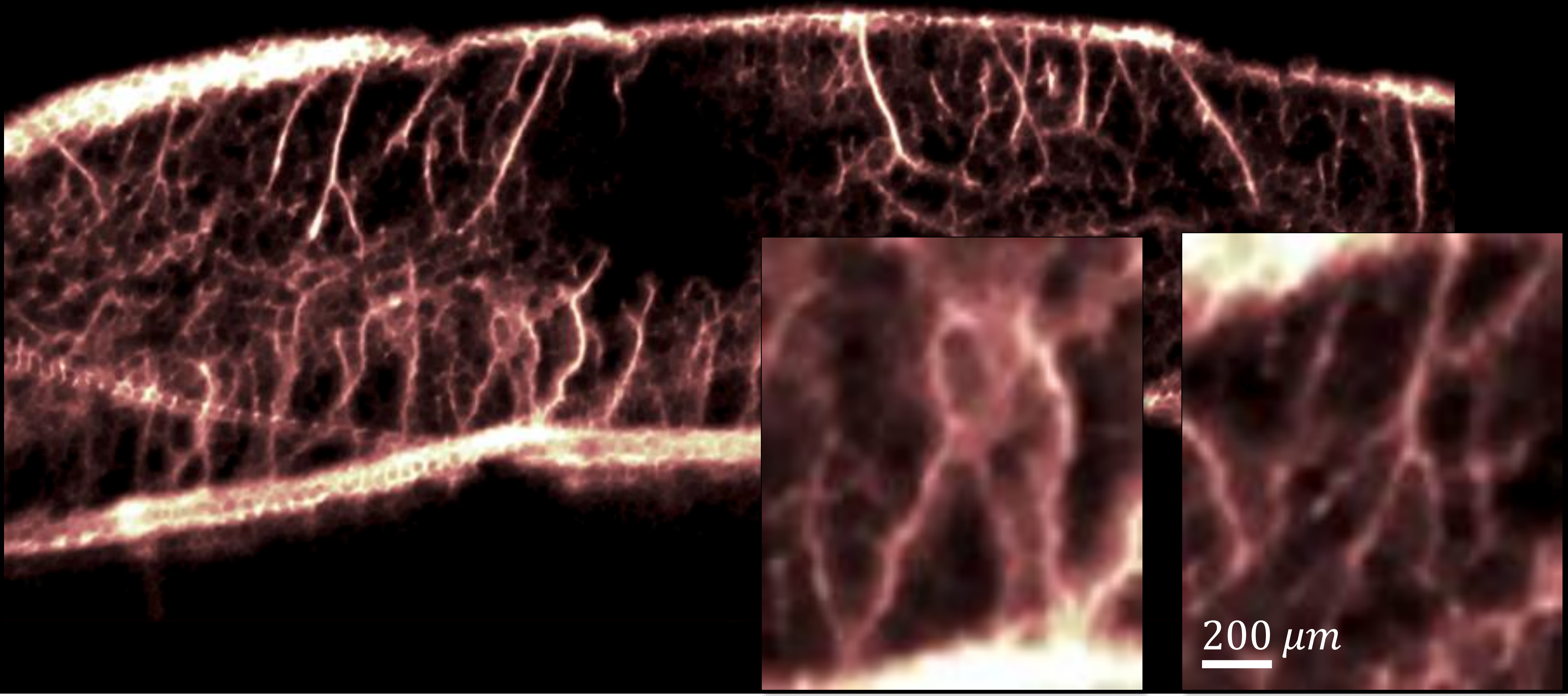
Deep unfolding

Only 10 unfolded iterations

van Sloun *et al. IUS,* 2019
van Sloun *et al. Proc. IEEE,* 2019

Deep unfolded sparse coding for ultrasound microscopy

Learned convolutional ISTA



Deep unfolded low-rank + sparse coding for clutter removal

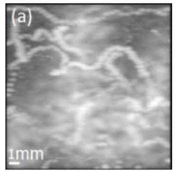
Learned convolutional ISTA for RPCA

$$\min_{\mathbf{L},\mathbf{S}} \frac{1}{2}||\mathbf{D}-\mathbf{L}-\mathbf{S}||_F^2 + \lambda_1||\mathbf{L}||_* + \lambda_2||\mathbf{S}||_{1,2}$$

