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

● 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.

\[ Y = AX + \varepsilon \]

*Physical imaging model*

*Restoration & Reconstruction*

*Inverse Problems*
Backgrounds--Image Processing & Analysis

- **Segmentation & Recognition**

Semantic Segmentation

Lesion (Pulmonary nodule) localization and classification
Conventional Models: Signal processing approaches

- **Wavelets**

- **Image Filtering**
Backgrounds--Models

- **Conventional Models:** *Energy model and its optimization*
  - *Energy Model with Regularization*
    \[ x^* = \arg \min_x D(x, y; w) + R(w) \]
  - *Dictionary Learning*
    \[ \{D^*, x^*\} = \arg \min_{D,x} ||D\alpha - y|| + ||\alpha||_p \]

*Applications:* Image Restoration / Segmentation / Classification / MRI / Lesion detection
Conventional Models: statistical models

Evidence lower bound (ELBO)

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

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:**

- **GAN**

Noises: $Z$

Training data: $X$

[Ian Goodfellow et al., 2014]

[Hochreiter & Schmidhuber, 1997]
**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
  - Gradient descent, ADMM, proximal operators, 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

● 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 \([J. \text{ Sun, M. Tappen, CVPR 2011}]\)
  - A novel Markov random field model
  - Discriminative parameter learning

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

\[
x^*(\Theta) = \arg\min_x \left\{ E(x \mid y, \Theta) = E_{\text{data}}(y \mid x) + E_{\text{prior}}(x; \Theta) \right\}
\]
Example

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

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

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

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

*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

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

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

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

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

\[
\Theta^* = \arg \min_\Theta L(x^K(\Theta), t)
\]

where \(x^K(\Theta) = \text{GradDesc}_K \{ E(x \mid y, \Theta) \} \)
**Example**

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

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

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

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

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

Non-local Range MRF

\[
\Theta^* = \arg \min_{\Theta} L(x^K (\Theta), t)
\]

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

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

**KEY IDEA:**

\[ \Theta^* = \underset{\Theta}{\operatorname{argmin}} L(x^K(\Theta), t) \]

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

Similar to a Neural Network with \( K \) layers

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

**Back-propagation:**

\[
\frac{\partial L(\{x^K_l, t_l\})}{\partial \theta} = \sum_l \frac{\partial L(x^K_l, t_l)}{\partial \theta} = - \sum_l \sum_{k=1}^K \frac{\partial L}{\partial x^K_l} \frac{\partial g(x^{k-1}_l)}{\partial \theta}
\]

\[
\frac{\partial L(x^K, t)}{\partial x^t} = \frac{\partial L}{\partial x^K} \prod_{k=t}^{K-1} \frac{\partial x^{k+1}}{\partial x^k}
\]

\[
\frac{\partial x^{k+1}}{\partial x^k} \quad \text{and} \quad \frac{\partial g(x^k)}{\partial \theta}
\]
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?

- Compressive sensing: A dominant approach in fast MRI reconstruction

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

- \(g\): regularization term, e.g., \(l_0, l_1\) norm

- \(\lambda_l\): 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:

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

[Y Yang, J Sun, et al., NIPS 2016]
Deep ADMM-Net for Compressive Sensing

Data Flow Graph (DFG) for ADMM

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

Unfolding to stage n in DFG
Deep ADMM-Net for Compressive Sensing

- Deep ADMM-Net:

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

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

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

Parameter optimization: L-BFGS
Deep ADMM-Net for Compressive Sensing

- **Training Data Generation**

- **Training loss**

\[
L(\theta) = \sum_{i=1}^{m} \frac{\sqrt{\|\hat{x}_i - x_i^{gt}\|^2}}{\sqrt{\|x_i^{gt}\|^2}}
\]
Deep ADMM-Net for Compressive Sensing

