Model-driven Deep Learning

Jian Sun (孙剑)

Xi'an Jiaotong University

Email: jiansun@mail.xjtu.edu.cn

Home page: http://jiansun.gr.xjtu.edu.cn

April, 2019

Outline

- Introduction
  - Background: *Image analysis / deep neural networks*
  - Motivation
- Model-driven Deep Learning Approach
  - Learning Markov Random Field Model for Image Restoration
  - Deep ADMM-Net for Fast Compressive Sensing MRI
  - Deep Fusion-Net for Multi-Atlas MR Image Segmentation
- Recent Progress
  - Learning proximal operators
  - Multimodal medical image synthesis
  - Learning Graph CNNs for 3D shape analysis
  - Learning to Optimize
- Discussion & Conclusion

Backgrounds--Image Processing & Analysis

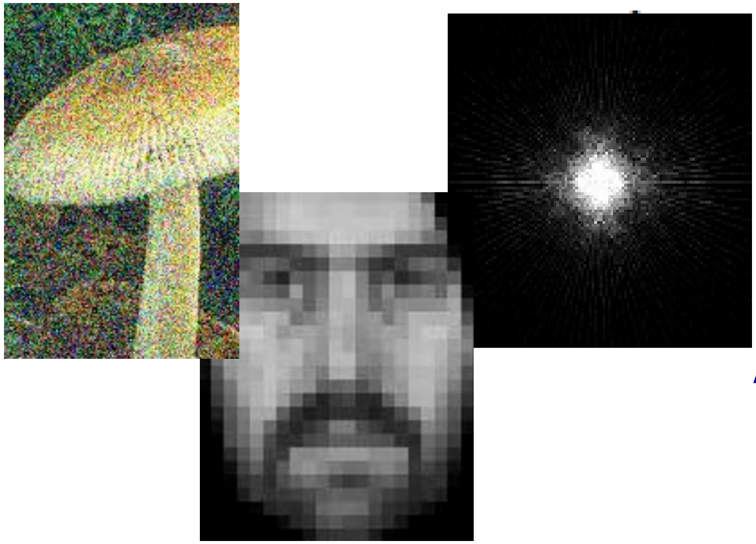
- Restoration & Reconstruction

Image Degradation: noises, motion blur, k-space sampling, etc.



$$\boxed{Y} = \boxed{A}\boxed{X} + \varepsilon$$

Physical imaging model



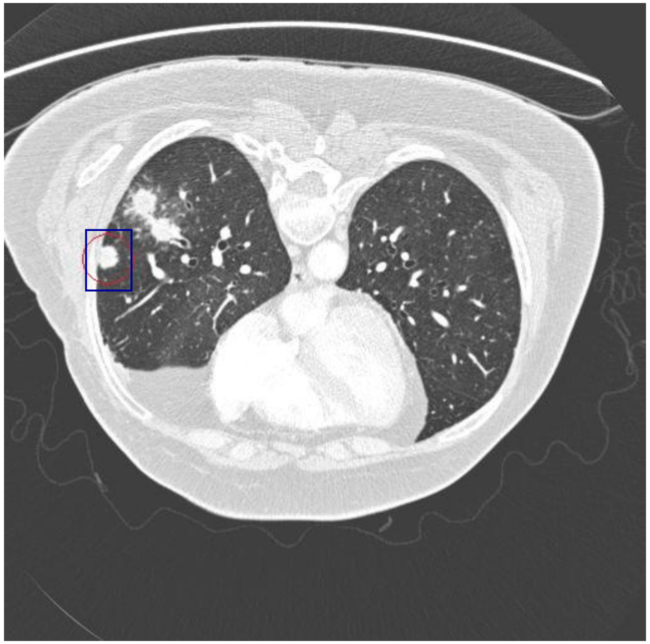
Restoration & Reconstruction

Inverse Problems

Backgrounds--Image Processing & Analysis

- Segmentation & Recognition

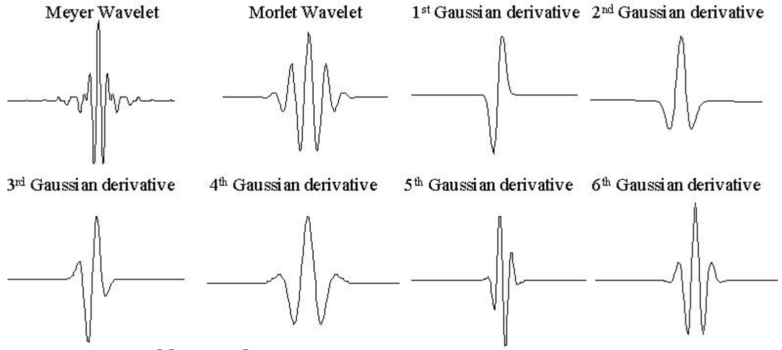
Semantic Segmentation

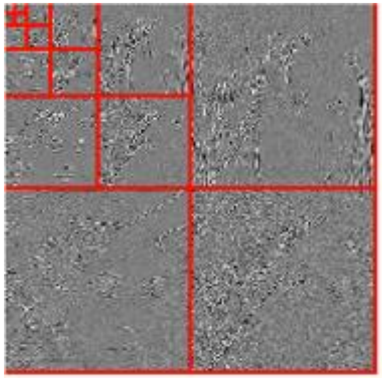


Lesion (Pulmonary nodule) localization and classification

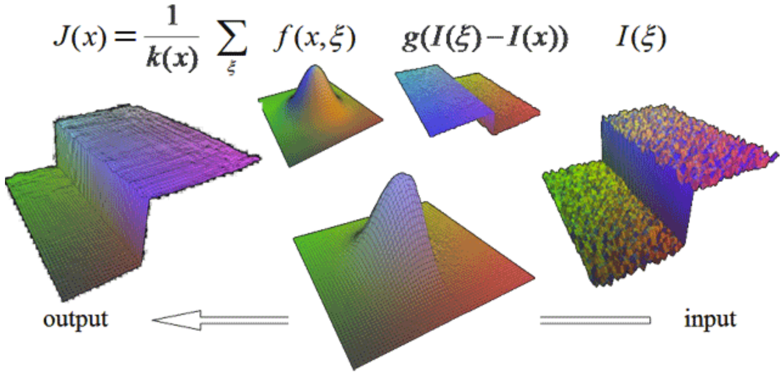
Backgrounds--Models

- Conventional Models: *Signal processing approaches*
  - *Wavelets*





  - *Image Filtering*



Backgroundsーー Models

- Conventional Models: *Energy model and its optimization*
  - *Energy Model with Regularization*

$$x^* = \arg\min_{x} D(x, y; w) + R(w)$$

  - *Dictionary Learning*

$$\{D^*, x^*\} = \mathrm{argmin}_{D,x} \, ||D\alpha - y|| + ||\alpha||_p^p$$

Applications: *Image Restoration / Segmentation / Classification / MRI / Lesion detection*

Backgrounds--Models

- Conventional Models: statistical models

  Evidence lower bound (ELBO)

$$\mathcal{L}(\boldsymbol{\theta}, \phi; \mathbf{x}^{(i)}) = -D_{KL}(q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})||p_{\boldsymbol{\theta}}(\mathbf{z})) + \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}^{(i)})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}^{(i)}|\mathbf{z})\right]$$

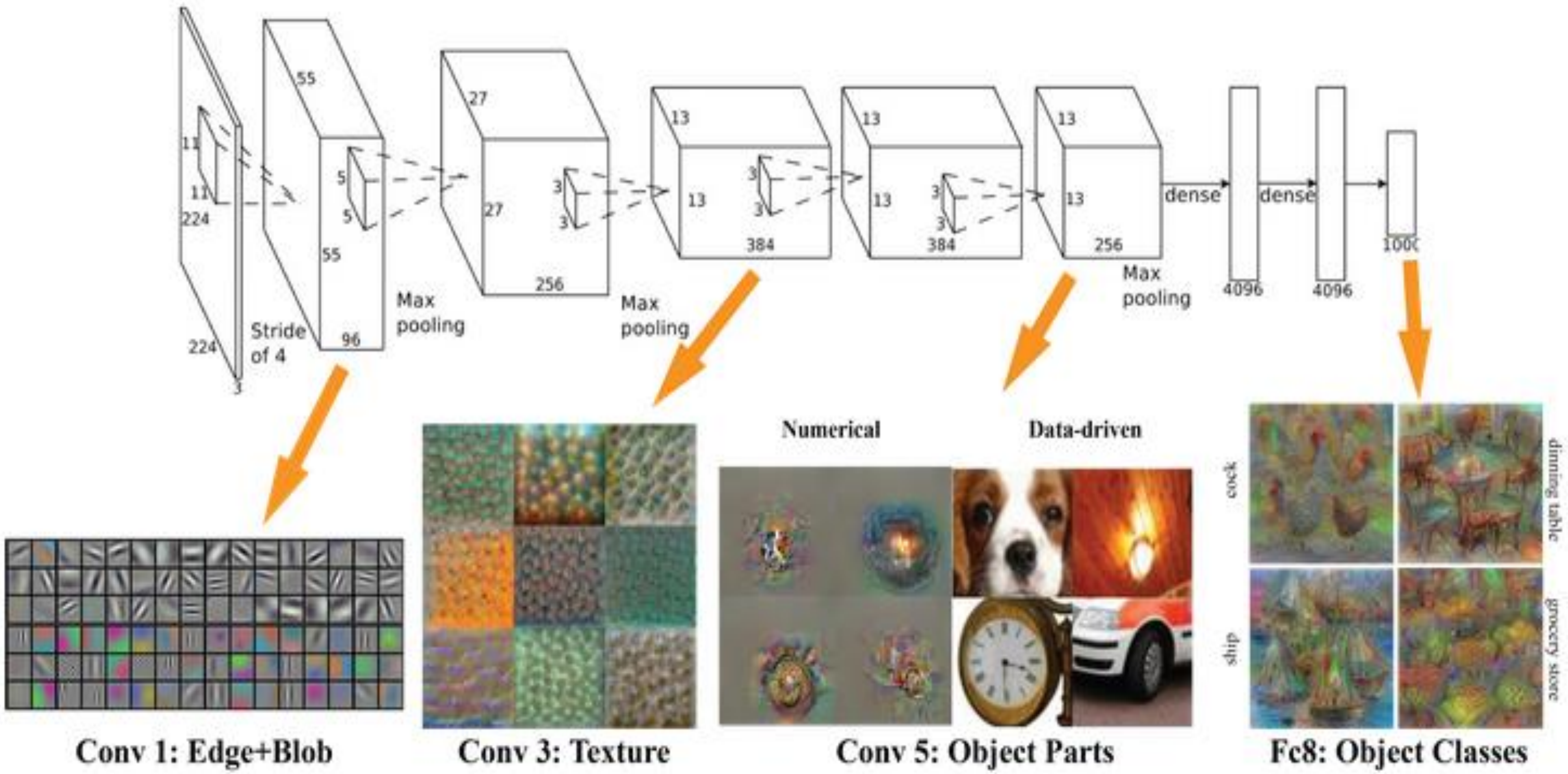
  Expectation-maximization (EM)

  Variational Inference

  Variational expectation-maximization

Backgrounds--Deep Neural Networks

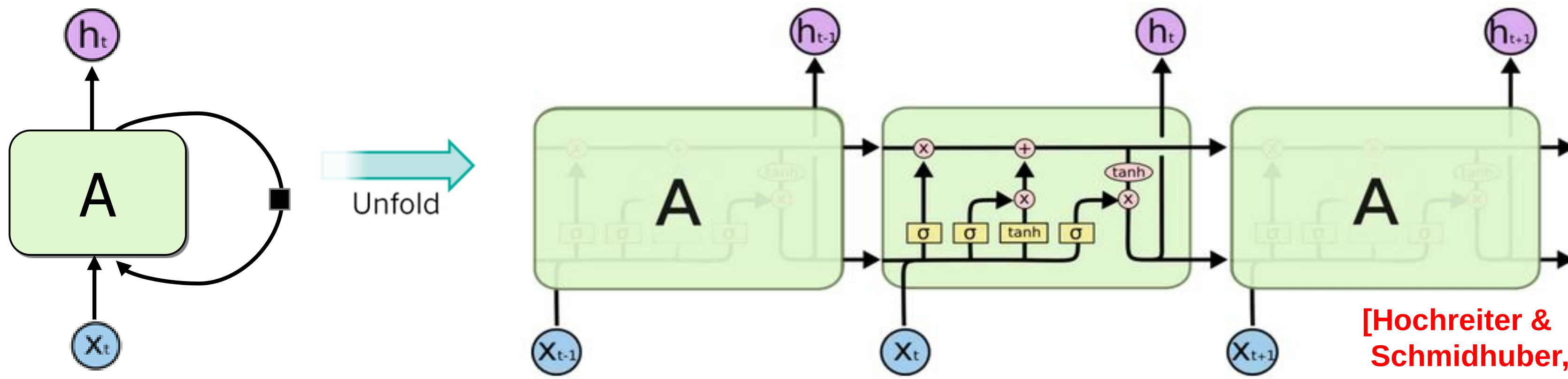
- Deep Convolutional Neural Network



CNN **[Krizhevsky A, et al., 2012]**

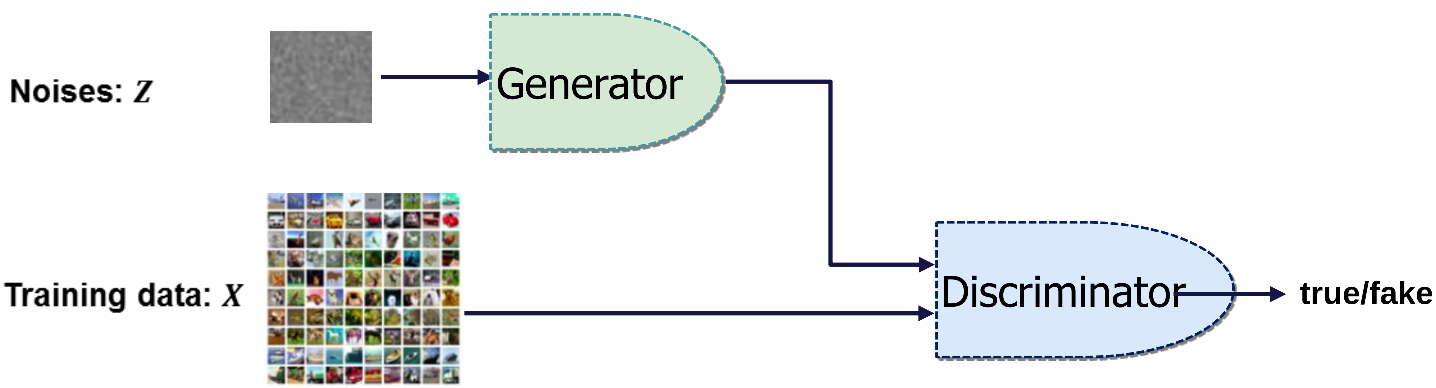
Backgrounds--Deep Neural Networks

- **LSTM:**



[Hochreiter & Schmidhuber,1997]

- **GAN**



[Ian Goodfellow et al., 2014]

Conventional Model *Vs.* Deep NNs

Conventional Models

(Optimization / statistics / energy model…)

Pros:

- Easy to incorporate domain knowledge
- Rely on less training data
- Good generalization ability

Cons:

- Maybe not optimal for specific task
- Parameter tuning

Deep Neural Networks

(CNN / LSTM / GAN….)

Pros:

- An universal regressor
- Efficiency
- Effectiveness

Cons:

- Rely on large training set
- Relatively fixed structure
- Hardly incorporate domain knowledge

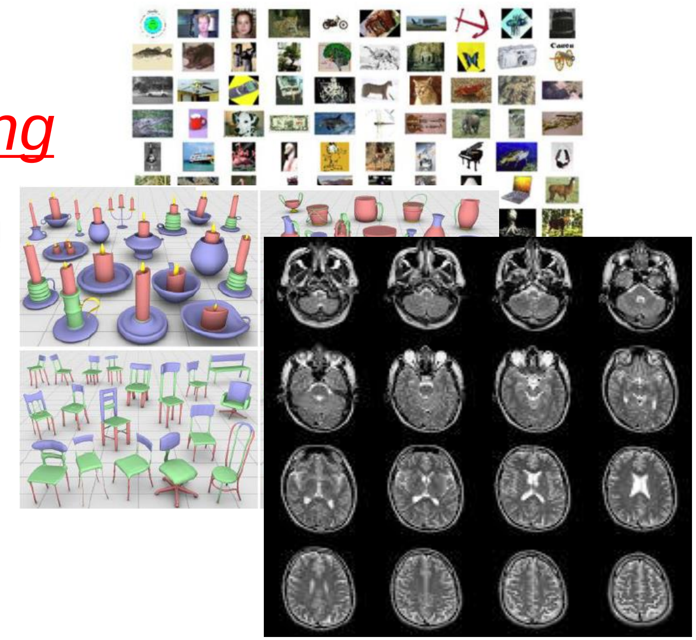
Model-driven Deep Learning

Model

- *Formulations?*
  - *Energy model*
  - *Statistical model*
  - *Image priors*
- *Parameters?*
  - *Hyperparameters*
  - *Statistical model parameters*
- *Strategies?*
  - *Gradient updates in optimization*
  - *Actions in control*

Deep learning

Task-specific training data



Why model-driven?