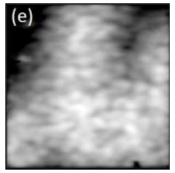
Recall sparse coding:

$$\hat{\mathbf{S}} = \min_{\mathbf{S}} \; ||\mathbf{AS}-\mathbf{D}||_2^2 + \lambda||\mathbf{S}||_1$$

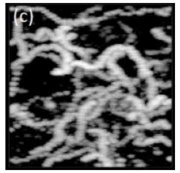
Stack of images



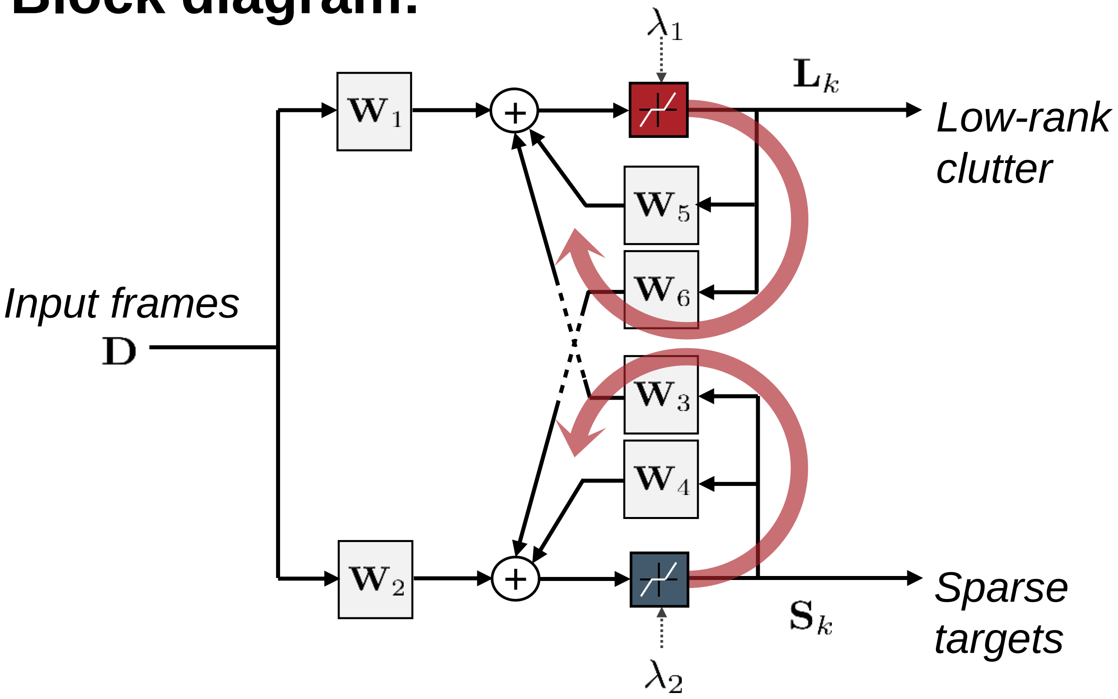
Low-rank clutter



Sparse targets



Block diagram:



Update steps:

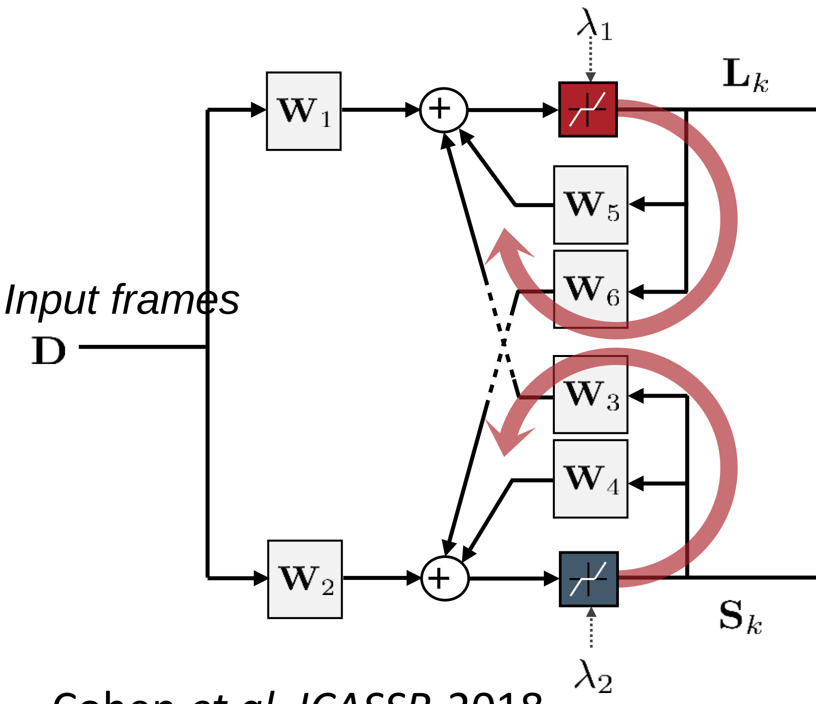
$$\mathbf{L}^{k+1} = \mathcal{SVT}_{\lambda_1/2}\left(\frac{1}{2}\mathbf{L}^k - \mathbf{S}^k + \mathbf{D}\right)$$