<table>
<thead>
<tr>
<th>Method</th>
<th>20% NMSE</th>
<th>20% PSNR</th>
<th>30% NMSE</th>
<th>30% PSNR</th>
<th>40% NMSE</th>
<th>40% PSNR</th>
<th>50% NMSE</th>
<th>50% PSNR</th>
<th>Test time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero-filling</td>
<td>0.1700</td>
<td>29.96</td>
<td>0.1247</td>
<td>32.59</td>
<td>0.0968</td>
<td>34.76</td>
<td>0.0770</td>
<td>36.73</td>
<td>0.0013s</td>
</tr>
<tr>
<td>TV [2]</td>
<td>0.0929</td>
<td>35.20</td>
<td>0.0673</td>
<td>37.99</td>
<td>0.0534</td>
<td>40.00</td>
<td>0.0440</td>
<td>41.69</td>
<td>0.7391s</td>
</tr>
<tr>
<td>RecPF [4]</td>
<td>0.0917</td>
<td>35.32</td>
<td>0.0668</td>
<td>38.06</td>
<td>0.0533</td>
<td>40.03</td>
<td>0.0440</td>
<td>41.71</td>
<td>0.3105s</td>
</tr>
<tr>
<td>SIDWT</td>
<td>0.0885</td>
<td>35.66</td>
<td>0.0620</td>
<td>38.72</td>
<td>0.0484</td>
<td>40.88</td>
<td>0.0393</td>
<td>42.67</td>
<td>7.8637s</td>
</tr>
<tr>
<td>PBDW [6]</td>
<td>0.0814</td>
<td>36.34</td>
<td>0.0627</td>
<td>38.64</td>
<td>0.0518</td>
<td>40.31</td>
<td>0.0437</td>
<td>41.81</td>
<td>35.3637s</td>
</tr>
<tr>
<td>PANO [10]</td>
<td>0.0800</td>
<td>36.52</td>
<td>0.0592</td>
<td>39.13</td>
<td>0.0477</td>
<td>41.01</td>
<td>0.0390</td>
<td>42.76</td>
<td>53.4776s</td>
</tr>
<tr>
<td>FDLCP [8]</td>
<td>0.0759</td>
<td>36.95</td>
<td>0.0592</td>
<td>39.13</td>
<td>0.0500</td>
<td>40.62</td>
<td>0.0428</td>
<td>42.00</td>
<td>52.2220s</td>
</tr>
<tr>
<td>BM3D-MRI [11]</td>
<td>0.0674</td>
<td>37.98</td>
<td>0.0515</td>
<td>40.33</td>
<td>0.0426</td>
<td>41.99</td>
<td>0.0359</td>
<td>43.47</td>
<td>40.9114s</td>
</tr>
<tr>
<td>Init-Net13</td>
<td>0.1394</td>
<td>31.58</td>
<td>0.1225</td>
<td>32.71</td>
<td>0.1128</td>
<td>33.44</td>
<td>0.1066</td>
<td>33.95</td>
<td>0.6914s</td>
</tr>
<tr>
<td>ADMM-Net13</td>
<td>0.0752</td>
<td>37.01</td>
<td>0.0553</td>
<td>39.70</td>
<td>0.0456</td>
<td>41.37</td>
<td>0.0395</td>
<td>42.62</td>
<td>0.6964s</td>
</tr>
<tr>
<td>ADMM-Net14</td>
<td>0.0742</td>
<td>37.13</td>
<td>0.0548</td>
<td>39.78</td>
<td>0.0448</td>
<td>41.54</td>
<td>0.0380</td>
<td>42.99</td>
<td>0.7400s</td>
</tr>
<tr>
<td>ADMM-Net15</td>
<td>0.0739</td>
<td>37.17</td>
<td>0.0544</td>
<td>39.84</td>
<td>0.0447</td>
<td>41.56</td>
<td>0.0379</td>
<td>43.00</td>
<td>0.7911s</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Method</th>
<th>TV</th>
<th>RecPF</th>
<th>PANO</th>
<th>FDLCP</th>
<th>ADMM-Net15-B</th>
<th>ADMM-Net15</th>
<th>ADMM-Net17</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMSE</td>
<td>0.1019</td>
<td>0.1017</td>
<td>0.0858</td>
<td>0.0775</td>
<td>0.0790</td>
<td>0.0775</td>
<td><strong>0.0767</strong></td>
</tr>
<tr>
<td>PSNR</td>
<td>35.49</td>
<td>35.51</td>
<td>37.01</td>
<td>37.77</td>
<td>37.68</td>
<td>37.84</td>
<td><strong>37.93</strong></td>
</tr>
</tbody>
</table>
Deep ADMM-Net for Compressive Sensing