Explainable ML; Prior knowledge; Traditional model-based approach

Model-driven Deep Learning

- Optimization-driven DL
  - Sparse coding optimization

    [Karol Gregor, et al, ICML 2010; P. Sprechmann, et al, PAMI 2015, etc.]
  - Gradient descent, ADMM, proximal operators, etc

    [J. Sun, et al., CVPR 2011; Y. Yang, J. Sun et al., NIPS 2016; Tim. Meinhardt, et al., ICCV 2017, etc.]
- Statistical model-driven DL
  - MRF, CRF

    [S. Zheng, et al., ICCV 2015; J. Sun, et. al., IEEE TIP 2013, etc.]
  - Variational inference

    [J. Marino, et al., ICLR 2018; etc]
  - EM [D. P. Kingma, ICLR 2014; Greff, Klaus, et al., NIPS 2017, etc]

……

Outline

- Introduction
  - Background: *Image analysis / deep neural networks*
  - Motivation
- Model-driven Deep Learning Approach
  - Learning Markov Random Field Model for Image Restoration
  - Deep ADMM-Net for Fast Compressive Sensing MRI
  - Deep Fusion-Net for Multi-Atlas MR Image Segmentation
- Recent Progress
  - Learning proximal operators
  - Multimodal medical image synthesis
  - Learning Graph CNNs for 3D shape analysis
  - Learning to Optimize
- Discussion & Conclusion

Example

- Non-local Range MRF [*J. Sun, M. Tappen, CVPR 2011*]
  - A novel Markov random field model
  - Discriminative parameter learning

Example

- Non-local Range MRF [*J. Sun, M. Tappen, CVPR 2011*]
  - A novel Markov random field model
  - Discriminative parameter learning

$$p(\mathbf{x}) = \frac{1}{Z(\Theta)} \exp(-\sum_{c \in C} V_c(\mathbf{x}; \Theta))$$

Non-local Range MRF

$$x^*(\Theta) = \arg\min_{x} \left\{ E(x \mid y, \Theta) = E_{data}(y \mid x) + \boxed{E_{prior}(x; \Theta)} \right\}$$

Example

- Non-local Range MRF [*J. Sun, M. Tappen, CVPR 2011*]
  - A novel Markov random field model
  - Discriminative parameter learning

$$p(\mathbf{x}) = \frac{1}{Z(\Theta)}\exp(-\sum_{c\in C} V_c(\mathbf{x};\Theta))$$

Non-local Range MRF

$$x^*(\Theta) = \arg\min_x \left\{ E(x \mid y, \Theta) = E_{data}(y \mid x) + \boxed{E_{prior}(x;\Theta)} \right\}$$

$$\Theta^* = \arg\min_{\Theta} L(x^*(\Theta), t)$$

$$where\ x^*(\Theta) = \arg\min_x E(x \mid y, \Theta)$$

Example

- Non-local Range MRF [*J. Sun, M. Tappen, CVPR 2011*]
  - A novel Markov random field model
  - Discriminative parameter learning

$$p(\mathbf{x}) = \frac{1}{Z(\Theta)}\exp(-\sum_{c\in C} V_c(\mathbf{x};\Theta))$$

Non-local Range MRF

$$x^*(\Theta) = \arg\min_x \left\{ E(x \mid y,\Theta) = E_{data}(y \mid x) + \boxed{E_{prior}(x;\Theta)} \right\}$$

$$\Theta^* = \arg\min_\Theta L(x^*(\Theta), t)$$

$$where\ x^*(\Theta) = \arg\min_x E(x \mid y,\Theta)$$

$$\Theta^* = \mathrm{argmin}_\Theta L(\mathbf{x}^K(\Theta), \mathbf{t})$$

$$\text{where } \mathbf{x}^K(\Theta) = \mathrm{GradDesc}_K\{E(\mathbf{x}|\mathbf{y},\Theta)\}$$

Example

- Non-local Range MRF [*J. Sun, M. Tappen, CVPR 2011*]
  - A novel Markov random field model
  - Discriminative parameter learning

$$p(\mathbf{x}) = \frac{1}{Z(\Theta)} \exp\left(-\sum_{c \in C} V_c(\mathbf{x}; \Theta)\right)$$

Non-local Range MRF

$$x^*(\Theta) = \arg\min_x \left\{ E(x \mid y, \Theta) = E_{data}(y \mid x) + \boxed{E_{prior}(x; \Theta)} \right\}$$

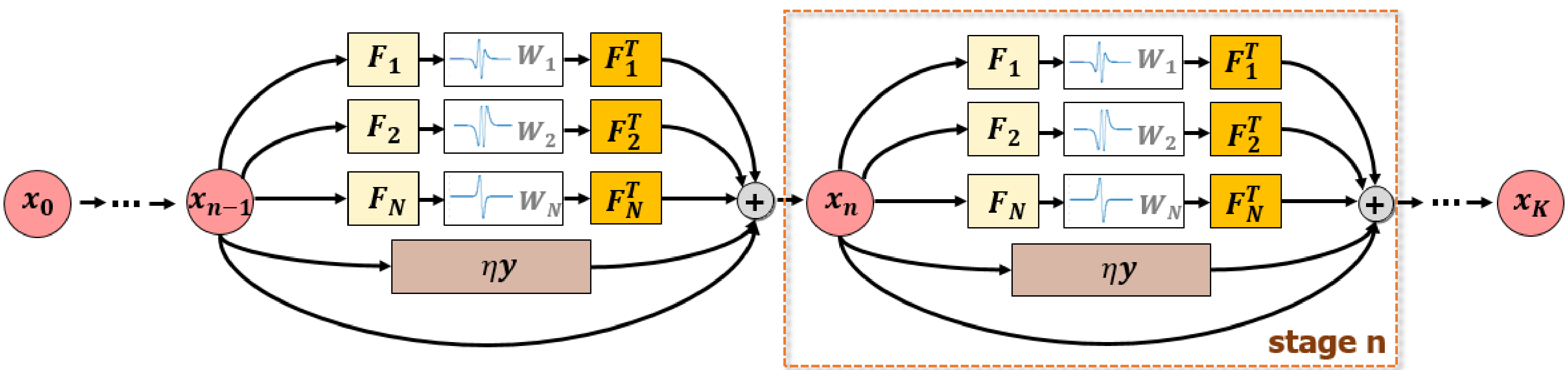
$$\Theta^* = \arg\min_\Theta L(x^*(\Theta), t)$$

$$where\ x^*(\Theta) = \arg\min_x E(x \mid y, \Theta)$$

$$\Theta^* = \operatorname{argmin}_\Theta L(\mathbf{x}^K(\Theta), \mathbf{t})$$

$$\text{where } \mathbf{x}^K(\Theta) = \text{GradDesc}_K\{E(\mathbf{x}|\mathbf{y}, \Theta)\}$$

unfolding



Non-local Range Markov Random Field Model

- Gradients of loss function w.r.t. model parameters

KEY IDEA:

$$\Theta^* = \mathrm{argmin}_{\Theta} L(\mathbf{x}^K(\Theta), \mathbf{t})$$
$$\text{where } \mathbf{x}^K(\Theta) = \mathrm{GradDesc}_K\{E(\mathbf{x}|\mathbf{y}, \Theta)\}.$$

Similar to a Neural Network with *K* layers

  – General framework to compute gradient of the parameter $\theta \in \Theta$

Back-propagation:

$$\frac{\partial L(\{\mathbf{x}_l^K, \mathbf{t}_l\})}{\partial \theta} = \sum_l \frac{\partial L(\mathbf{x}_l^K, \mathbf{t}_l)}{\partial \theta} = -\sum_l \sum_{k=1}^{K} \frac{\partial L}{\partial \mathbf{x}_l^k} \frac{\partial g(\mathbf{x}_l^{k-1})}{\partial \theta}$$

$$\frac{\partial L(\mathbf{x}^K, \mathbf{t})}{\partial \mathbf{x}^t} = \frac{\partial L}{\partial \mathbf{x}^K} \prod_{k=t}^{K-1} \frac{\partial \mathbf{x}^{k+1}}{\partial \mathbf{x}^k} \qquad \frac{\partial \mathbf{x}^{k+1}}{\partial \mathbf{x}^k} \text{ and } \frac{\partial g(\mathbf{x}^k)}{\partial \theta}$$

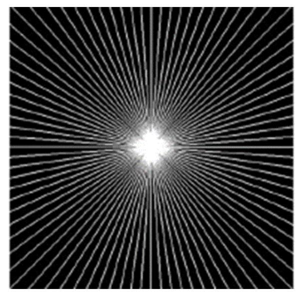
Outline

- Introduction
  - Background: *Image analysis / deep neural networks*
  - Motivation
- Model-driven Deep Learning Approach
  - Learning Markov Random Field Model for Image Restoration
  - Deep ADMM-Net for Fast Compressive Sensing MRI
  - Deep Fusion-Net for Multi-Atlas MR Image Segmentation
- Recent Progress
  - Learning proximal operators
  - Multimodal medical image synthesis
  - Learning Graph CNNs for 3D shape analysis
  - Learning to Optimize
- Discussion & Conclusion

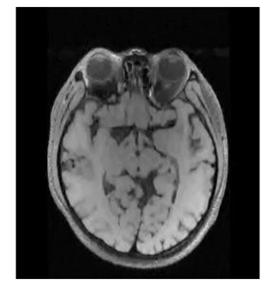
Deep ADMM-Net for Compressive Sensing

MRI Image Reconstruction

- Less sampling and fast reconstruction ?



Reconstruction



- Compressive sensing： A dominant approach in fast MRI reconstruction

[1] Michael Lustig,David L. Donoho,Compressed Sensing MRI, IEEE SIGNAL PROCESSING MAGAZINE, 2008.

Deep ADMM-Net for Compressive Sensing

A basic compressive sensing (CS) model:

$$\hat{x} = \arg\min_{x} \left\{ \frac{1}{2} \|Ax - y\|_2^2 + \sum_{l=1}^{L} \lambda_l g(D_l x) \right\}$$

A : measurement matrix,

$A = PF$ (P: Sampling matrix; F: Fourier transform)

D_l : filter matrix corresponding to convolution operation

: regularization term, e.g., l_0, l_1 norm

$/_l$: regularization term

Deep ADMM-Net for Compressive Sensing

ADMM (Alternating Direction Method of Multipliers)

Augmented Lagrangian function:

$$L_\rho(x,z,\alpha) = \frac{1}{2}\|Ax-y\|_2^2 + \sum_{l=1}^{L}\lambda_l g(z_l) - \sum_{l=1}^{L}\langle \alpha_l, z_l - D_l x\rangle + \sum_{l=1}^{L}\frac{\rho_l}{2}\|z_l - D_l x\|_2^2,$$

ADMM iterations:

$$\begin{cases} \mathbf{X}^{(\mathbf{n})}: x^{(n)} = F^T[P^T P + \sum_{l=1}^{L}\rho_l F D_l^T D_l F^T]^{-1}[P^T y + \sum_{l=1}^{L}\rho_l F D_l^T(z_l^{(n-1)} - \beta_l^{(n-1)})], \\ \mathbf{Z}^{(\mathbf{n})}: z_l^{(n)} = S(D_l x^{(n)} + \beta_l^{(n-1)}; \lambda_l/\rho_l), \\ \mathbf{M}^{(\mathbf{n})}: \beta_l^{(n)} = \beta_l^{(n-1)} + \eta_l(D_l x^{(n)} - z_l^{(n)}), \end{cases}$$

[Y Yang, J Sun, et al., NIPS 2016]

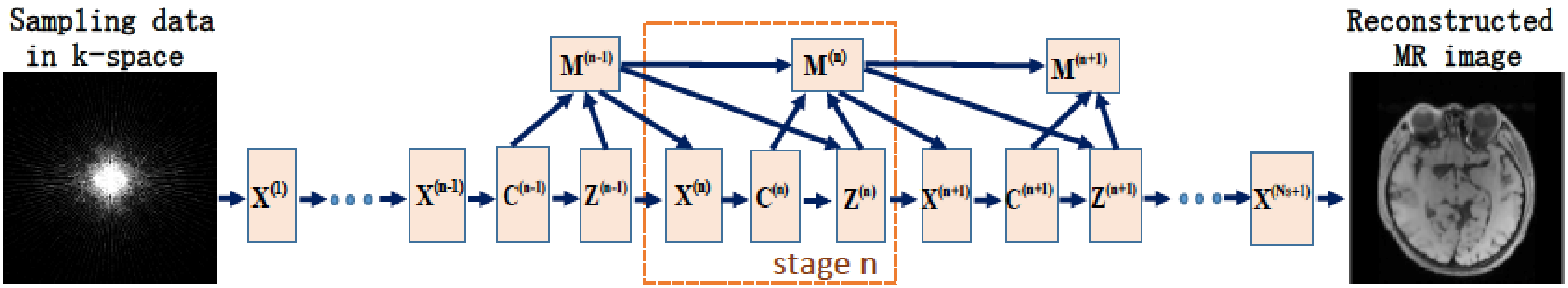
Deep ADMM-Net for Compressive Sensing

Data Flow Graph (DFG) for ADMM

$$
\begin{cases}
\mathbf{X}^{(\mathbf{n})}: x^{(n)} = F^T[P^T P + \sum_{l=1}^{L} \rho_l F D_l^T D_l F^T]^{-1}[P^T y + \sum_{l=1}^{L} \rho_l F D_l^T (z_l^{(n-1)} - \beta_l^{(n-1)})], \\
\mathbf{Z}^{(\mathbf{n})}: z_l^{(n)} = S(\boxed{D_l x^{(n)}} + \beta_l^{(n-1)}; \lambda_l/\rho_l), \\
\mathbf{M}^{(\mathbf{n})}: \beta_l^{(n)} = \beta_l^{(n-1)} + \eta_l(\boxed{D_l x^{(n)}} - z_l^{(n)}),
\end{cases}
$$

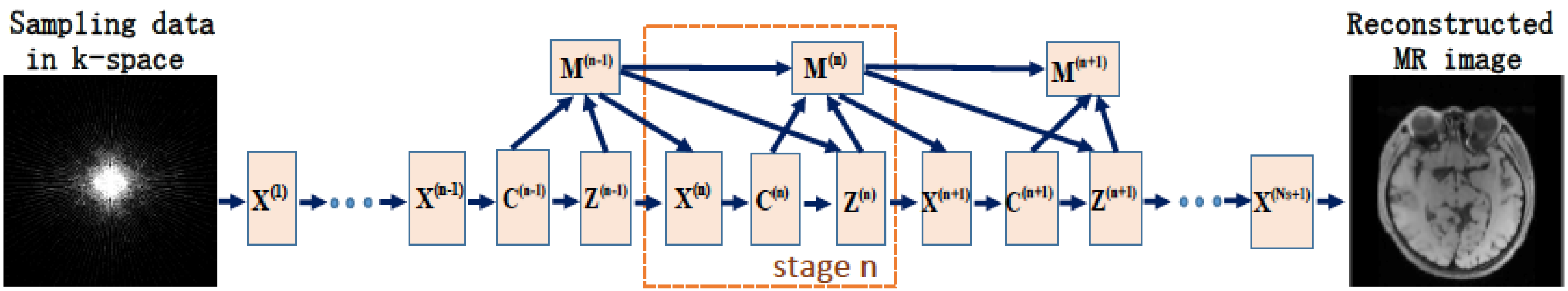
$$\boxed{C^{(n)} = D_l x^{(n)}}$$

Unfolding to stage n in DFG



Deep ADMM-Net for Compressive Sensing

- Deep ADMM-Net:



Reconstruction layer ($X^{(n)}$):

$$x^{(n)} = F^T(P^T P + \sum_{l=1}^{L} \rho_l^{(n)} F H_l^{(n)T} H_l^{(n)} F^T)^{-1}[P^T y + \sum_{l=1}^{L} \rho_l^{(n)} F H_l^{(n)T}(z_l^{(n-1)} - \beta_l^{(n-1)})],$$

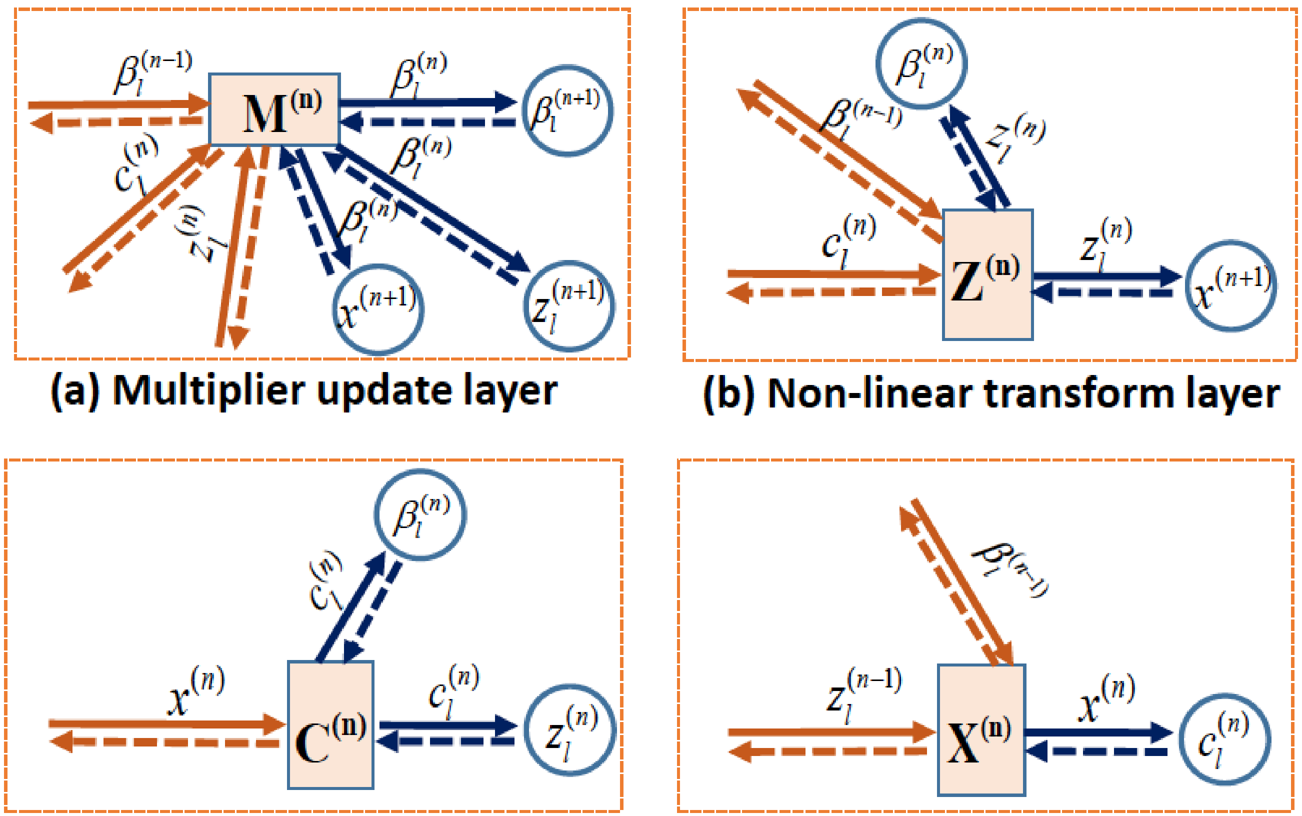
Convolution layer ($C^{(n)}$): $c_l^{(n)} = D_l^{(n)} x^{(n)}$