$$\mathbf{S}^{k+1} = \mathcal{T}_{\lambda_2/2}\left(\frac{1}{2}\mathbf{S}^k - \mathbf{L}^k + \mathbf{D}\right)$$

Deep unfolded low-rank + sparse coding for clutter removal

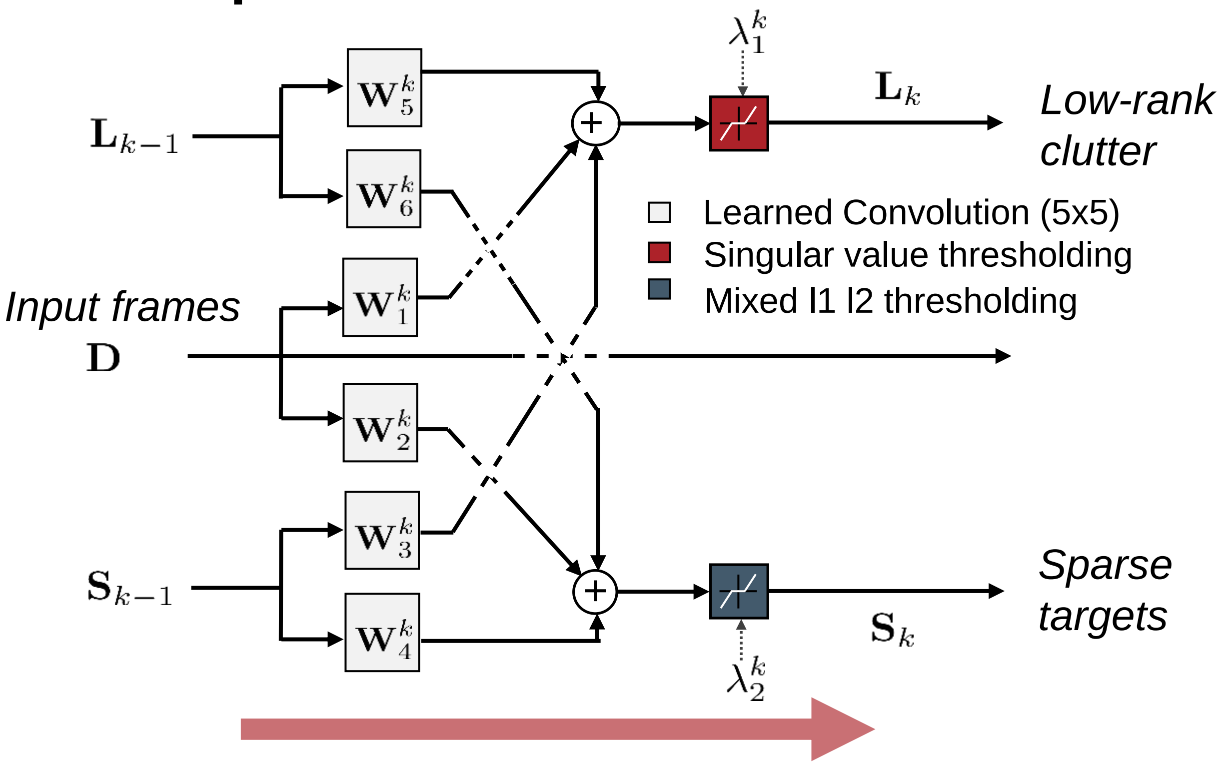
Learned convolutional ISTA for RPCA

ISTA for RPCA



Cohen *et al. ICASSP*, 2018
Cohen *et al. IEEE TMI*, 2019
van Sloun *et al. Proc. IEEE*, 2019