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

\[
\min_x \left\{ \frac{1}{2} \|Ax - y\|_2^2 + \sum_{l=1}^L \lambda_l g(D_l x) \right\} \quad \text{s.t. } z = [z_1, z_2, \ldots, z_l]
\]

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

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

\[
\begin{align*}
x^{(n)} &= \arg \min_x L_\rho(x, z^{(n-1)}, \beta^{(n-1)}) = F^T (P^T P + \rho I)^{-1} [P^T y + \rho F(z^{(n-1)} - \beta^{(n-1)})] \\
z^{(n)} &= \arg \min_z L_\rho(x^{(n)}, z, \beta^{(n-1)}) = \arg \min_z \frac{\rho}{2} \|x^{(n)} + \beta^{(n-1)} - z\|_2^2 + \sum_{l=1}^L \lambda_l g(D_l z) \\
\beta^{(n)} &= \beta^{(n-1)} + \eta(x - z)
\end{align*}
\]
Deep ADMM-Net for Compressive Sensing

ADMM-Net-v2

\[
\arg \min_z \frac{\rho}{2} \| (x^{(n)} + \beta^{(n-1)}) - z \|_2^2 + \sum_{l=1}^{L} \lambda_l g(D_lz)
\]

\(X^{(1)} \rightarrow X^{n-1} \rightarrow Z^{n-1} \rightarrow X^{(n)} \rightarrow Z^{(n)} \rightarrow X^{(n+1)} \rightarrow Z^{(n+1)} \rightarrow X^{(N_c+1)} \rightarrow \hat{X}

stage n
Deep ADMM-Net for Compressive Sensing

<table>
<thead>
<tr>
<th>Method</th>
<th>20%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
<th>Test time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NMSE</td>
<td>PSNR</td>
<td>NMSE</td>
<td>PSNR</td>
<td>NMSE</td>
</tr>
<tr>
<td>Zero-filling</td>
<td>0.1700</td>
<td>29.96</td>
<td>0.1247</td>
<td>32.59</td>
<td>0.0968</td>
</tr>
<tr>
<td>TV [3]</td>
<td>0.0929</td>
<td>35.20</td>
<td>0.0673</td>
<td>37.99</td>
<td>0.0534</td>
</tr>
<tr>
<td>RecPF [4]</td>
<td>0.0917</td>
<td>35.32</td>
<td>0.0668</td>
<td>38.06</td>
<td>0.0533</td>
</tr>
<tr>
<td>SIDWT</td>
<td>0.0885</td>
<td>35.66</td>
<td>0.0620</td>
<td>38.72</td>
<td>0.0484</td>
</tr>
<tr>
<td>PRIDW [5]</td>
<td>0.0814</td>
<td>36.34</td>
<td>0.0627</td>
<td>38.64</td>
<td>0.0518</td>
</tr>
<tr>
<td>PANO [6]</td>
<td>0.0800</td>
<td>36.52</td>
<td>0.0592</td>
<td>39.13</td>
<td>0.0477</td>
</tr>
<tr>
<td>FDLCP [7]</td>
<td>0.0759</td>
<td>36.95</td>
<td>0.0592</td>
<td>39.13</td>
<td>0.0500</td>
</tr>
<tr>
<td>BM3D-MRI [8]</td>
<td>0.0674</td>
<td>37.98</td>
<td>0.0515</td>
<td>40.33</td>
<td>0.0426</td>
</tr>
<tr>
<td>Init-Net_{10}</td>
<td>0.1737</td>
<td>29.64</td>
<td>0.1299</td>
<td>32.16</td>
<td>0.1025</td>
</tr>
<tr>
<td>ADMM-Net_{10}</td>
<td>0.0620</td>
<td>38.72</td>
<td>0.0480</td>
<td>40.95</td>
<td>0.0395</td>
</tr>
</tbody>
</table>

**TABLE 6**
Performance comparisons on brain data with different sampling ratios.
Deep ADMM-Net for Compressive Sensing
Deep ADMM-Net for Compressive Sensing

Our results:

ground truth:
Applications to more general compressive imaging:

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

Fast inversion:

**Proposition 2.** Suppose we are given an \( M \times N \) matrix \( \Phi \), 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

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.