Nonlinear transform layer ($Z^{(n)}$): $z_l^{(n)} = S_{PLF}(c_l^{(n)} + \beta_l^{(n-1)}; \{p_i, q_{l,i}^{(n)}\}_{i=1}^{N_c}),$

Multiplier updating layer ($M^{(n)}$): $\beta_l^{(n)} = \beta_l^{(n-1)} + \eta_l^{(n)}(c_l^{(n)} - z_l^{(n)}),$

Deep ADMM-Net for Compressive Sensing

- **Network training:** Gradient computation by backpropagation



(a) Multiplier update layer

(b) Non-linear transform layer

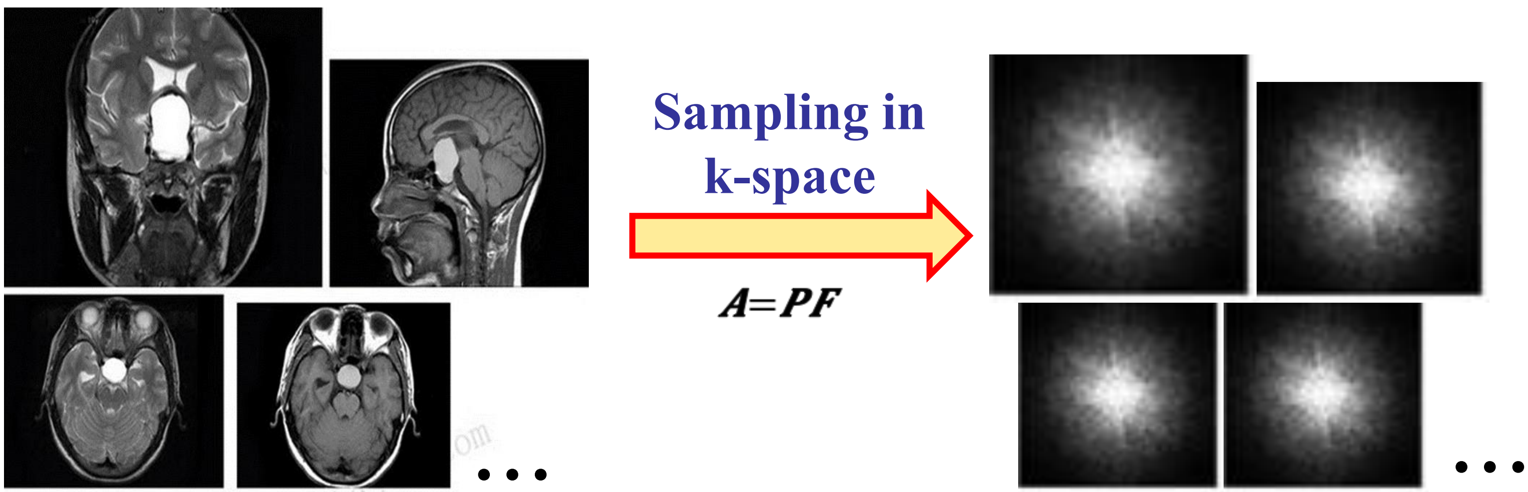
(c) Convolution layer

(d) Reconstruction layer

Parameter optimization: L-BFGS

Deep ADMM-Net for Compressive Sensing

- Training Data Generation



ground truth

Observed data

- Training loss

$$L(\theta) = \sum_{i=1}^{m} \frac{\sqrt{\|\hat{x}_i - x_i^{gt}\|_2^2}}{\sqrt{\|x_i^{gt}\|_2^2}}$$

Deep ADMM-Net for Compressive Sensing

Method	20%		30%		40%		50%		Test time
	NMSE	PSNR	NMSE	PSNR	NMSE	PSNR	NMSE	PSNR	
Zero-filling	0.1700	29.96	0.1247	32.59	0.0968	34.76	0.0770	36.73	0.0013s
TV [2]	0.0929	35.20	0.0673	37.99	0.0534	40.00	0.0440	41.69	0.7391s
RecPF [4]	0.0917	35.32	0.0668	38.06	0.0533	40.03	0.0440	41.71	0.3105s
SIDWT	0.0885	35.66	0.0620	38.72	0.0484	40.88	0.0393	42.67	7.8637s
PBDW [6]	0.0814	36.34	0.0627	38.64	0.0518	40.31	0.0437	41.81	35.3637s
PANO [10]	0.0800	36.52	0.0592	39.13	0.0477	41.01	0.0390	42.76	53.4776s
FDLCP [8]	0.0759	36.95	0.0592	39.13	0.0500	40.62	0.0428	42.00	52.2220s
BM3D-MRI [11]	0.0674	37.98	0.0515	40.33	0.0426	41.99	0.0359	43.47	40.9114s
Init-Net$_{13}$	0.1394	31.58	0.1225	32.71	0.1128	33.44	0.1066	33.95	0.6914s
ADMM-Net$_{13}$	0.0752	37.01	0.0553	39.70	0.0456	41.37	0.0395	42.62	0.6964s
ADMM-Net$_{14}$	0.0742	37.13	0.0548	39.78	0.0448	41.54	0.0380	42.99	0.7400s
ADMM-Net$_{15}$	0.0739	37.17	0.0544	39.84	0.0447	41.56	0.0379	43.00	0.7911s

Table 2: Comparisons of NMSE and PSNR on chest data with 20% sampling ratio.

Method	TV	RecPF	PANO	FDLCP	ADMM-Net$_{15}$-B	ADMM-Net$_{15}$	ADMM-Net$_{17}$
NMSE	0.1019	0.1017	0.0858	0.0775	0.0790	0.0775	**0.0767**
PSNR	35.49	35.51	37.01	37.77	37.68	37.84	**37.93**

Deep ADMM-Net for Compressive Sensing

- Extensions of ADMM-Net ([IEEE PAMI, 2018])
  - More flexible network structure

$$\min_{x}\left\{\frac{1}{2}\|Ax-y\|_2^2+\sum_{l=1}^{L}\lambda_l g(D_l x)\right\} \xrightarrow{z=[z_1,z_2,\dots,z_l]} \min_{x,z}\left\{\frac{1}{2}\|Ax-y\|_2^2+\sum_{l=1}^{L}\lambda_l g(D_l z)\right\}$$

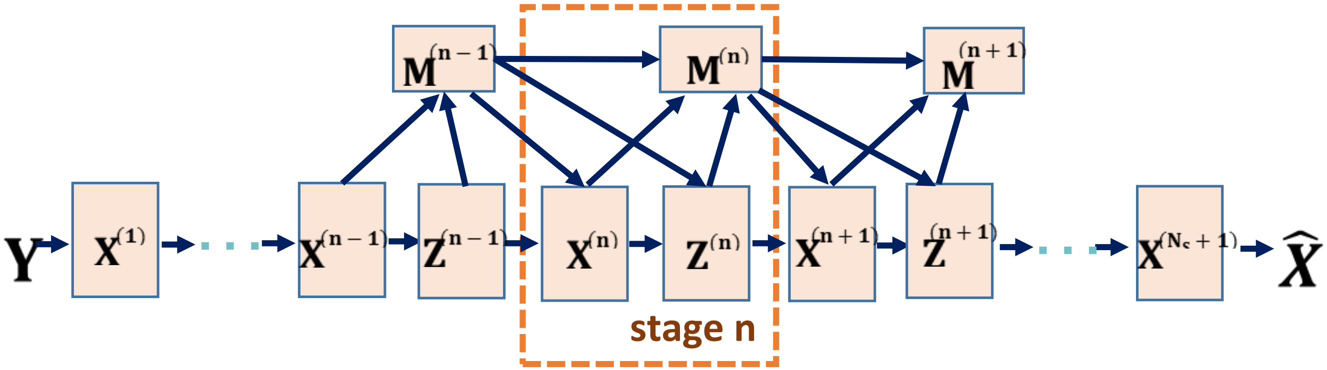
$$s.t.\ z=x$$

$$\boxed{L_\rho(x,z,\beta)=\frac{1}{2}\|Ax-y\|_2^2+\sum_{l=1}^{L}\lambda_l g(D_l z)-\langle\alpha,z-x\rangle+\frac{\rho}{2}\|z-x\|_2^2}$$

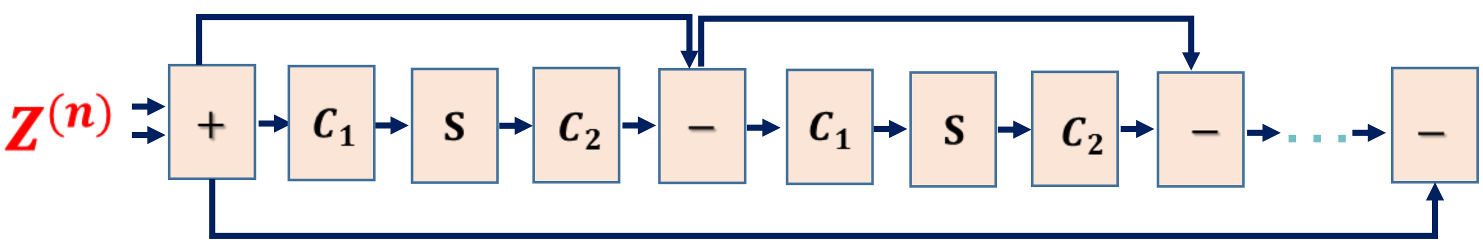
$$\begin{cases} x^{(n)}=\arg\min_{x} L_\rho\left(x,z^{(n-1)},\beta^{(n-1)}\right)=F^T\left(P^TP+\rho I\right)^{-1}\left[P^Ty+\rho F\left(z^{(n-1)}-\beta^{(n-1)}\right)\right] \\ z^{(n)}=\arg\min_{z} L_\rho\left(x^{(n)},z,\beta^{(n-1)}\right)=\arg\min_{z}\frac{\rho}{2}\left\|\left(x^{(n)}+\beta^{(n-1)}\right)-z\right\|_2^2+\sum_{l=1}^{L}\lambda_l g(D_l z) \\ \beta^{(n)}=\beta^{(n-1)}+\eta(x-z) \end{cases}$$

Deep ADMM-Net for Compressive Sensing

ADMM-Net-v2



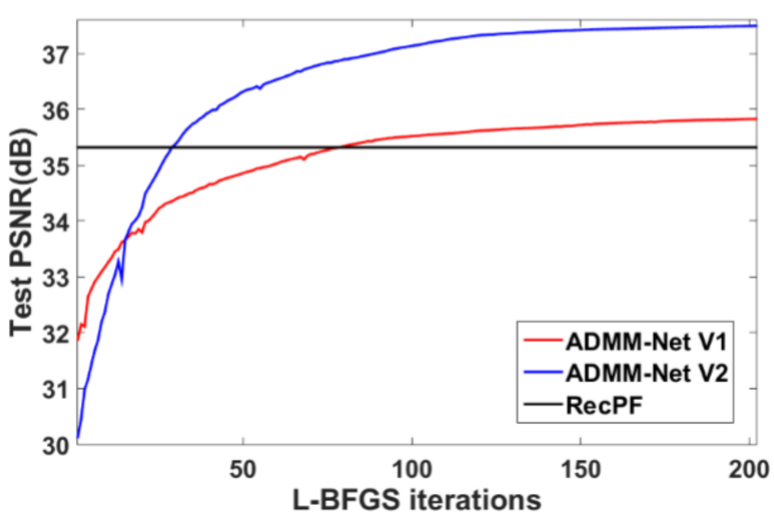
$$\arg\min_{z} \frac{\rho}{2} \left\| \left(x^{(n)} + \beta^{(n-1)}\right) - z \right\|_2^2 + \sum_{l=1}^{L} \lambda_l g(D_l z)$$



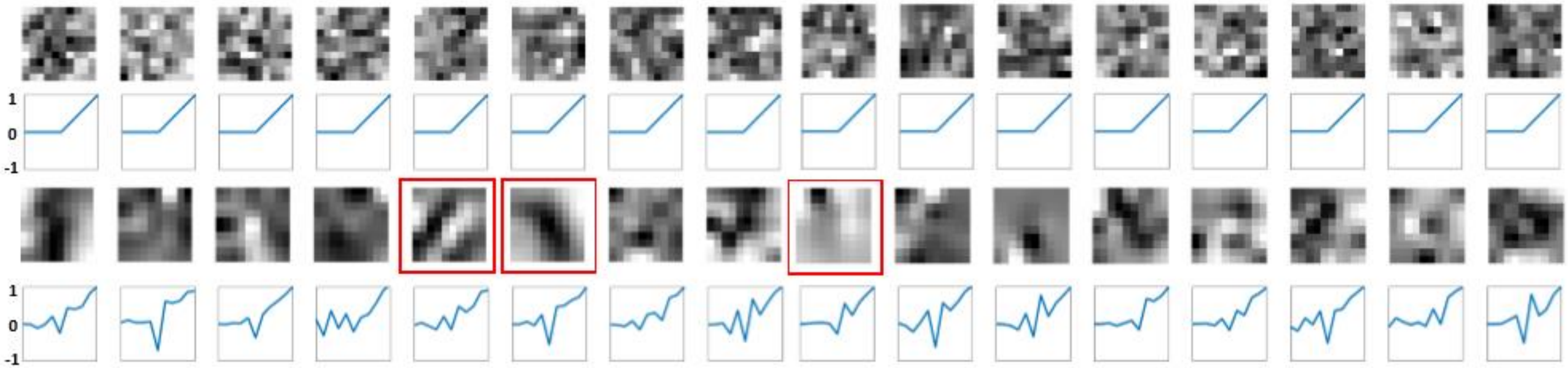
Deep ADMM-Net for Compressive Sensing

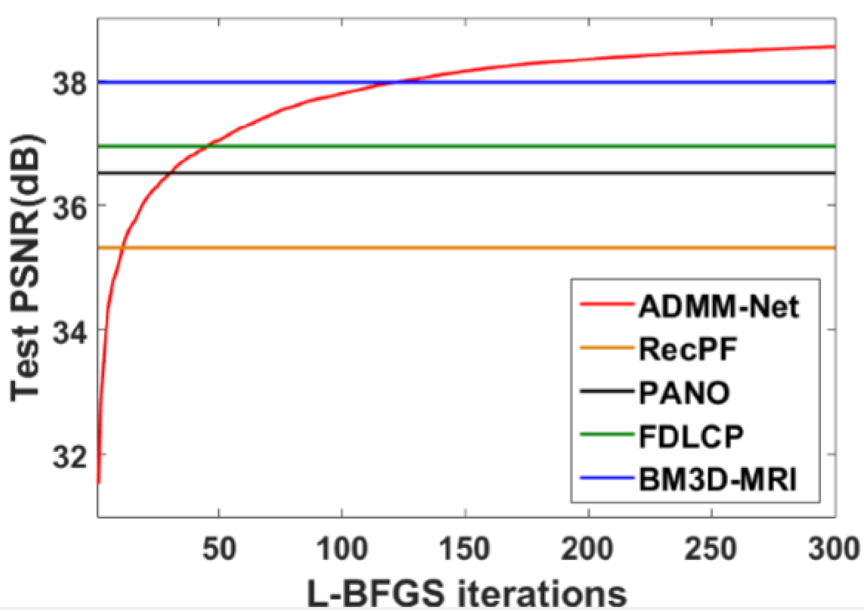
TABLE 6
Performance comparisons on brain data with different sampling ratios.