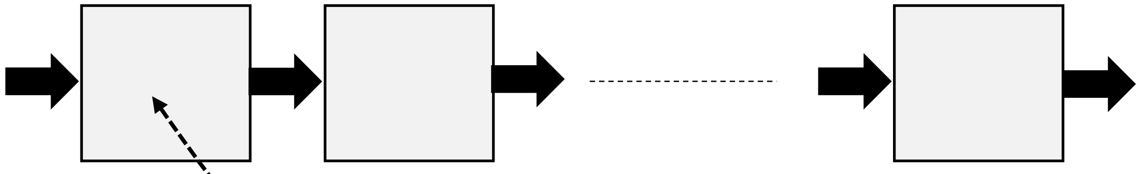
Deep convolutional Robust PCA

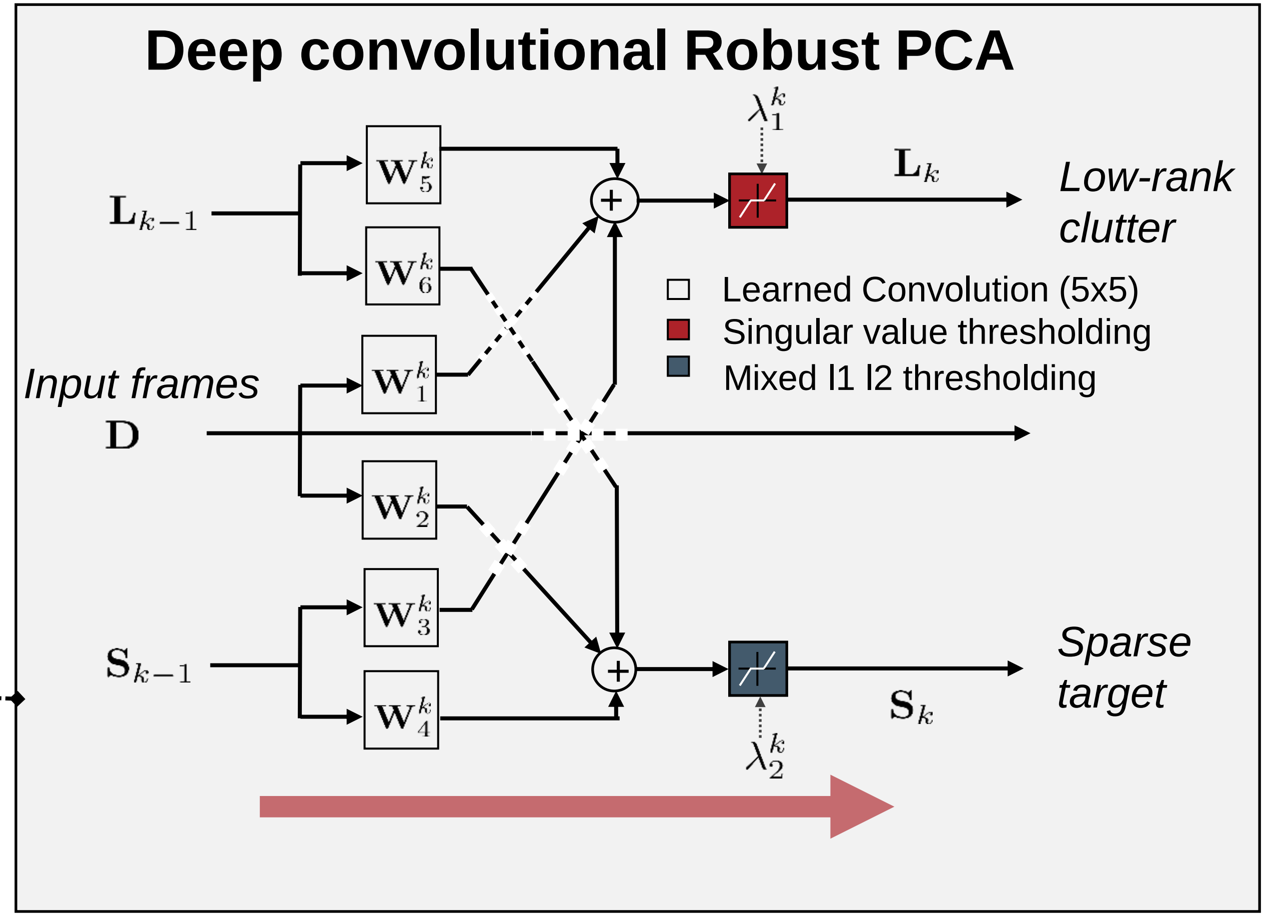


Deep unfolded low-rank + sparse coding for clutter removal

Learned convolutional ISTA for RPCA

Model-based feedforward neural network



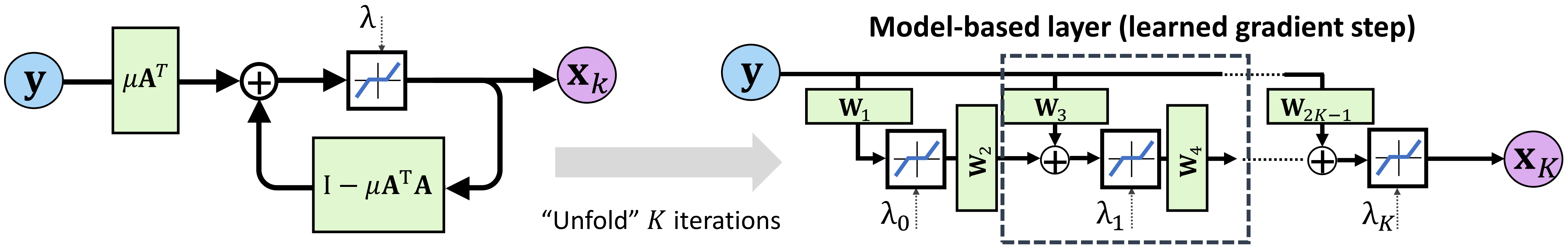


Neural proximal gradient descent

What if you know the model (*e.g.* $\mathbf{y} = \mathbf{A}\mathbf{x}$) but not the signal prior?

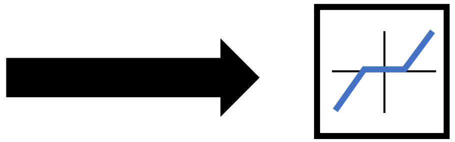