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

● Recent Progress
  – Learning proximal operators
  – Multimodal medical image synthesis
  – Learning Graph CNNs for 3D shape analysis
  – Learning to Optimize

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

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)

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

\[
F(T; \cdot) \quad F(X_1; \cdot) \quad F(X_2; \cdot)
\]

**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.

\[
\begin{align*}
    w_{i,p,q}(\Theta) &= \frac{\exp(-\|F_p(T; \Theta) - F_q(X_i; \Theta)\|^2)}{\sum_j \sum_{q \in N_p} \exp(-\|F_p(T; \Theta) - F_q(X_j; \Theta)\|^2)}, \\
    \hat{L}_p(T; \Theta) &= \sum_i \sum_{q \in N_p} w_{i,p,q}(\Theta)L_q(X_i),
\end{align*}
\]

- \( L(T) \) is the estimated label.

[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 \mathbb{R}_{nl} = \{(u,v) \in \mathbb{Z}^2 | -t \leq u,v \leq t\}.

- **Distance Layer**: 
  \[
  D^\alpha_p(T, X_i) = \left\| [C(F(T))]_p - [C(S^\alpha(F(X_i)))_p \right\|^2.
  \]

- **Weight Layer**: 
  \[
  w_{i,p,q} = w^\alpha_p(X_i) = \frac{\exp(-D^\alpha_p(T, X_i))}{\sum_j \sum_{\alpha \in \mathbb{R}_{nl}} \exp(-D^\alpha_p(T, X_j))}.
  \]

- **Voting Layer**: 
  \[
  \hat{L}_p(T) = \sum_i \sum_{\alpha \in \mathbb{R}_{nl}} w^\alpha_p(X_i) \left[C \left(S^\alpha(L(X_i))\right)\right]_p.
  \]

- **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: \( d(T, X_i) = \|F(T) - F(X_i)\| \)

**Database:** MICCAI 2013 SATA Segmentation Challenge
Deep Fusion Net for MR Image Segmentation

- Atlas selection

![Bar chart showing the averaged Dice metric for different numbers of selected atlases. The chart compares two methods: DFN and DFN NMI.](image)
● Segmentation accuracy

![Images of segmentation results](image)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>0.653</td>
<td>0.683</td>
<td>0.754</td>
<td>0.726</td>
<td>0.681</td>
<td>0.803</td>
</tr>
</tbody>
</table>

Table 1. The mean Dice metrics of different methods.

**MICCAI 2013 SATA Dataset**


Deep Fusion Net for MR Image Segmentation

- **Examples of results**

<table>
<thead>
<tr>
<th>Slice 1</th>
<th>Target</th>
<th>Ground-truth</th>
<th>Output</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slice 2</td>
<td>Target</td>
<td>Ground-truth</td>
<td>Output</td>
<td>Segment</td>
</tr>
<tr>
<td>Slice 3</td>
<td>Target</td>
<td>Ground-truth</td>
<td>Output</td>
<td>Segment</td>
</tr>
<tr>
<td>Slice 4</td>
<td>Target</td>
<td>Ground-truth</td>
<td>Output</td>
<td>Segment</td>
</tr>
<tr>
<td>Slice 5</td>
<td>Target</td>
<td>Ground-truth</td>
<td>Output</td>
<td>Segment</td>
</tr>
<tr>
<td>Slice 6</td>
<td>Target</td>
<td>Ground-truth</td>
<td>Output</td>
<td>Segment</td>
</tr>
<tr>
<td>Slice 7</td>
<td>Target</td>
<td>Ground-truth</td>
<td>Output</td>
<td>Segment</td>
</tr>
<tr>
<td>Slice 8</td>
<td>Target</td>
<td>Ground-truth</td>
<td>Output</td>
<td>Segment</td>
</tr>
<tr>
<td>Slice 9</td>
<td>Target</td>
<td>Ground-truth</td>
<td>Output</td>
<td>Segment</td>
</tr>
<tr>
<td>Slice 10</td>
<td>Target</td>
<td>Ground-truth</td>
<td>Output</td>
<td>Segment</td>
</tr>
</tbody>
</table>
Deep Fusion Net for MR Image Segmentation