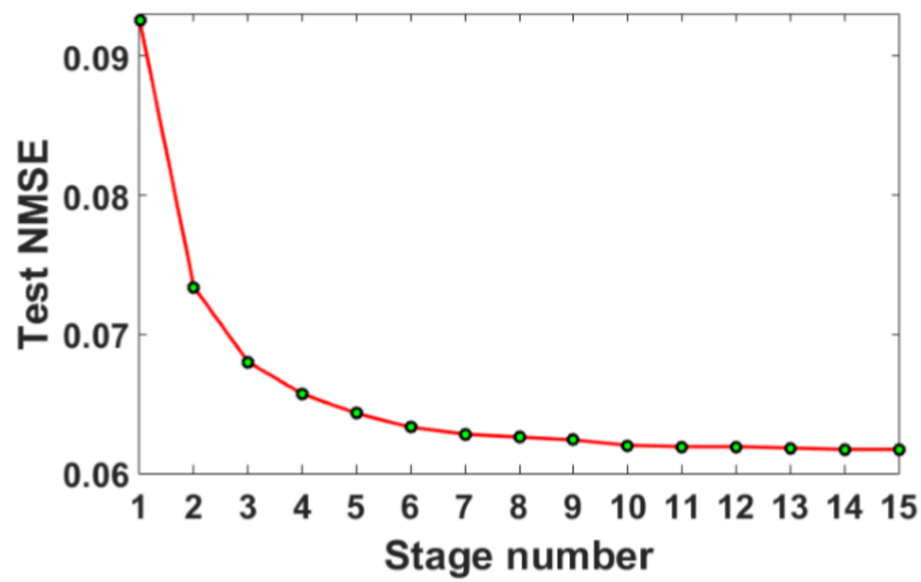
Method	20%		30%		40%		50%		Test time
	NMSE	PSNR	NMSE	PSNR	NMSE	PSNR	NMSE	PSNR	CPU \ GPU
Zero-filling	0.1700	29.96	0.1247	32.59	0.0968	34.76	0.0770	36.73	0.001s\--
TV [3]	0.0929	35.20	0.0673	37.99	0.0534	40.00	0.0440	41.69	0.739s\--
RecPF [4]	0.0917	35.32	0.0668	38.06	0.0533	40.03	0.0440	41.71	0.311s\--
SIDWT	0.0885	35.66	0.0620	38.72	0.0484	40.88	0.0393	42.67	7.864s\--
PBDW [5]	0.0814	36.34	0.0627	38.64	0.0518	40.31	0.0437	41.81	35.364s\--
PANO [6]	0.0800	36.52	0.0592	39.13	0.0477	41.01	0.0390	42.76	53.478s\--
FDLCP [7]	0.0759	36.95	0.0592	39.13	0.0500	40.62	0.0428	42.00	52.222s\--
BM3D-MRI [8]	0.0674	37.98	0.0515	40.33	0.0426	41.99	0.0359	43.47	40.911s\--
Init-Net$_{10}$	0.1737	29.64	0.1299	32.16	0.1025	34.21	0.0833	36.01	3.827s\0.652s
ADMM-Net$_{10}$	**0.0620**	**38.72**	**0.0480**	**40.95**	**0.0395**	**42.66**	**0.0328**	**44.29**	3.827s\0.652s



Deep ADMM-Net for Compressive Sensing



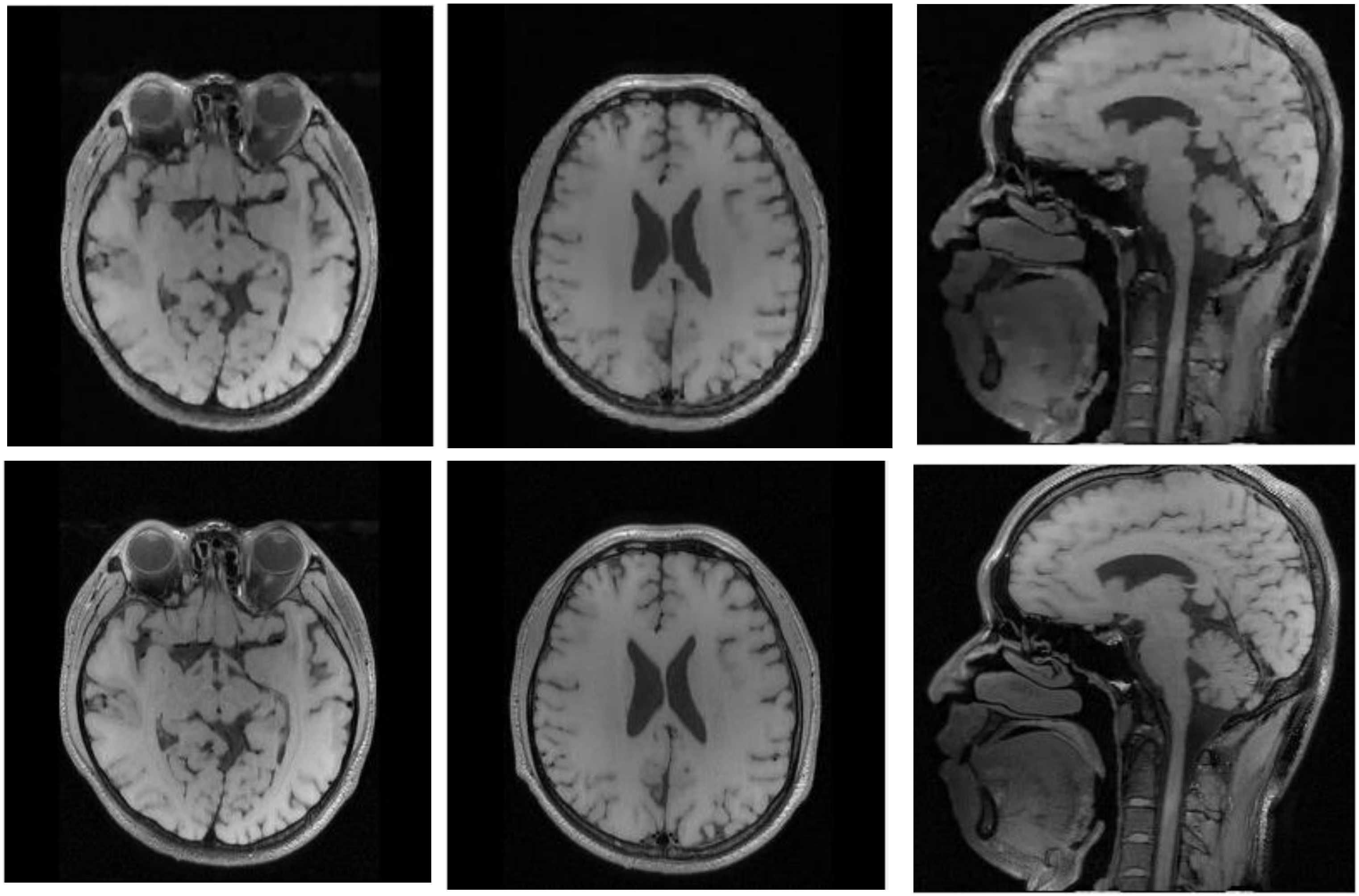




Deep ADMM-Net for Compressive Sensing

Our results：

ground truth：



Applications to more general compressive imaging:

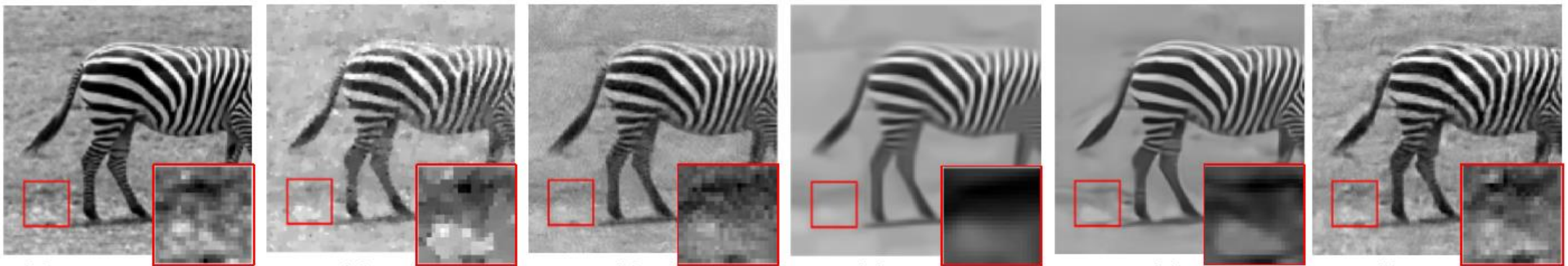
$$\mathbf{X}^{(n)} : x^{(n)} = \underbrace{(\Phi^H\Phi + \rho I)^{-1}}_{\text{Bottleneck}}[\Phi^H y + \rho(z^{(n-1)} - \beta^{(n-1)})],$$

Fast inversion:

Proposition 2. *Suppose we are given an $M \times N$ matrix Φ, a vector $b \in \mathbb{C}^N$ and constants C_0, C_1. For $x \in \mathbb{C}^N$, (1) if a condition $F\Phi^H\Phi F^H = \hat{\Phi}$ holds, where $\hat{\Phi}$ is a diagonal matrix and F is a Fourier transform matrix, then the linear system $(C_0 I_{N\times N} + C_1\Phi^H\Phi)x = b$ can be solved efficiently using FFTs by a closed-form solution $F^H(C_0 I_{N\times N} + C_1\hat{\Phi})^{-1}Fb$; (2) if a condition $\Phi\Phi^H = CI_{M\times M}$ holds, where C is a constant, then this linear system can be solved efficiently using a closed-form solution $(I_{N\times N} - \frac{C_1}{C_0+CC_1}\Phi^H\Phi)\frac{b}{C_0}$, where C is a constant.*

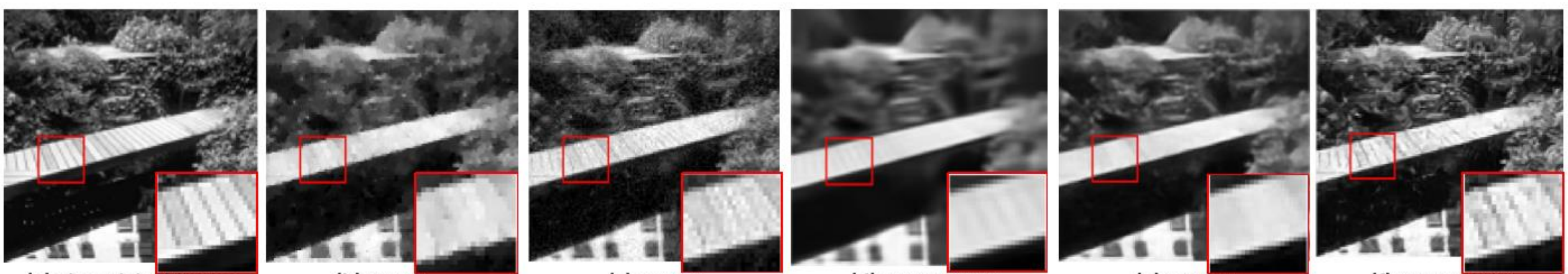
- Partial Fourier matrix
- Random matrix with orthogonal rows
- Structurally random matrix

Deep ADMM-Net for Compressive Sensing



(a) The original image (b) TVAL3 (c) NLR-CS (d) BM3D-AMP (e) LDAMP (f) ADMM-Net

Fig. 19. Reconstruction of 30% sampled Zebra image with Walsh-Hadamard measurements. (a) The ground truth image; (b)-(f) Reconstructed images based on TVAL3, NLR-CS, BM3D-AMP, LDAMP and ADMM-Net. The PSNRs (dB) are 17.30, 23.54, 19.46, 22.75 and 23.79, respectively.



(a) The original image (b) TVAL3 (c) NLR-CS (d) BM3D-AMP (e) LDAMP (f) ADMM-Net

Fig. 20. Examples of reconstruction results from the 50 testing images with 20% sampling rate with code diffraction measurements (a) The ground truth image; (b)-(f) Reconstructed images based on TVAL3, NLR-CS, BM3D-AMP, LDAMP and ADMM-Net. The PSNRs (dB) are 22.21, 23.96, 21.26, 22.90 and 25.12, respectively.

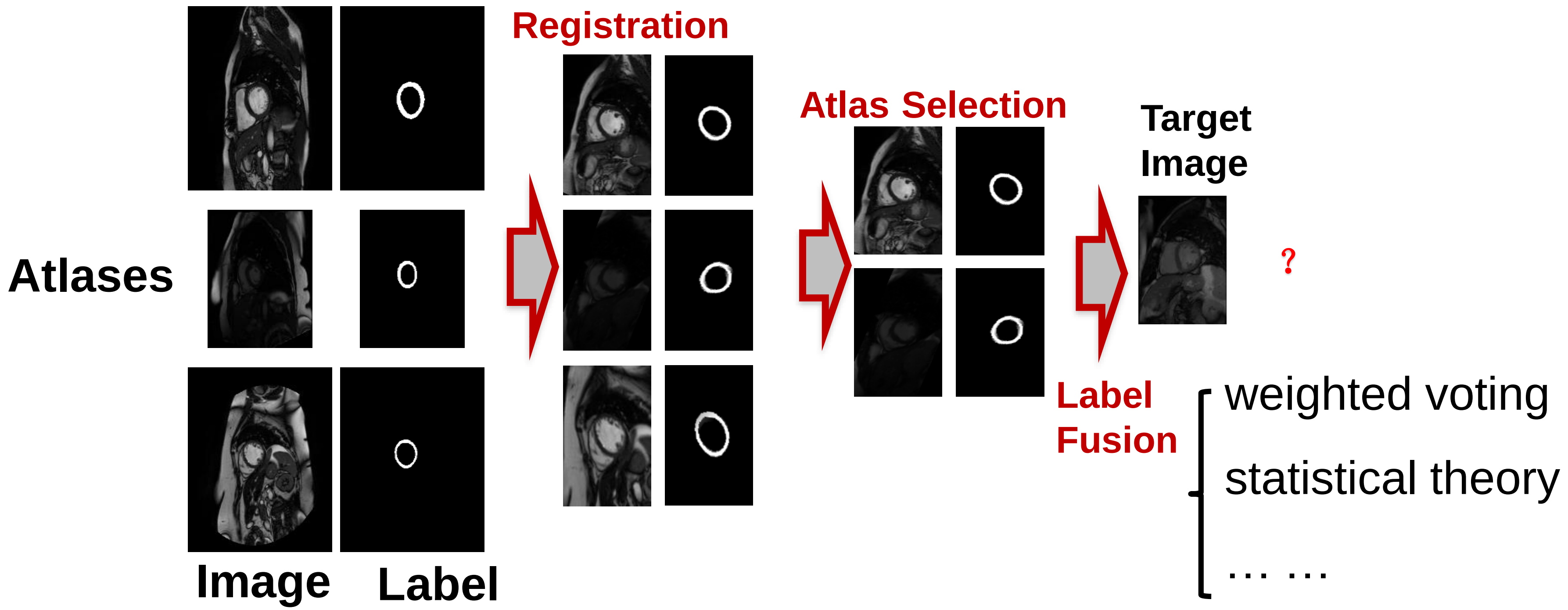
Natural image compressive sensing

Outline

- Introduction
  - Background: *Image analysis / deep neural networks*
  - Motivation
- Model-driven Deep Learning Approach
  - Learning Markov Random Field Model for Image Restoration
  - Deep ADMM-Net for Fast Compressive Sensing MRI
  - Deep Fusion-Net for Multi-Atlas MR Image Segmentation
- Recent Progress
  - Learning proximal operators
  - Multimodal medical image synthesis
  - Learning Graph CNNs for 3D shape analysis
  - Learning to Optimize
- Discussion & Conclusion

Deep Fusion Net for MR Image Segmentation

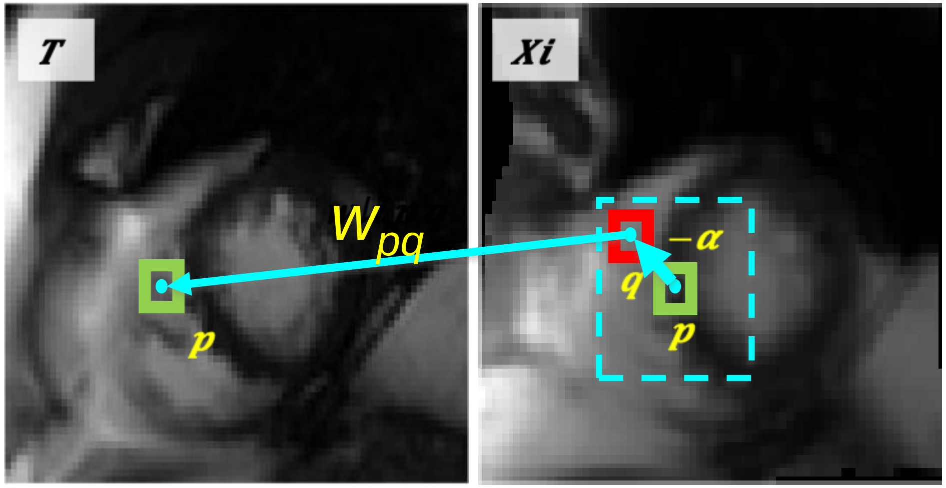
- ***Background*: Multi-atlas segmentation** has been one of the most widely-used and successful medical image segmentation techniques in the past decade.



Iglesias, J.E., et. al: Multi-atlas segmentation of biomedical images: a survey. (Med. Image Anal. 2015)

Deep Fusion Net for MR Image Segmentation

- **Non-local patch-based label fusion (NL-PLF) model**



Label fusion:

$$\hat{L}_p(T;\Theta) = \sum_i \sum_{q \in N_p} w_{i,p,q}(\Theta) L_q(X_i),$$

Fusion weight:

$$w_{i,p,q}(\Theta) = \frac{\exp(-||F_p(T;\Theta) - F_q(X_i;\Theta)||^2)}{\sum_j \sum_{q \in N_p} \exp(-||F_p(T;\Theta) - F_q(X_j;\Theta)||^2)},$$

Hand-crafted features

1. Intensity (Coupe et al., 2011)
2. Intensity + spatial context (Wang et al., 2014)
3. Intensity + gradient + contextual (Bai et al., 2015)

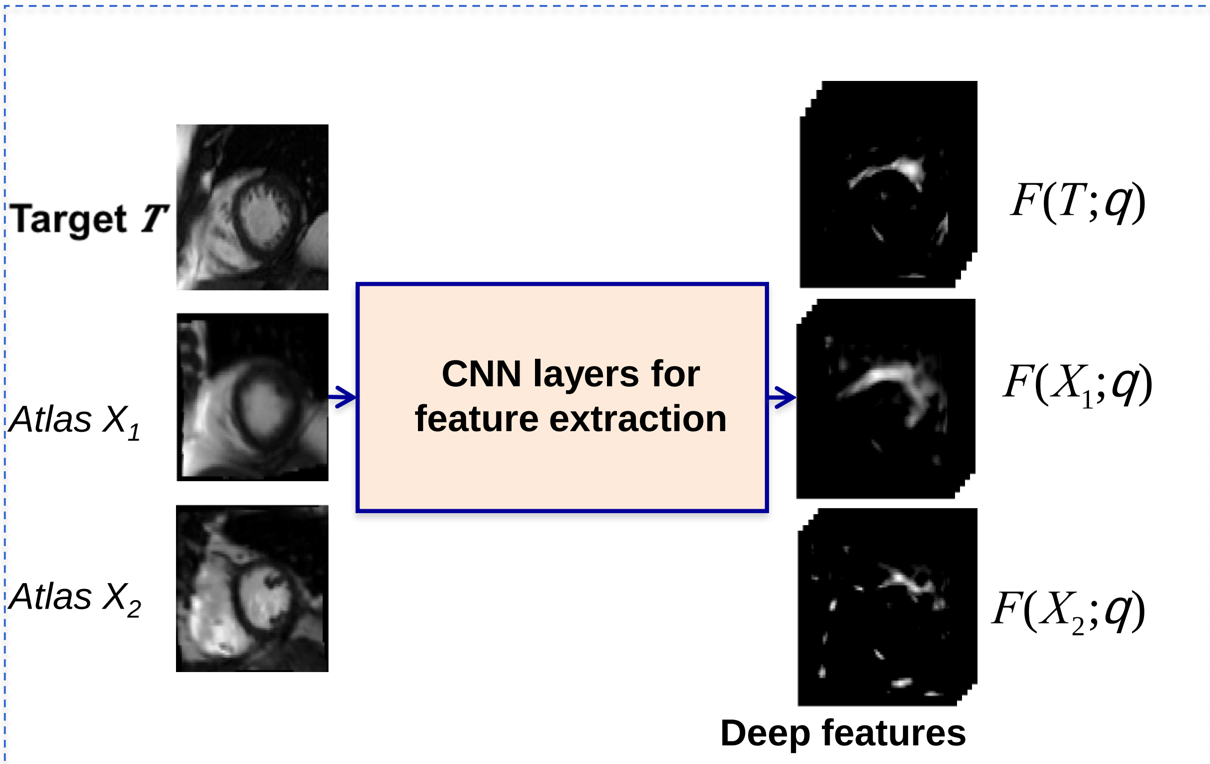
[1] Coupe, P., et al. Patch-based segmentation using expert priors: Application to hippocampus and ventricle segmentation. (NeuroImage 2011)