Again, many applications: denoising, compressed sensing, image reconstruction, super-resolution, …

Recall unfolded ISTA for sparse coding (known prior, sparsity):



Remember the iterative update rule:

1. Take a gradient step towards $\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 = 0$
2. Move it in the proximity of $\lambda\|\mathbf{x}\|_1 = 0$



= proximal operator/mapping

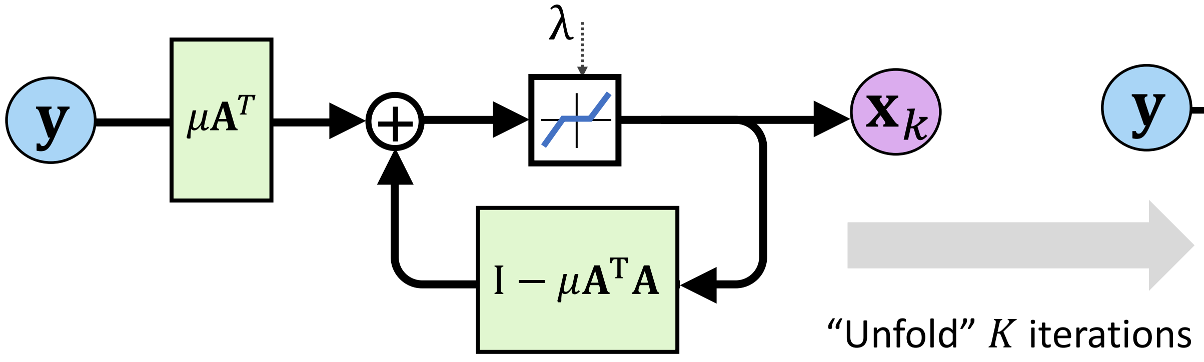
= the prior/regularizer
(here: sparsity of x)