### 2009 LV segmentation challenge

ADM: averaged Dice Metric; AJM: averaged Jaccard Metric

<table>
<thead>
<tr>
<th>Method</th>
<th>Epicardium ADM</th>
<th>Epicardium AJM</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFN</td>
<td><strong>0.9453(0.0228)</strong></td>
<td><strong>0.8972(0.0399)</strong></td>
</tr>
<tr>
<td>DLLS</td>
<td>0.94(0.02)</td>
<td>–</td>
</tr>
<tr>
<td>DLDM</td>
<td>0.94(0.02)</td>
<td>–</td>
</tr>
<tr>
<td>DLDM_init</td>
<td>0.89(0.03)</td>
<td>–</td>
</tr>
<tr>
<td>SVMAF</td>
<td>0.9259(0.0251)</td>
<td>0.8630(0.0419)</td>
</tr>
<tr>
<td>PB</td>
<td>0.9170(0.0318)</td>
<td>0.8483(0.0523)</td>
</tr>
<tr>
<td>MV</td>
<td>0.9155(0.0326)</td>
<td>0.8458(0.0535)</td>
</tr>
</tbody>
</table>

### 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 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)

Background

MR (excellent soft-tissue contrast)

CT (provide tissue electron densities)

(Paired training data)
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:

\[
F_x^{(\alpha)}(I) = \frac{1}{Z} \exp\left(-\frac{D_P(I, x, x + \alpha)}{V(I, x)}\right)
\]

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

[H. R. Yang, J. Sun, et al., MICCAI-DLMIA, 2018]
Multi-modal Medical Image Synthesis

\[
\mathcal{L}(G_{CT}, G_{MR}, D_{CT}, D_{MR}) = \mathcal{L}_{GAN}(G_{CT}, D_{CT}) + \mathcal{L}_{GAN}(G_{MR}, D_{MR}) + \lambda_1 \mathcal{L}_{cycle}(G_{CT}, G_{MR}) - \lambda_2 \mathcal{L}_{structure}(G_{CT}, G_{MR})
\]

Training Loss

CT domain

MR domain
Multi-modal Medical Image Synthesis

**Compared methods**

- “cycleGAN”: Conventional cycleGAN
- “cycleGAN (paired)”: CycleGAN trained with paired data

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

<table>
<thead>
<tr>
<th></th>
<th>MAE</th>
<th>PSNR</th>
<th>SSIM</th>
<th>SSIM (HG)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CycleGAN (unpaired)</td>
<td>150.28 (18.06)</td>
<td>23.09 (1.04)</td>
<td>0.732 (0.029)</td>
<td>0.546 (0.042)</td>
</tr>
<tr>
<td>CycleGAN (paired)</td>
<td>122.66* (16.71)</td>
<td>24.57* (1.25)</td>
<td>0.785* (0.029)</td>
<td>0.630* (0.043)</td>
</tr>
<tr>
<td>Proposed</td>
<td>127.78* (16.21)</td>
<td>24.67* (1.27)</td>
<td>0.780* (0.026)</td>
<td>0.622* (0.044)</td>
</tr>
</tbody>
</table>

* denotes $p < 0.001$ compared to the conventional cycleGAN using a paired sample.
Learning proximal operators

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

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

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

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

\[
U_n = \text{prox}_{\frac{1}{b_n} g} \left( \hat{U}_n \right),
\]

\[
T_n = \text{prox}_{\frac{1}{c_n} f} \left( \hat{T}_n \right),
\]

\[
\hat{Q}_n = \frac{\alpha(\hat{P} + \hat{T}_n - 1) \circ \hat{T}_n + \gamma D^T \hat{U}_n}{\alpha \hat{T}_n \circ \hat{T}_n + \gamma \text{diag}(D^T D)}.
\]
Learning proximal operators