[2] Wang Z, et al. Geodesic patch-based segmentation. (MICCAI 2014)

[3] Bai, W., et al. Multi-atlas segmentation with augmented features for cardiac MR images. (Med. Image Anal. 2015)

Deep Fusion Net for MR Image Segmentation

- *Deep Fusion Net (MICCAI 2016)*: An end-to-end learnable deep architecture for NL-PLF concatenating feature extraction and non-local patch-based label fusion

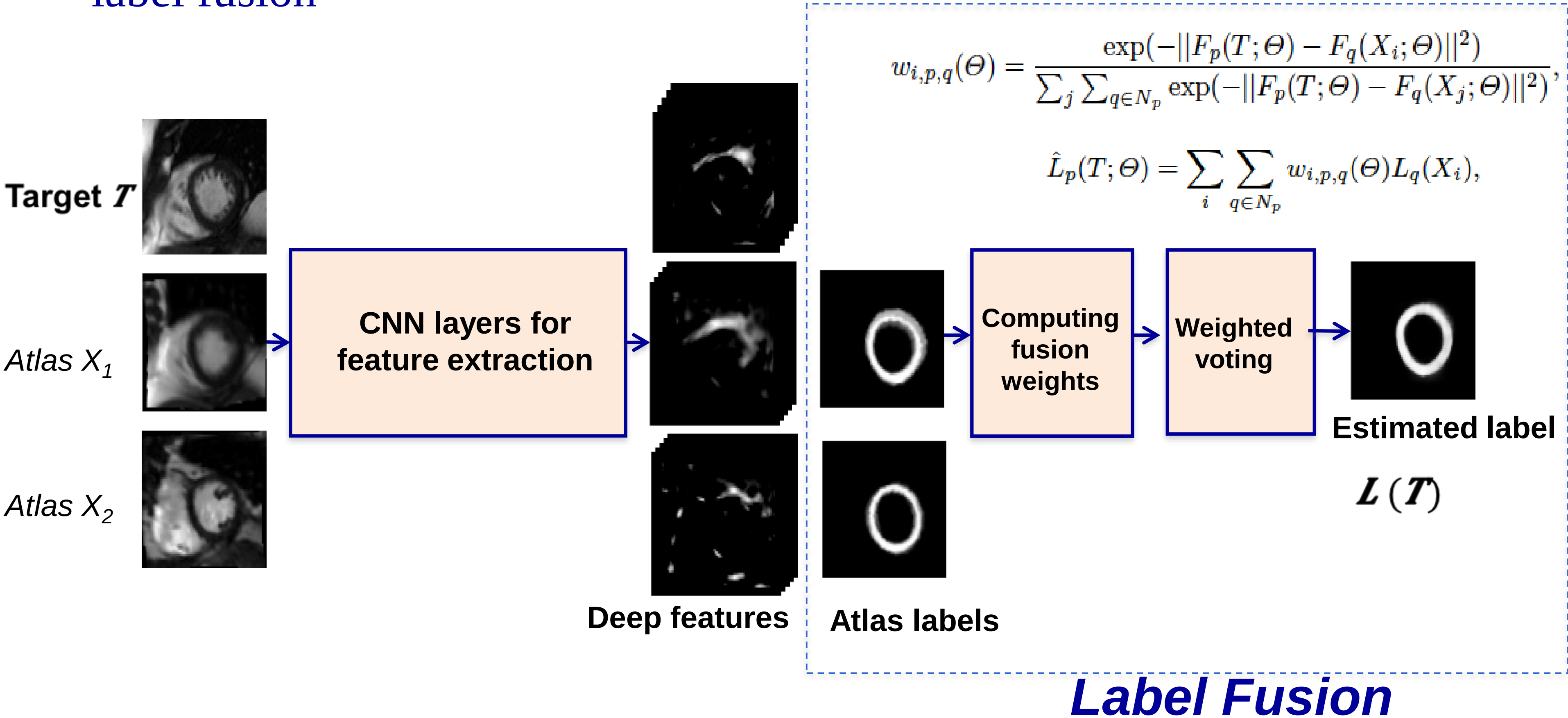


Feature extraction

[H. R. Yang, J. Sun, et al., MICCAI 2016, Medical Image Analysis, 2018]

Deep Fusion Net for MR Image Segmentation

- *Deep Fusion Net (MICCAI 2016)*: An end-to-end learnable deep architecture for NL-PLF concatenating feature extraction and non-local patch-based label fusion



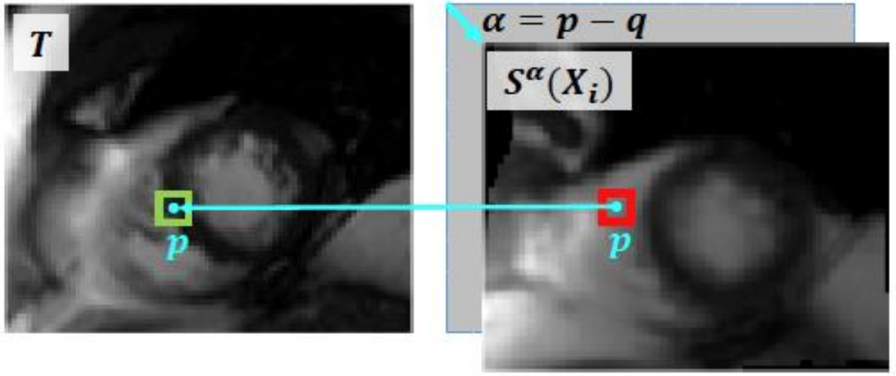
[H. R. Yang, J. Sun, et al., MICCAI 2016, Medical Image Analysis, 2018]

Deep Fusion Net for MR Image Segmentation

Implementation of Label Fusion Sub-Net

$$w_{i,p,q}(\Theta) = \frac{\exp(-\|F_p(T;\Theta) - F_q(X_i;\Theta)\|^2)}{\sum_j \sum_{q\in N_p} \exp(-\|F_p(T;\Theta) - F_q(X_j;\Theta)\|^2)},$$

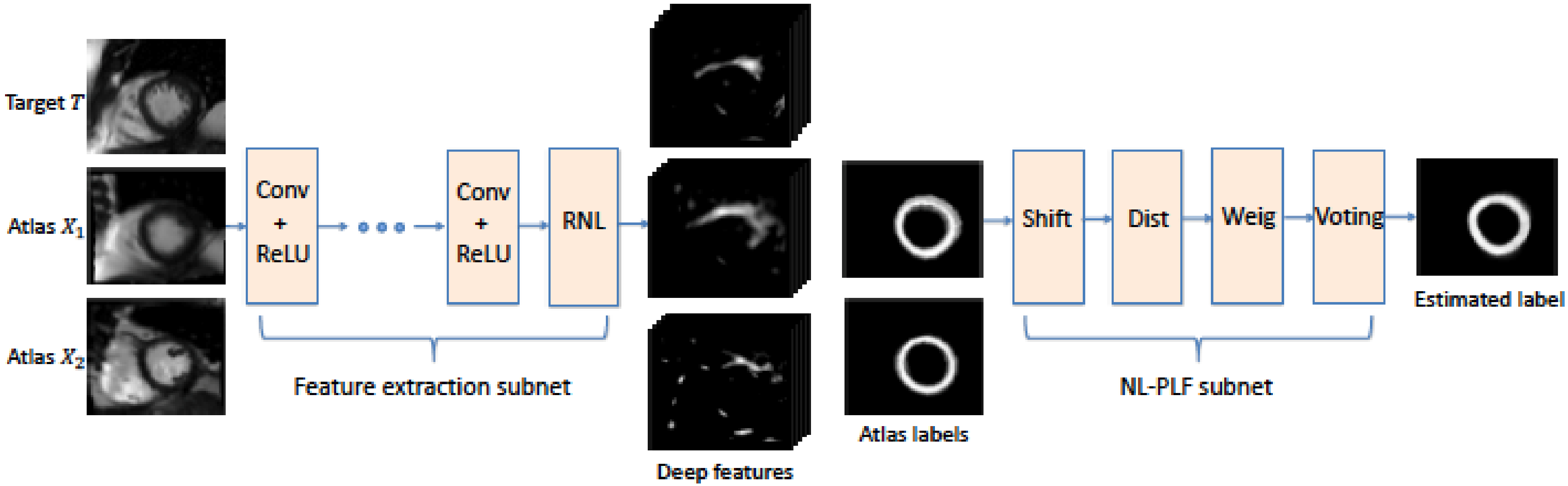
$$\hat{L}_p(T;\Theta) = \sum_i \sum_{q\in N_p} w_{i,p,q}(\Theta) L_q(X_i),$$



- ***Shift Layer***: spatially shifts features and labels along each direction $\alpha \in R_{nl} = \{(u,v) \in \mathbb{Z}^2 | -t \le u, v \le t\}$.
- ***Distance Layer***: $D_p^{\alpha}(T, X_i) = \left\|[C(F(T))]_p - [C(S^{\alpha}(F(X_i))]_p\right\|^2.$
- ***Weight Layer***: $w_{i,p,q} = w_p^{\alpha}(X_i) = \frac{exp(-D_p^{\alpha}(T,X_i))}{\sum_j \sum_{\alpha\in R_{nl}} exp(-D_p^{\alpha}(T,X_j))}.$
- ***Voting Layer***: $\hat{L}_p(T) = \sum_i \sum_{\alpha\in R_{nl}} w_p^{\alpha}(X_i) \left[C\left(S^{\alpha}(L(X_i))\right)\right]_p.$
- ***Loss layer***: $E\left(\hat{L}(T;\Theta), L(T)\right) = \frac{1}{|T|}\left\|\hat{L}(T;\Theta) - L(T)\right\|^2.$

Deep Fusion Net for MR Image Segmentation

- **Network structure**

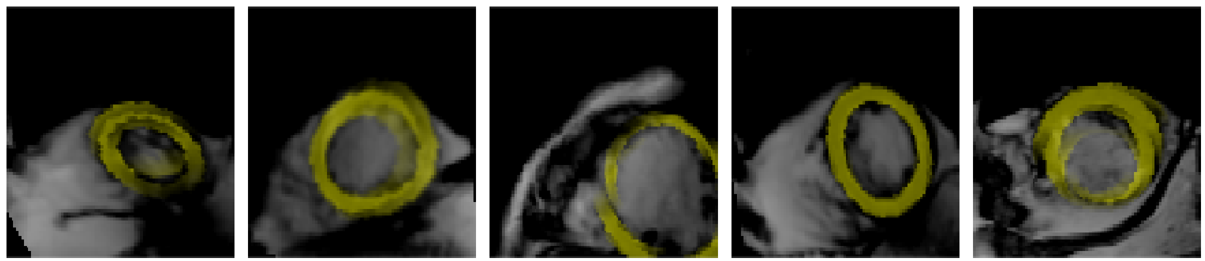


Deep Fusion Net for MR Image Segmentation

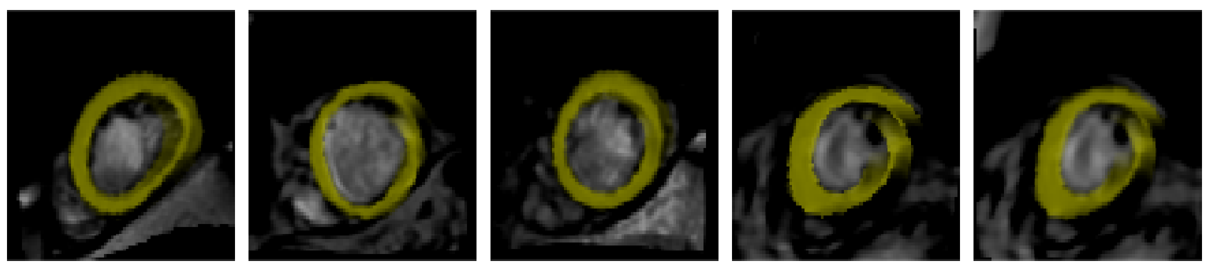
- **Atlas selection**

Deep feature distance: $d(T, X_i) = \|F(T) - F(X_i)\|$

A target image



Top-5 atlas images selected by normalized mutual information(NMI).

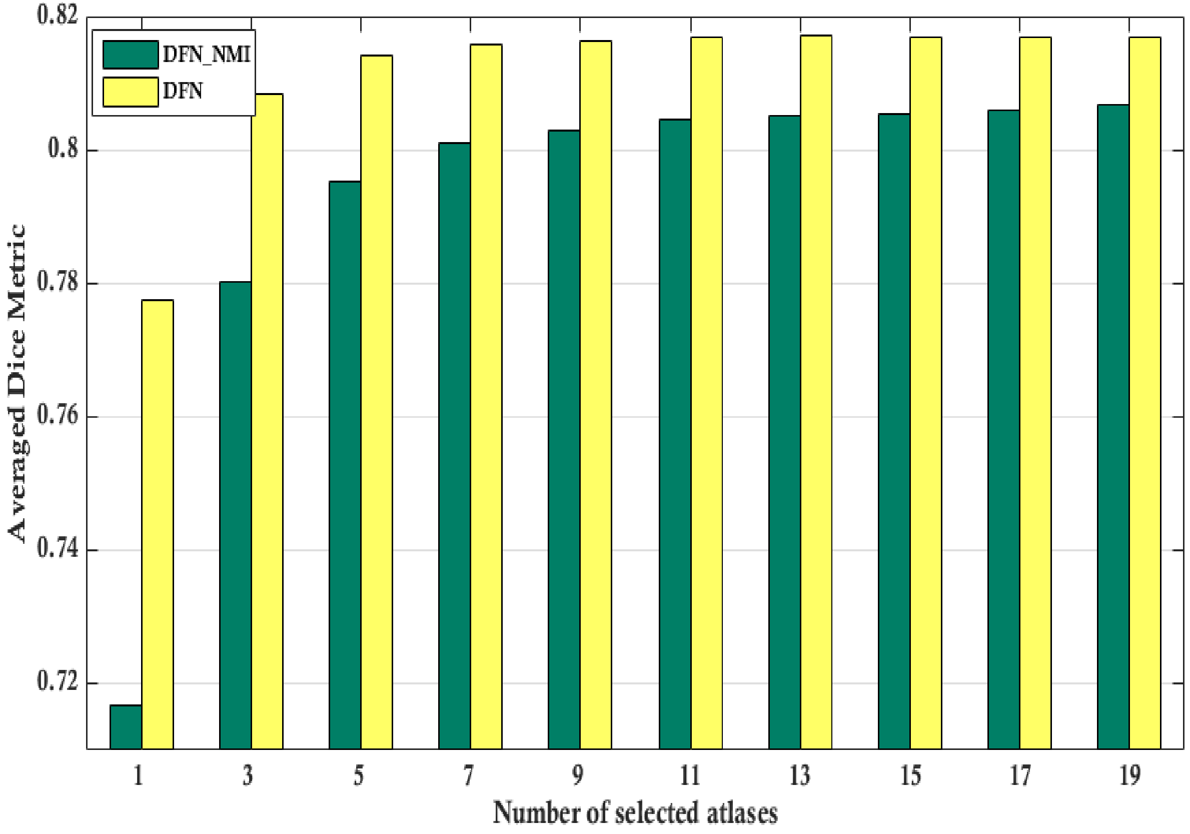


Top-5 atlas images selected by deep feature distance.

Database: MICCAI 2013 SATA Segmentation Challenge

Deep Fusion Net for MR Image Segmentation

- **Atlas selection**



Deep Fusion Net for MR Image Segmentation

- **Segmentation accuracy**

Groundtruth MV PB [1] MAPM [2] SVMAF [3] CNN DFN

Method	MV	PB [1]	MAPM [2]	SVMAF [3]	CNN	DFN_NMI	DFN
Accuracy	0.653	0.683	0.754	0.726	0.681	0.803	**0.816**

Table 1. The mean Dice metrics of different methods.