Neural proximal gradient descent

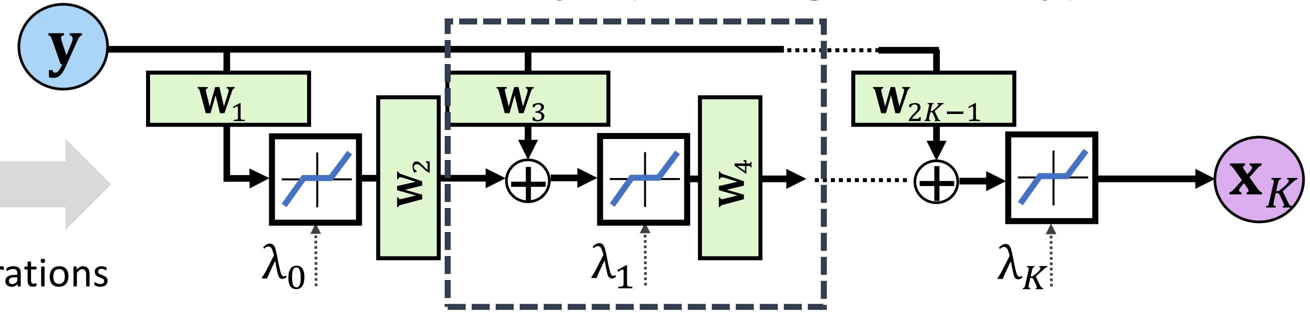
What if you know the model (*e.g.* $\mathbf{y} = \mathbf{A}\mathbf{x}$) but not the signal prior?

Again, many applications: denoising, compressed sensing, image reconstruction, super-resolution, …

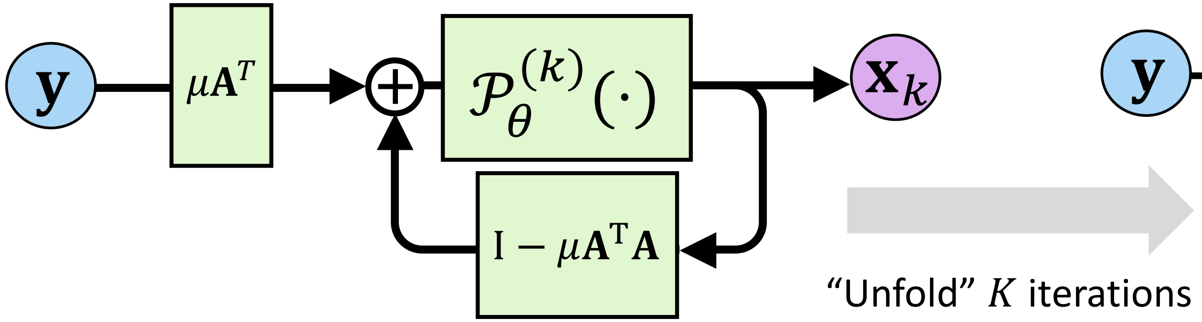
Recall unfolded ISTA for sparse coding (known prior, sparsity):



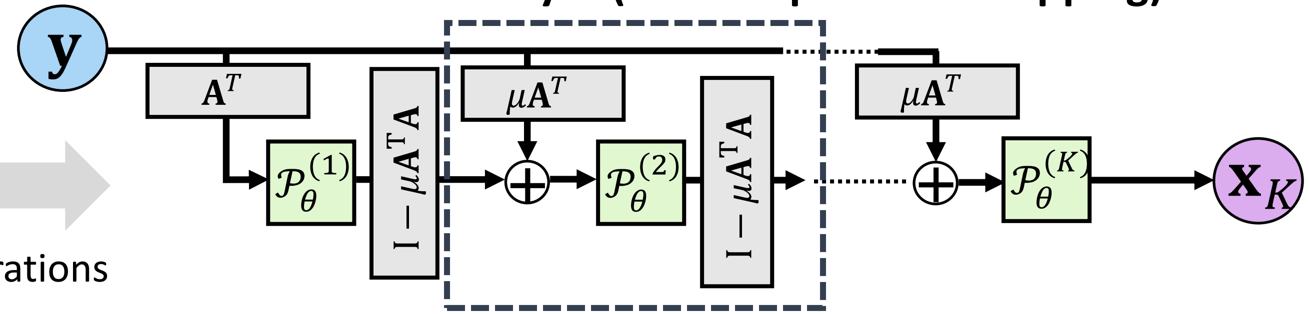
Model-based layer (learned gradient step)