MICCAI 2013 SATA Dataset

[1] Coupe, P., et al. Patch-based segmentation using expert priors: Application to hippocampus and ventricle segmentation. (NeuroImage 2011)

[2] Shi, W., et al. Cardiac image super-resolution with global correspondence using multi-atlas patchmatch. (MICCAI 2013)

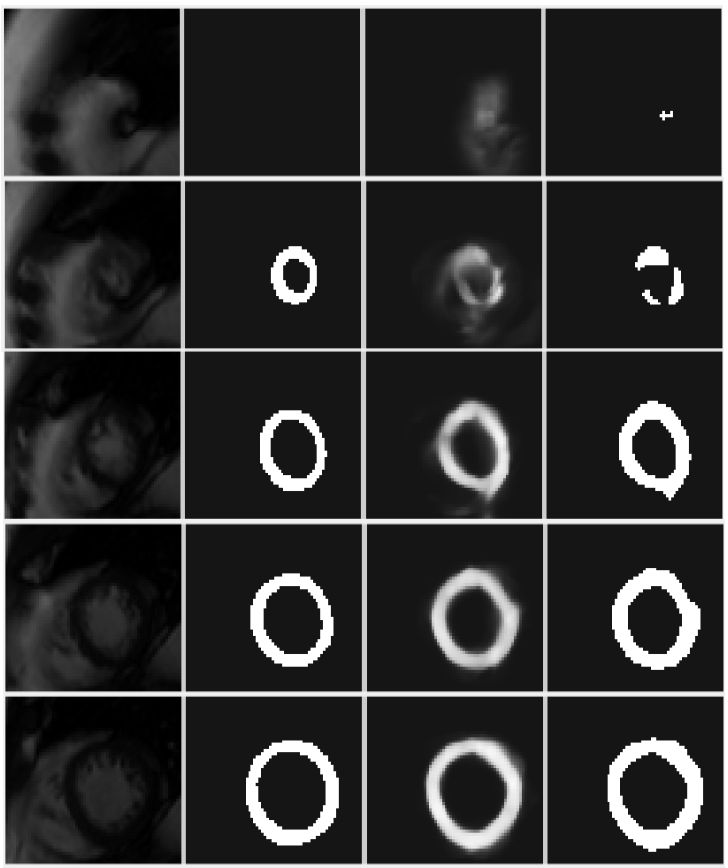
[3] Bai, W., et al. Multi-atlas segmentation with augmented features for cardiac MR images. (Med. Image Anal. 2015)

Deep Fusion Net for MR Image Segmentation

● Examples of results

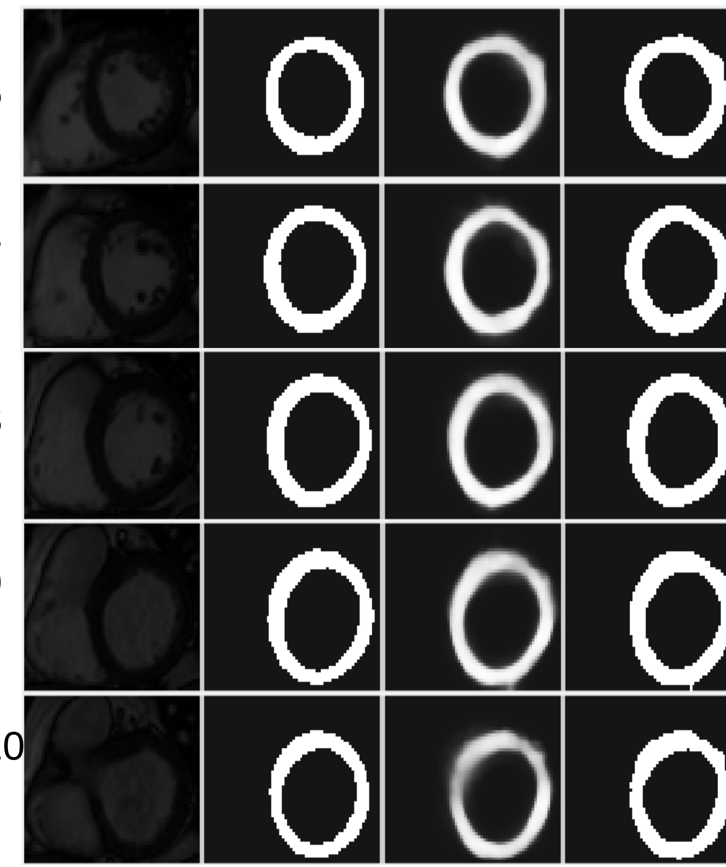
Target | Ground-truth | Output | Segment

Slice 1
Slice 2
Slice 3
Slice 4
Slice 5



Target | Ground-truth | Output | Segment

Slice 6
Slice 7
Slice 8
Slice 9
Slice 10



Deep Fusion Net for MR Image Segmentation

Method	Epicardium ADM	Epicardium AJM
DFN	**0.9453**(0.0228)	**0.8972**(0.0399)
DLLS	0.94(0.02)	–
DLDM	0.94(0.02)	–
DLDM_init	0.89(0.03)	–
SVMAF	0.9259(0.0251)	0.8630(0.0419)
PB	0.9170(0.0318)	0.8483(0.0523)
MV	0.9155(0.0326)	0.8458(0.0535)

2009 LV segmentation challenge

ADM: averaged Dice Metric; AJM: averaged Jaccard Metric

Epicardium (心外膜)

DLLS: Combining deep learning and level set for the automated segmentation of the left ventricle of the heart from cardiac cine magnetic resonance. Medical Image Analysis, 2017

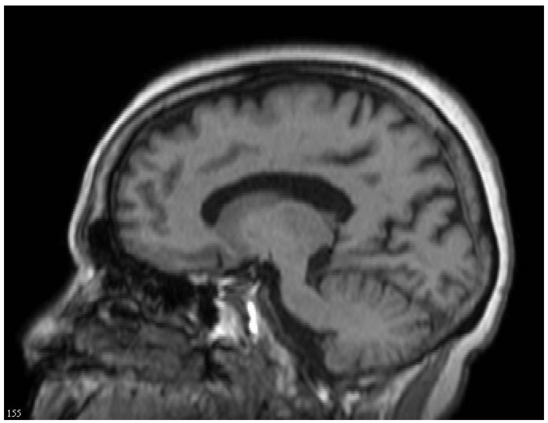
DLDM: A combined deep-learning and deformable-model approach to fully automatic segmentation of the left 545 ventricle in cardiac MRI, Medical Image Analysis, 2016

Outline

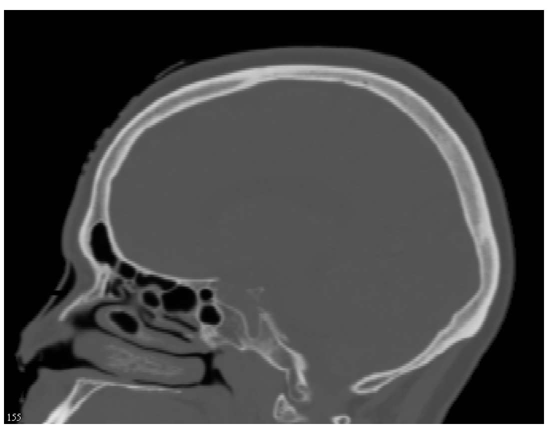
- Introduction
  - Background: *Image analysis / deep neural networks*
  - Motivation
- Model-driven Deep Learning Approach
  - Learning Markov Random Field Model for Image Restoration
  - Deep ADMM-Net for Fast Compressive Sensing MRI
  - Deep Fusion-Net for Multi-Atlas MR Image Segmentation
- Recent Progress
  - Multimodal medical image synthesis
  - Learning proximal operators
  - Learning Graph CNNs for 3D shape analysis
  - Learning to Optimize
- Discussion & Conclusion

Multi-modal Medical Image Synthesis

- Background



MR
(excellent soft-tissue contrast)



CT
(provide tissue electron densities)

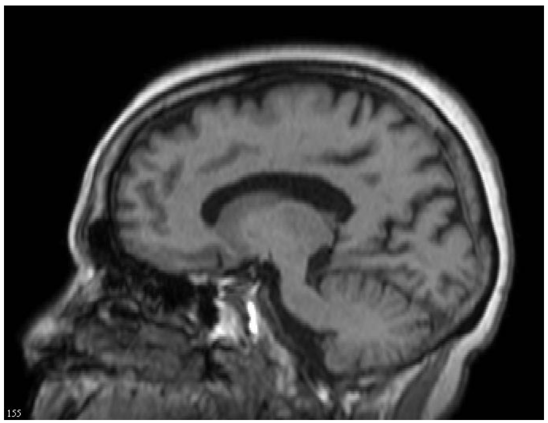
(Paired training data)

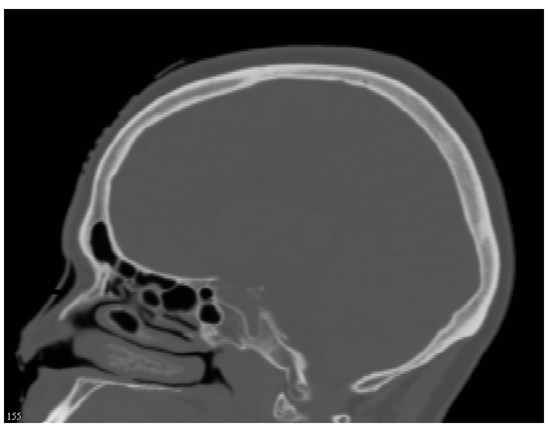
Atlas MR | **Target MR**

Atlas CT | **Target CT (unknown)**

Multi-modal Medical Image Synthesis

- Background



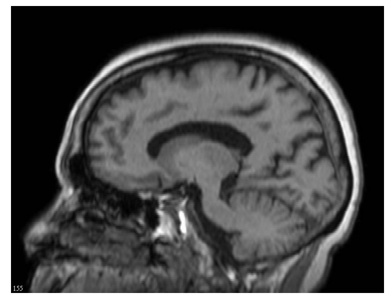


MR
(excellent soft-tissue contrast)

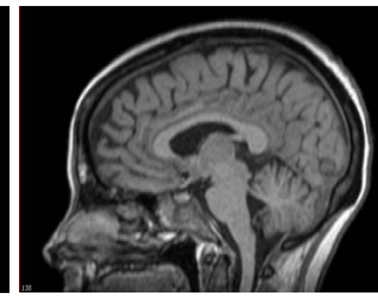
CT
(provide tissue electron densities)

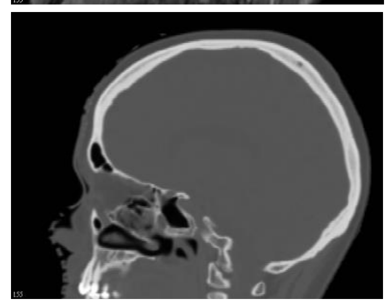
(Unpaired training data)

Atlas MR



Target MR





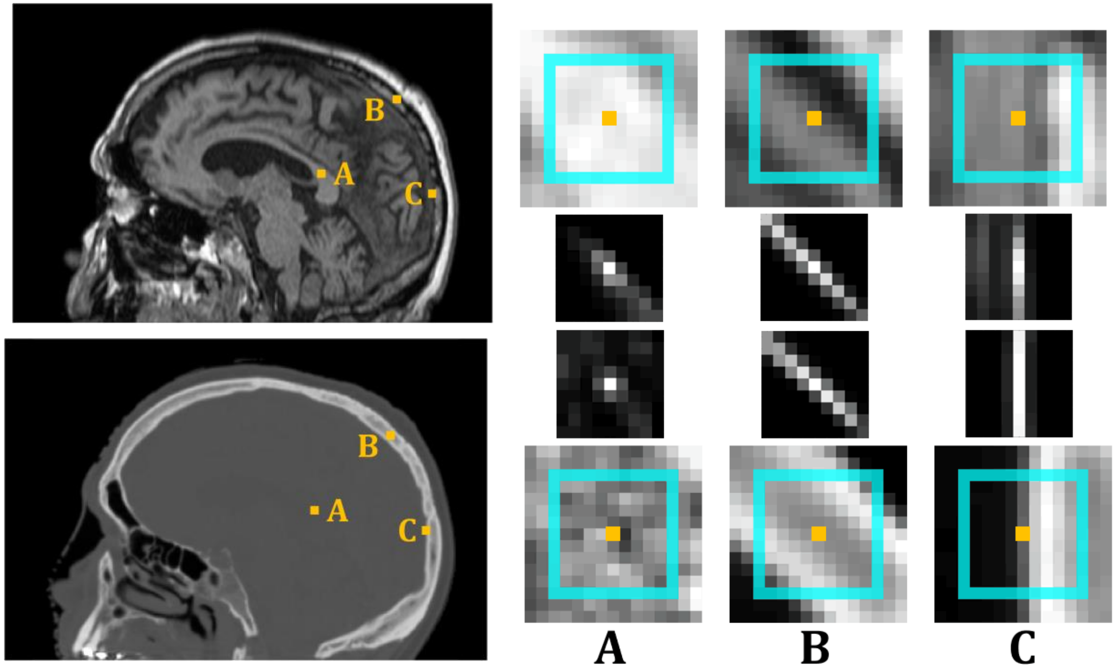
Atlas CT

?

Target CT (unknown)

Multi-modal Medical Image Synthesis

- MR images ⟹ CT Images



Non-local structure:

$$F_x^{(\alpha)}(I) = \frac{1}{Z}\exp\left(-\frac{D_{\mathcal{P}}(I, x, x+\alpha)}{V(I, x)}\right)$$

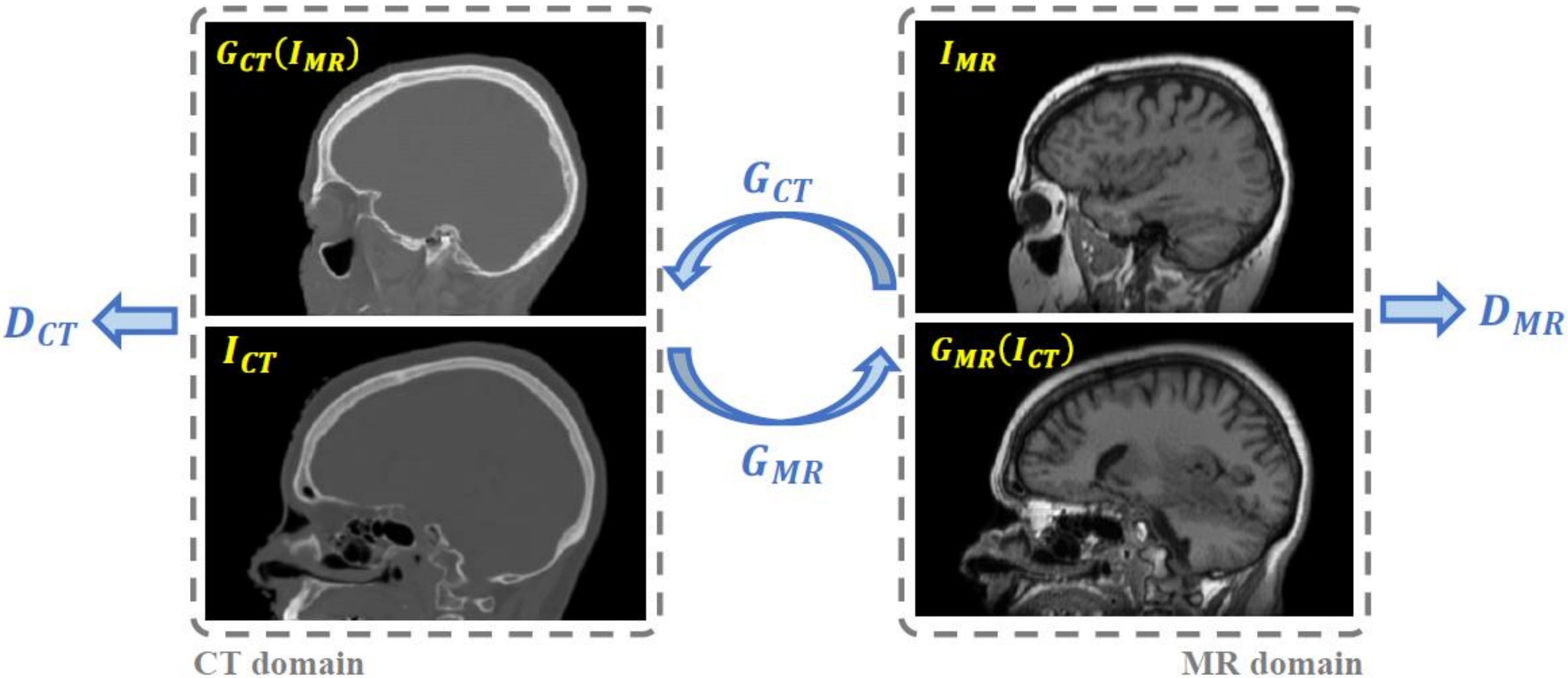
$$\mathcal{L}_{structure}(G_{MR}, G_{CT}) = \frac{1}{N_{MR}|R_{nl}|}\sum_x \left\| F_x\big(G_{CT}(I_{MR})\big) - F_x(I_{MR}) \right\|_1 + \frac{1}{N_{CT}|R_{nl}|}\sum_x \left\| F_x\big(G_{MR}(I_{CT})\big) - F_x(I_{CT}) \right\|_1$$

[H. R. Yang, J. Sun, et al., MICCAI-DLMIA, 2018]

Multi-modal Medical Image Synthesis

$$\mathcal{L}(G_{\mathrm{CT}}, G_{\mathrm{MR}}, D_{\mathrm{CT}}, D_{\mathrm{MR}}) = \mathcal{L}_{\mathrm{GAN}}(G_{\mathrm{CT}}, D_{\mathrm{CT}}) + \mathcal{L}_{\mathrm{GAN}}(G_{\mathrm{MR}}, D_{\mathrm{MR}}) + \lambda_1 \mathcal{L}_{\mathrm{cycle}}(G_{\mathrm{CT}}, G_{\mathrm{MR}}) + \boxed{\lambda_2 \mathcal{L}_{\mathrm{structure}}(G_{\mathrm{CT}}, G_{\mathrm{MR}})}$$

Training Loss



Multi-modal Medical Image Synthesis

- **Compared methods**

- ☐ "cycleGAN": Conventional cycleGAN
- ☐ "cycleGAN (paired)": CycleGAN trained with paired data

- **Evaluation:** MAE, PSNR, SSIM, SSIM(HG).

	MAE	PSNR	SSIM	SSIM (HG)
CycleGAN (unpaired)	150.28 (18.06)	23.09 (1.04)	0.732 (0.029)	0.546 (0.042)
CycleGAN (paired)	122.66^* (16.71)	24.57^* (1.25)	0.785^* (0.029)	0.630^* (0.043)
Proposed	127.78^* (16.21)	24.67^* (1.27)	0.780^* (0.026)	0.622^* (0.044)

$*$ denotes $p < 0.001$ compared to the conventional cycleGAN using a paired sample t

Learning proximal operators

- Learning proximal operators for optimization ([ECCV, 2018])

$$\frac{I^c(\mathrm{x})}{A^c} = \frac{J^c(\mathrm{x})}{A^c} T(\mathrm{x}) + (1 - T(\mathrm{x})), c \in \{r, g, b\}.$$

$$E(Q,T) = \frac{\alpha}{2} \sum_{c \in \{r,g,b\}} \|Q^c \circ T + 1 - T - P^c\|_F^2 + \frac{\beta}{2} \|Q^{dk} \circ T + 1 - T - P^{dk}\|_F^2 + f(T) + g(Q^{dk}),$$

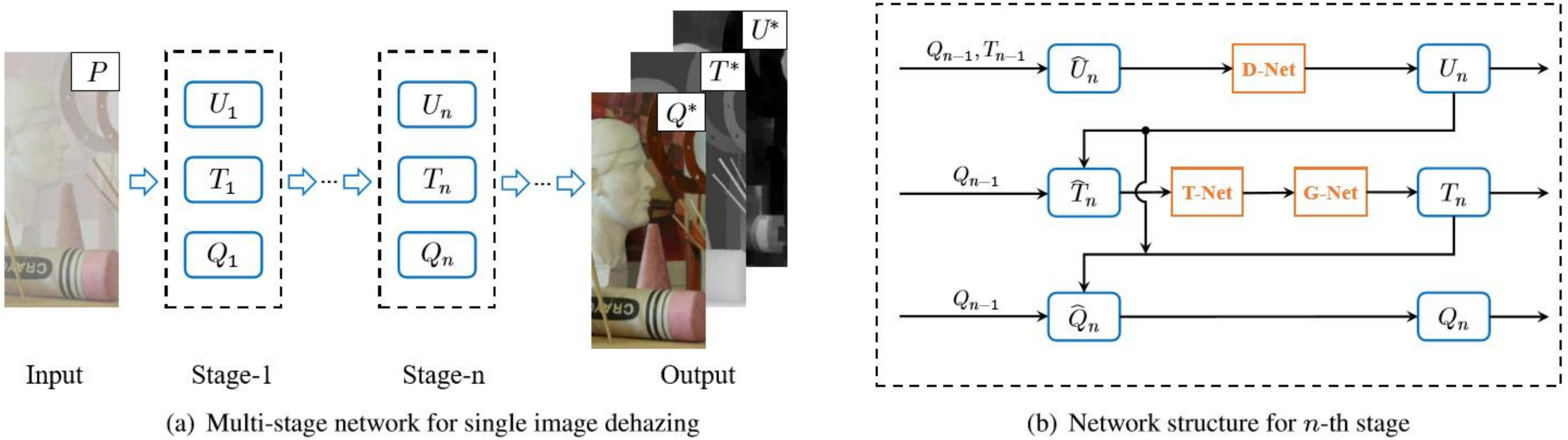
$$E(Q,T,U) = \frac{\alpha}{2} \sum_{c \in \{r,g,b\}} \|Q^c \circ T + 1 - T - P^c\|_F^2 + \frac{\beta}{2} \|U \circ T + 1 - T - P^{dk}\|_F^2 + \frac{\gamma}{2} \|U - Q^{dk}\|_F^2 + f(T) + g(U),$$

$$U_n = \mathrm{prox}_{\frac{1}{b_n} g}\left(\hat{U}_n\right),$$

$$T_n = \mathrm{prox}_{\frac{1}{c_n} f}\left(\hat{T}_n\right),$$

$$\overrightarrow{Q}_n = \frac{\alpha(\overrightarrow{P} + \overrightarrow{T_n} - 1) \circ \overrightarrow{T_n} + \gamma D^\top \overrightarrow{U_n}}{\alpha \overrightarrow{T_n} \circ \overrightarrow{T_n} + \gamma \mathrm{diag}(D^\top D)}.$$

Learning proximal operators



Proximal-Dehaze Network Structure *[ECCV 2018]*

Learning proximal operators

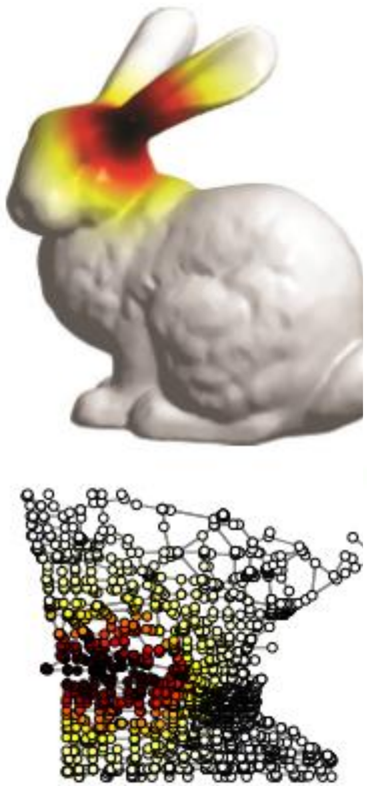
Learning proximal operators

Learning proximal operators

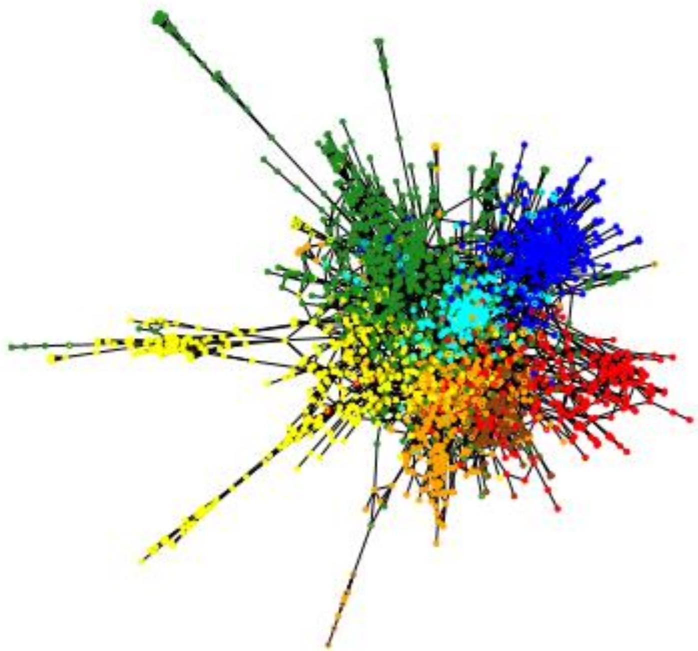
Learning proximal operators

Learning on 3D shapes

- Matrix Deep Learning / Graph-based Deep Learning



Shape

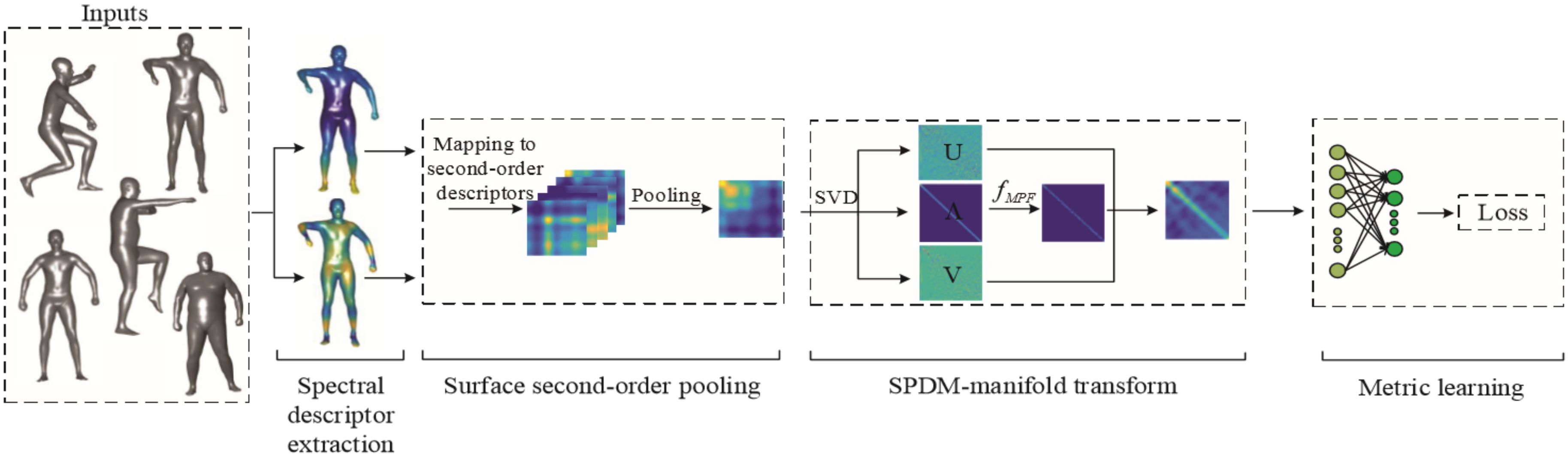


Data graph

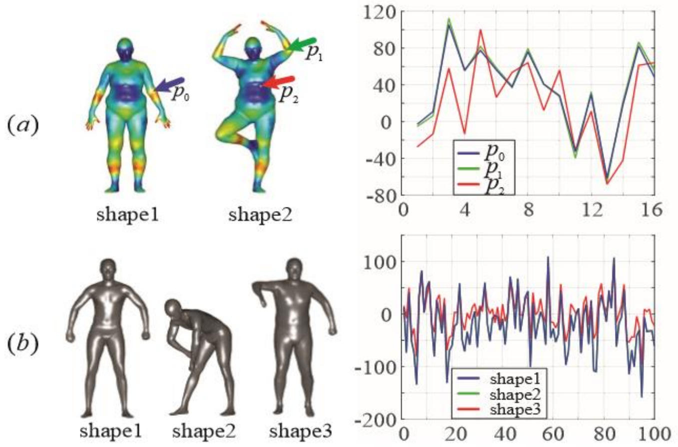
Graph representation:

Graph ⇨ Matrix
Hyper-graph ⇨ Tensor

Learning on 3D shapes

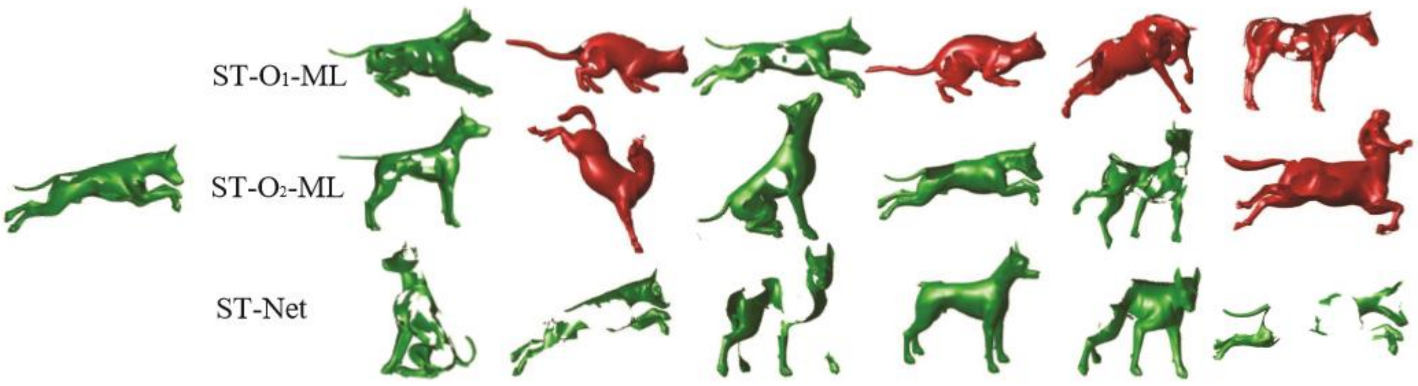


Method	Synthetic					Real				
	NN	1-T	2-T	E-M	DCG	NN	1-T	2-T	E-M	DCG
CSDLMNN [12]	99.67	98.02	99.86	51.14	99.63	97.92	92.78	98.68	27.03	97.60
Surf-ML-Net	**100**	96.92	99.95	51.16	99.65	96.67	91.92	98.33	27.03	96.78
ST-Net (w/o SPDM-T)	**100**	**100**	**100**	51.16	**100**	98.75	96.08	99.58	27.03	99.93
ST-Net	**100**	**100**	**100**	51.16	**100**	**100**	**99.83**	**100**	27.03	**99.98**

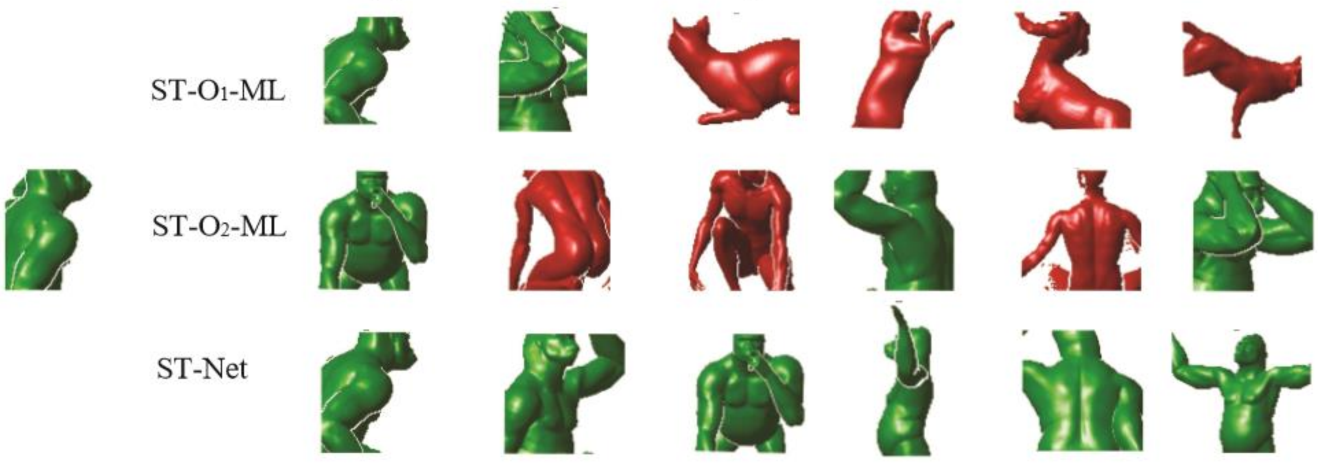


Spectral Network *[ECCV-GMDL, 2018]*

Learning on 3D shapes



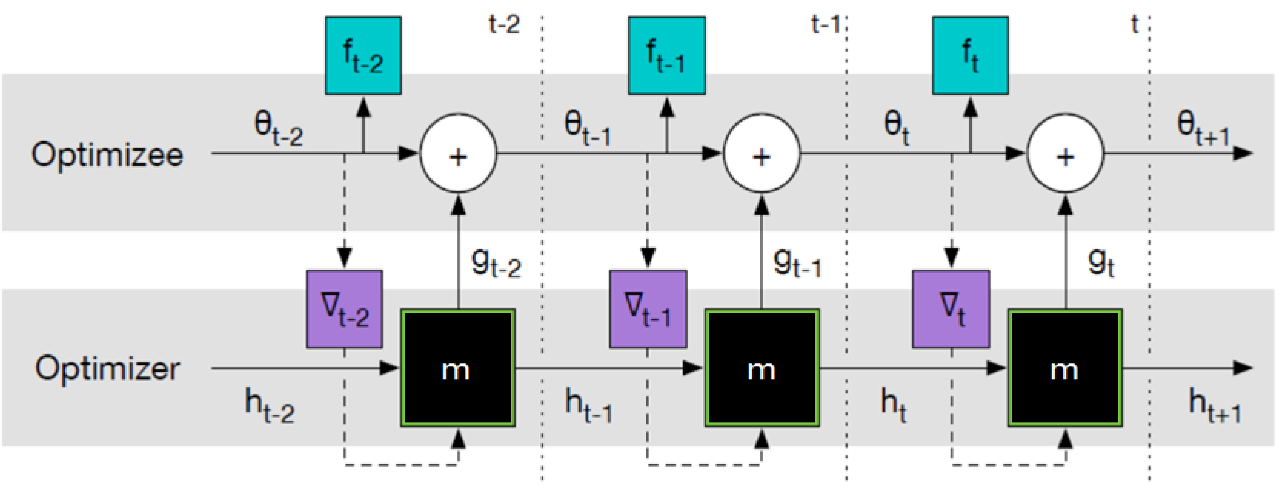
(a) Retrieval results for a shape with "holes"



(b) Retrieval results for a range data

Learning to optimize

- Network optimizers
  - Traditional approach designed by experts
    SGD, Adam, RMSProp, AdaGrad,….
  - Learning-based approach
    Learn the optimizer by Recurrent Neural Network



$$g_t = m(\mathrm{h}_t, \nabla_t)$$

RNN: *black-box*

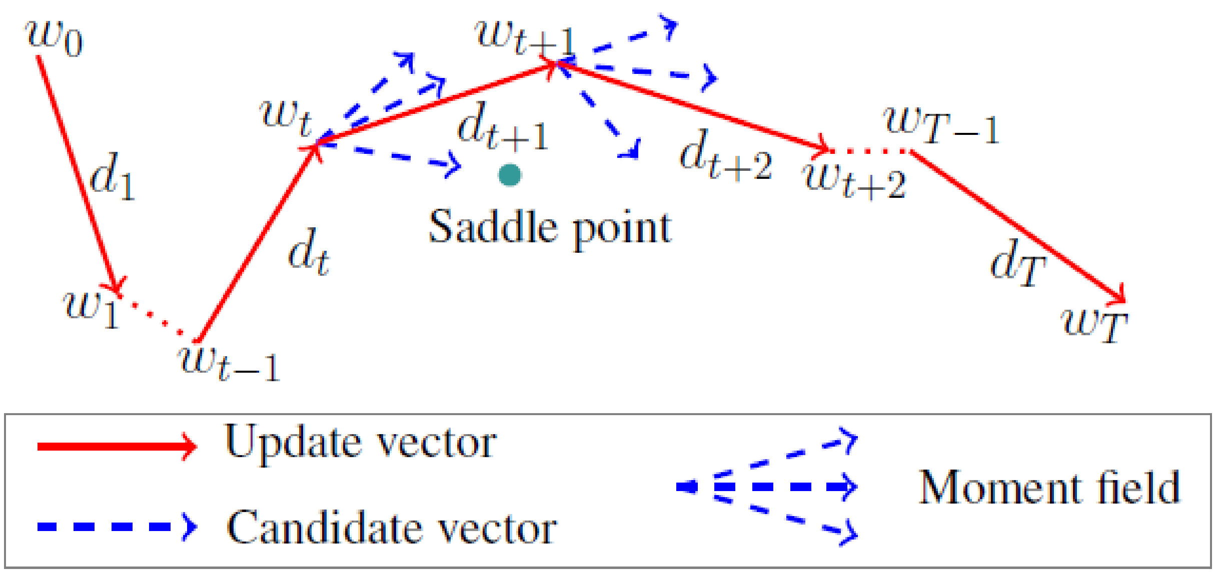
Andrychowicz, Marcin, et al，Learning to Learn by Gradient Descent by Gradient Descent. In NIPS，2016

Learning to optimize

- Hyper-Adam [*AAAI 2019*]:

In each iteration of network parameter updating:

- Generate multiple parameter updates using Adam with multiple weight decay rates
- Adaptive combination of updates to generate final update



Learning to optimize

Algorithm 2 Task-Adaptive HyperAdam

Require:

Initialized parameter w_0, step size α, batch size N_B.