Neural proximal gradient descent (known model, learned prior)



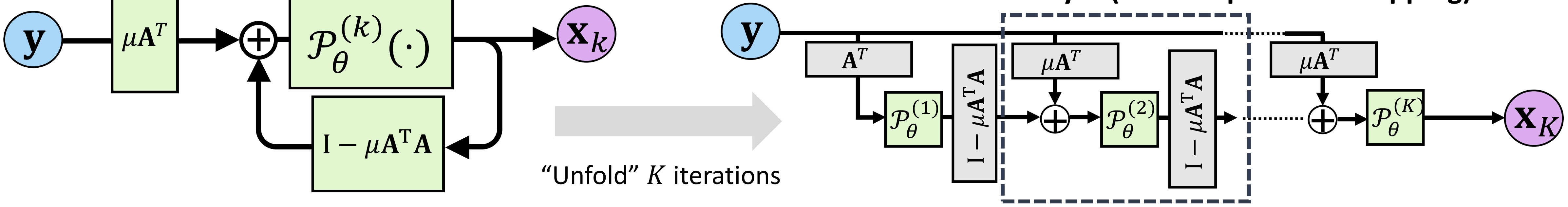
Model-based layer (learned proximal mapping)



Neural proximal gradient descent

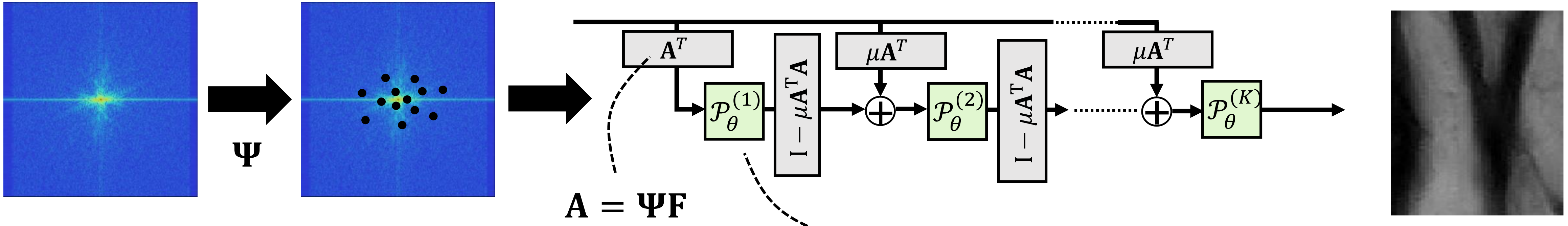
Neural proximal gradient descent (known model, learned prior)

Model-based layer (learned proximal mapping)



Example: fast MRI reconstruction from known undersampled/compressed acquisitions

K-space **Undersampled K-space** **Neural proximal gradient network** **Reconstruction**