Dataset $\{(x_i, y_i)\}_{i=1}^{N}$.

Initialize:

$\mathbf{m}_0, \mathbf{v}_0, \hat{\boldsymbol{\beta}}_0, \hat{\boldsymbol{\gamma}}_0, \mathbf{s}_0 = \mathbf{0} \in \mathbb{R}^{p \times J}$, $\mathbf{1} \in \mathbb{R}^{p \times J}$, ε=1e-24 .

for all $t = 1, \ldots, T$ **do**

Draw random batch $\{(x_{i_k}, y_{i_k})\}_{k=1}^{N_B}$ from dataset

$g_t = \Sigma_{k=1}^{N_B} \nabla l(x_{i_k}, y_{i_k}, w_{t-1})$

$\mathbf{G_t} = [g_t, \ldots, g_t]$ ▷ $\mathbf{G_t} \in \mathbb{R}^{p \times J}$

$\mathbf{s}_t = F_h(\mathbf{s}_{t-1}, g_t; \Theta_h)$ ▷ *current state*

$\boldsymbol{\beta}_t \triangleq [\beta_t^1, \ldots, \beta_t^J] = F_u(\mathbf{s}_t, \mathbf{m}_{t-1}; \Theta_u)$

$\boldsymbol{\gamma}_t \triangleq [\gamma_t^1, \ldots, \gamma_t^J] = F_r(\mathbf{s}_t, \mathbf{m}_{t-1}; \Theta_r)$

$\mathbf{m}_t = \boldsymbol{\beta}_t \odot \mathbf{m}_{t-1} + (\mathbf{1} - \boldsymbol{\beta}_t) \odot \mathbf{G_t}$

$\mathbf{v}_t = \boldsymbol{\gamma}_t \odot \mathbf{v}_{t-1} + (\mathbf{1} - \boldsymbol{\gamma}_t) \odot \mathbf{G}_t^2$

$\hat{\boldsymbol{\beta}}_t = \boldsymbol{\beta}_t \odot \hat{\boldsymbol{\beta}}_{t-1} + (\mathbf{1} - \boldsymbol{\beta}_t) \odot \mathbf{1}$

$\hat{\boldsymbol{\gamma}}_t = \boldsymbol{\gamma}_t \odot \hat{\boldsymbol{\gamma}}_{t-1} + (\mathbf{1} - \boldsymbol{\gamma}_t) \odot \mathbf{1}$

$\tilde{\mathbf{m}}_t = \mathbf{m}_t / \hat{\boldsymbol{\beta}}_t$, $\tilde{\mathbf{v}}_t = \mathbf{v}_t / \hat{\boldsymbol{\gamma}}_t$, ▷ *correcting bias*

$\hat{\mathbf{m}}_t \triangleq [\hat{m}_t^1, \ldots, \hat{m}_t^J] = \frac{\tilde{\mathbf{m}}_t}{\sqrt{\tilde{\mathbf{v}}_t + \varepsilon}}$ ▷ *moment field*

$\boldsymbol{\rho}_t \triangleq [\rho_t^1, \ldots, \rho_t^J] = F_q(\mathbf{s}_t; \Theta_q)$ ▷ *weight field*

$d_t = \Sigma_{j=1}^{J} \rho_t^j \odot \hat{m}_t^j$

$w_t = w_{t-1} - \alpha d_t$

end for

return final parameter w_T.

Hyper-Adam Algorithm

Current State

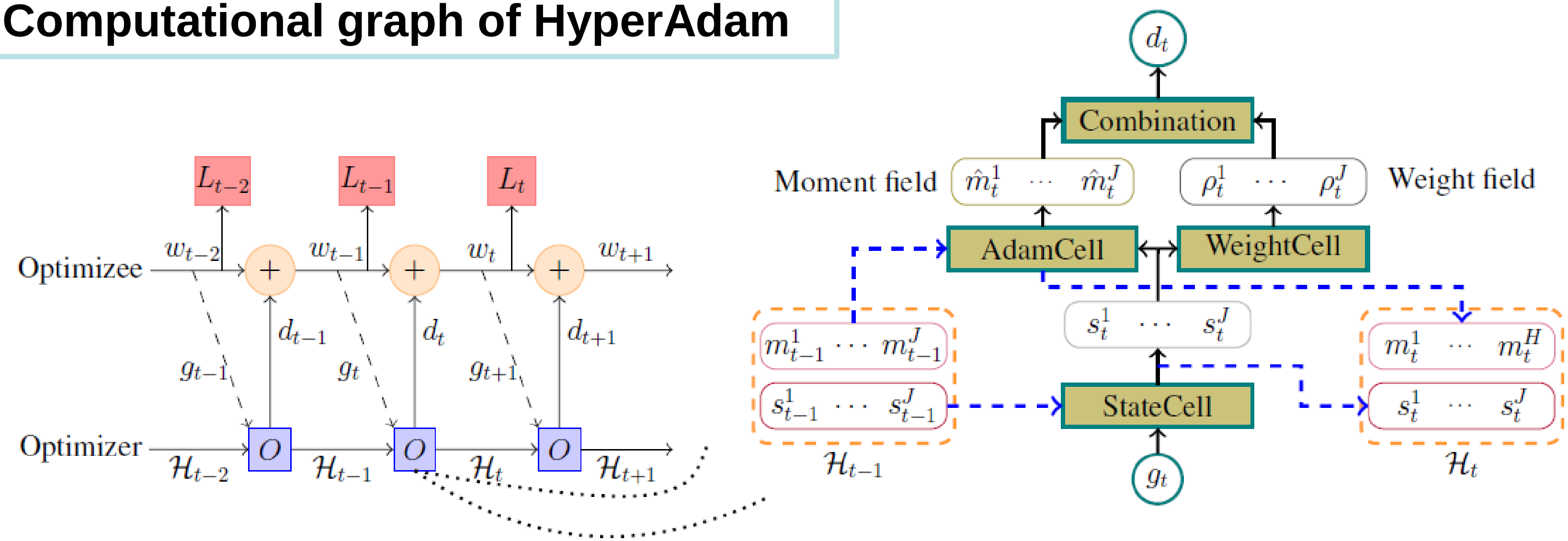
Determining multiple groups of hyper-parameters

Generating multiple candidate updates with corresponding hyper-parameters in parallel

Combining these updates to get the final update using adaptively learned combination weights

Learning to optimize

Computational graph of HyperAdam



O: optimizer; g_t: gradient of the optimizee L; d_t: update vectors

StateCell: encoding the current state $S_t = [s_t^1, \ldots, s_t^J]$

AdamCell: outputting moment field that contains multiple candidate update vectors

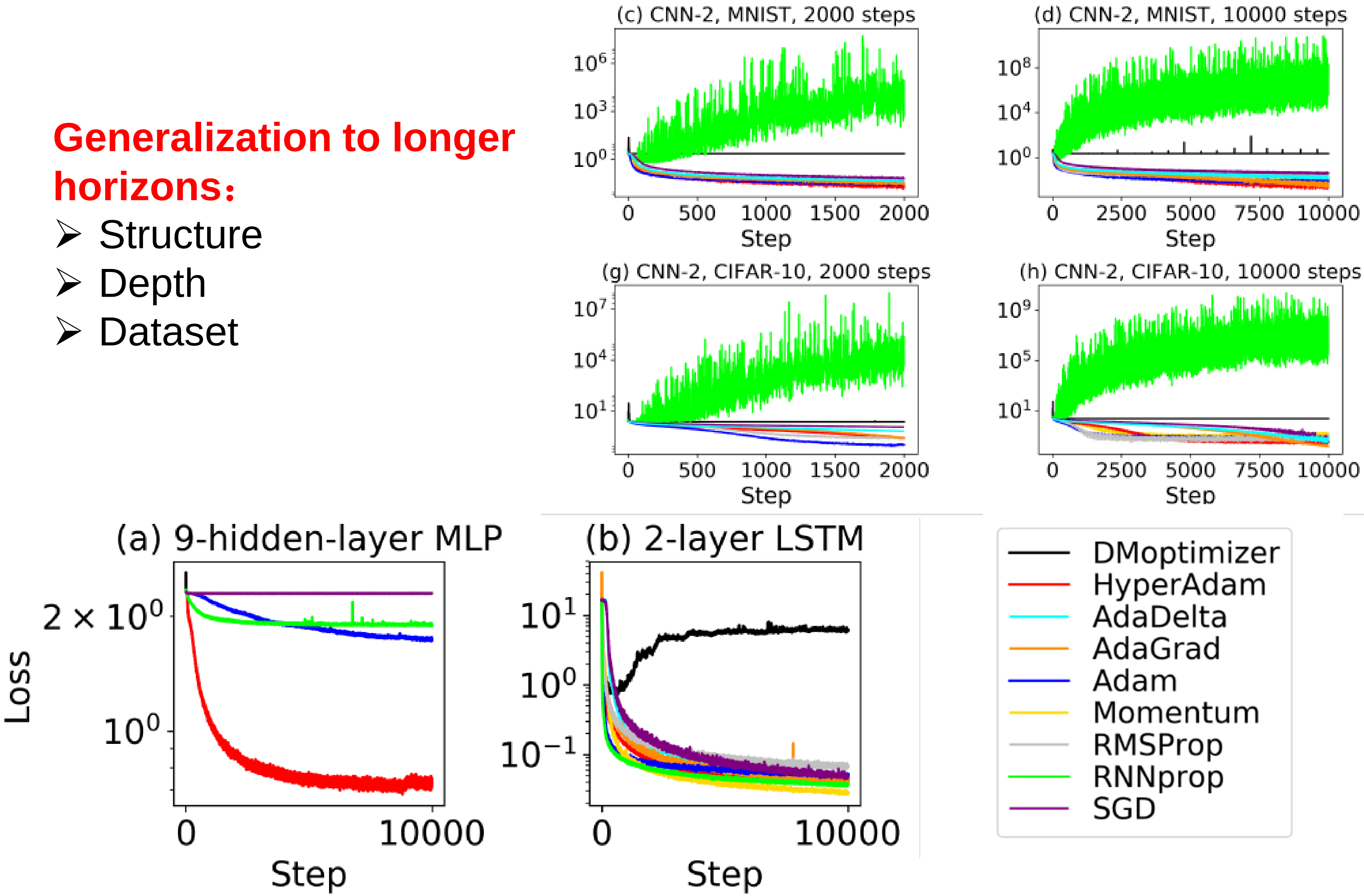
WeightCell: outputting weight field that contains multiple weight vectors

Combination: combining these candidate vectors to give the final update vector

Learning to optimize

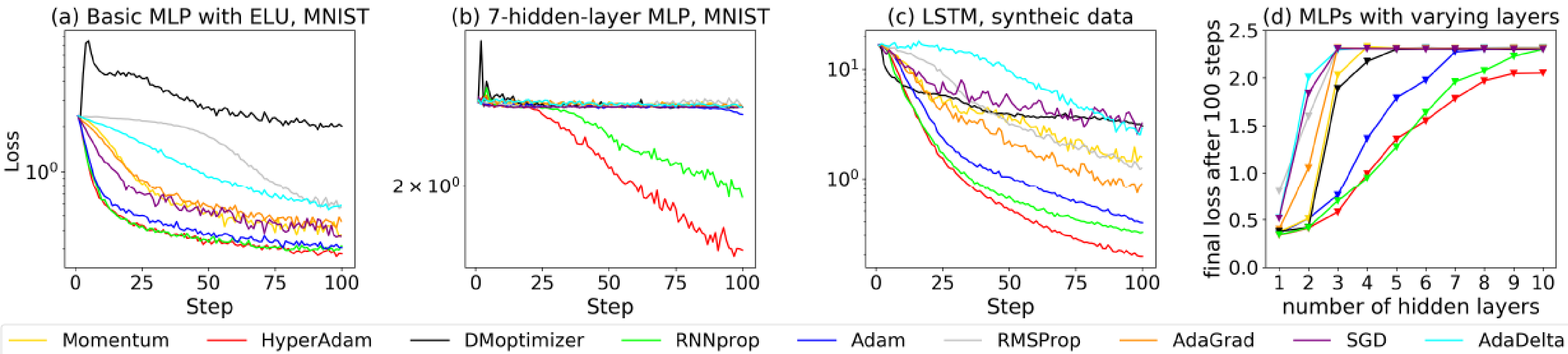
Generalization to longer horizons:

- Structure
- Depth
- Dataset



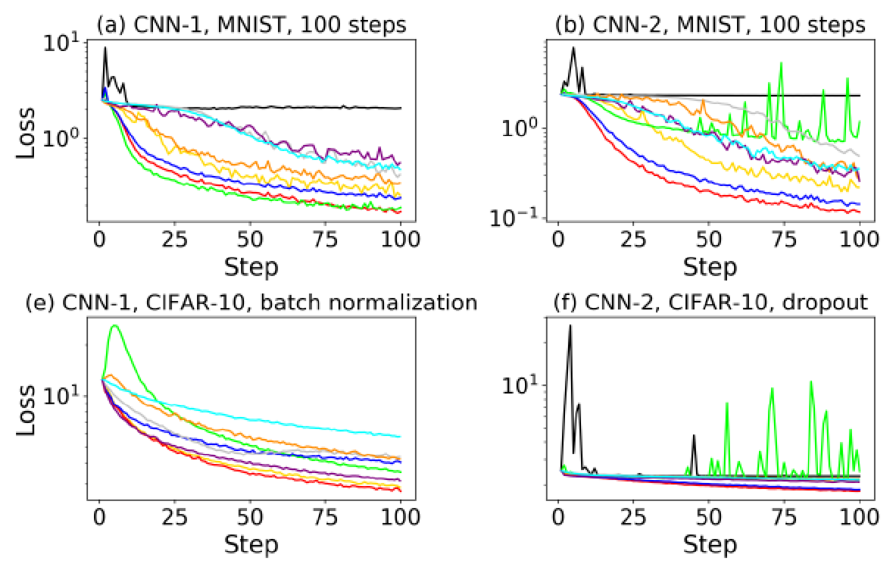
Learning to optimize

Generalization with fixed steps



Activation	Adam	DMoptimizer	RNNprop	HyperAdam
sigmoid	0.35	0.38	0.34	**0.33**
ReLU	0.32	1.42	0.31	**0.29**
ELU	0.31	2.02	0.31	**0.28**
tanh	0.34	0.83	**0.33**	0.36

Table 1: Performance for training basic MLP in 100 steps with different activation functions. Each value is the average final loss for optimizing networks in 100 times.



Task	Adam	DMoptimizer	RNNprop	HyperAdam
Baseline	0.65	3.10	0.49	**0.42**
Small noise	0.39	3.06	0.32	**0.19**
2-layer	0.51	2.05	0.27	**0.26**

Table 2: Performance on different sequence prediction tasks.

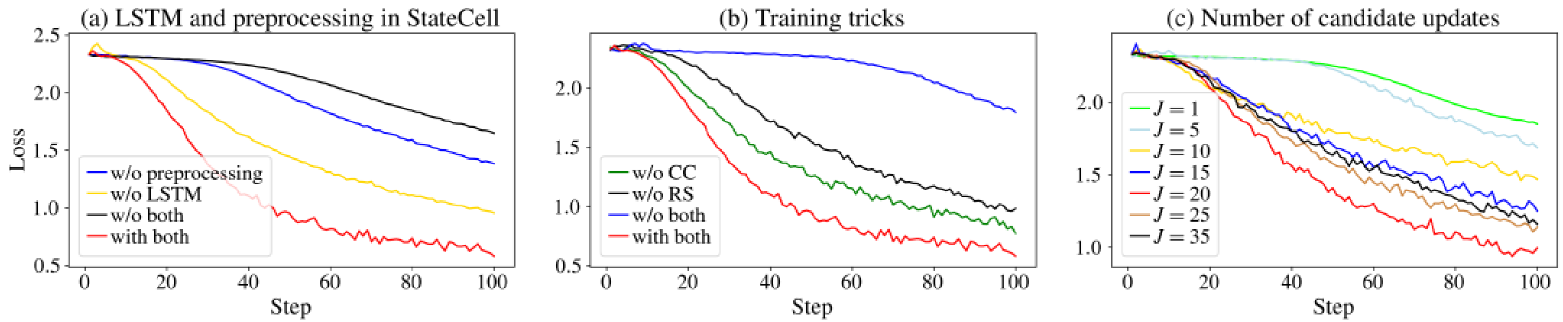
Learning to optimize

Generalization of the Learners

Task	Measure	Adam	DMoptimizer	RNNprop	HyperAdam
CNN-1 (MNIST)	loss	0.10	2.30	0.36	**0.05**
	top-1	**98.50%**	10.10%	96.46%	98.48%
	top-2	99.59%	20.38%	99.03%	**99.63%**
CNN-2 (MNIST)	loss	0.09	2.30	2.30	**0.07**
	top-1	98.98%	11.35%	11.37%	**99.02%**
	top-2	**99.80%**	21.45%	21.69%	99.78%

Table 3: Generalization of the learner trained by Adam, DMoptimizer, RNNprop and HyperAdam for 10000 steps.

Ablation Study



Summary

- Summarization:

Model-driven Deep Learning: proposed deep learning approaches by taking the merits of modeling-based approach and deep learning-based approach

  – Gradient descent for energy minimization → deep CNN
  – ADMM algorithm → deep ADMM-net
  – Non-local approach -> deep fusion-net
  – Graph-based deep models

- Current work (IMAGINE: Image Intelligence Group)
  - Deep learning on graphs / manifolds
  - Learning to learn
  - Applications: Natural & medical images analysis / data analysis

Thanks for your attention!