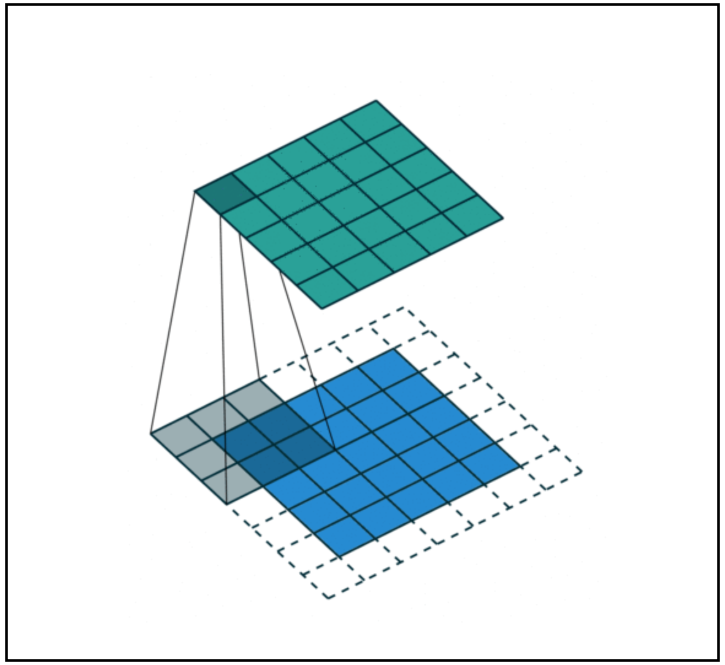


(Sampling matrix) *(partial Fourier matrix)* *Learn the MR image prior*

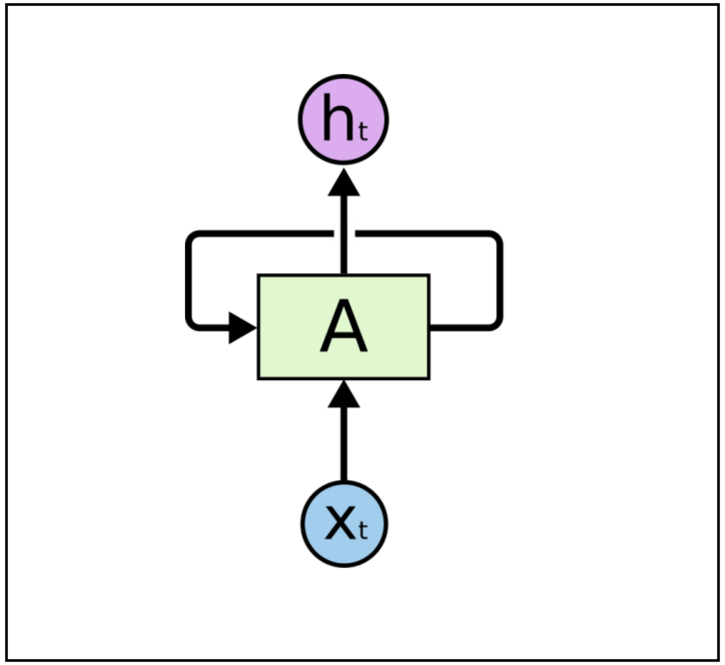
[Huijben et al., *IEEE ICASSP*, 2020]

TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY

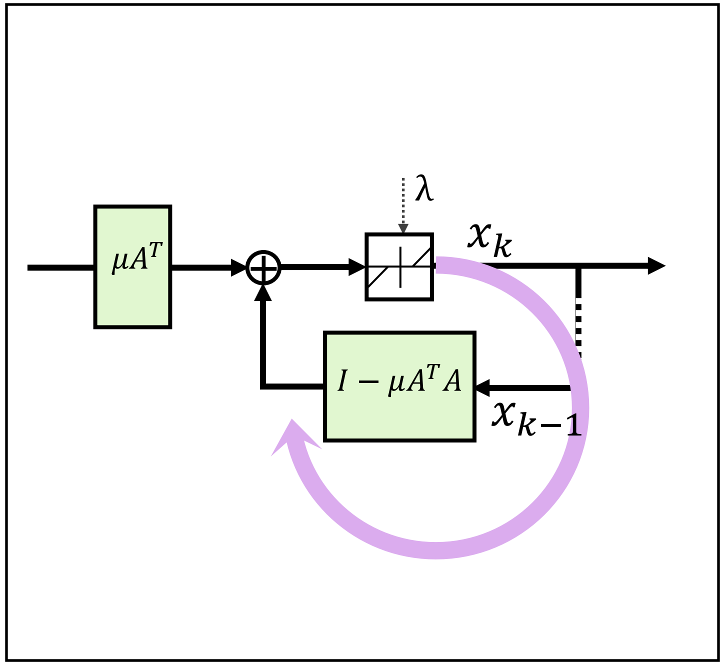
Architectural priors



(Group) convolutional neural networks



Recurrent neural networks



Deep unfolding

Lecture 5, Deep Unfolding

Change of notation:
Let $m \in M$ be a model choice
$p(\theta, m) = p(\theta)p(m)$

Do we have any prior preferences for the model choice?

$$p(\theta, m|D) = \frac{p(D|\theta, m)}{p(D)} p(\theta)p(m)$$

What are the best model parameters?
And what is the best model choice?