(a) Multi-stage network for single image dehazing

(b) Network structure for $n$-th stage

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

Graph representation:

Graph \rightarrow Matrix

Hyper-graph \rightarrow Tensor

Shape

Data graph
Learning on 3D shapes

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, etc.
  - Learning-based approach
    Learn the optimizer by Recurrent Neural Network

\[ g_t = m(h_t, \nabla_t) \]

RNN: black-box

Hyper-Adam: 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

Hyper-Adam Algorithm

Algorithm 2 Task-Adaptive HyperAdam

Require:
1: Initialized parameter $w_0$, step size $\alpha$, batch size $N_B$.
2: Dataset $\{(x_i, y_i)\}_{i=1}^N$.

Initialize:
3: $m_0, v_0, \hat{\beta}_0, \hat{\gamma}_0, s_0 = 0 \in \mathbb{R}^{p \times J}$, $l \in \mathbb{R}^{p \times J}$, $\varepsilon = 1 \times 10^{-4}$.
4: for all $t = 1, \ldots, T$ do
5: Draw random batch $\{(x_{ik}, y_{ik})\}_{k=1}^{N_B}$ from dataset
6: $g_t = \sum_{k=1}^{N_B} \nabla l(x_{ik}, y_{ik}, w_{t-1})$\n7: $G_t = [g_t, \ldots, g_t]$ \hfill $\triangleright G_t \in \mathbb{R}^{p \times J}$
8: $s_t = F_h(s_{t-1}, g_t; \Theta_h)$ \hfill $\triangleright$ current state
9: $\beta_t \triangleq [\beta^1_t, \ldots, \beta^J_t] = F_u(s_t, m_{t-1}; \Theta_u)$
10: $\gamma_t \triangleq [\gamma^1_t, \ldots, \gamma^J_t] = F_r(s_t, m_{t-1}; \Theta_r)$
11: $m_t = \beta_t \odot m_{t-1} + (1 - \beta_t) \odot G_t$
12: $v_t = \gamma_t \odot v_{t-1} + (1 - \gamma_t) \odot G_t^2$
13: $\hat{\beta}_t = \beta_t \odot \hat{\beta}_{t-1} + (1 - \beta_t) \odot 1$
14: $\hat{\gamma}_t = \gamma_t \odot \hat{\gamma}_{t-1} + (1 - \gamma_t) \odot 1$
15: $\tilde{m}_t = m_t / \hat{\beta}_t$, $\tilde{v}_t = v_t / \hat{\gamma}_t$, \hfill $\triangleright$ correcting bias
16: $\tilde{m}_t \triangleq [\tilde{m}^1_t, \ldots, \tilde{m}^J_t] = \frac{\tilde{m}_t}{\sqrt{\tilde{v}_t} + \varepsilon}$ \hfill $\triangleright$ moment field
17: $\rho_t \triangleq [\rho^1_t, \ldots, \rho^J_t] = F_q(s_t; \Theta_q)$ \hfill $\triangleright$ weight field
18: $d_t = \sum_{j=1}^J \rho^j_t \odot \tilde{m}^j_t$
19: $w_t = w_{t-1} - \alpha d_t$
20: end for
21: return final parameter $w_T$. 

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 $S_t = [s_t^1, ..., s_t^J]$

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

(a) 9-hidden-layer MLP
(b) 2-layer LSTM

(c) CNN-2, MNIST, 2000 steps
(d) CNN-2, MNIST, 10000 steps
(g) CNN-2, CIFAR-10, 2000 steps
(h) CNN-2, CIFAR-10, 10000 steps

Legend:
- DMoptimizer
- HyperAdam
- AdaDelta
- AdaGrad
- Adam
- Momentum
- RMSProp
- RNNprop
- SGD
Learning to optimize

Generalization with fixed steps

(a) Basic MLP with ELU, MNIST
(b) 7-hidden-layer MLP, MNIST
(c) LSTM, synthetic data
(d) MLPs with varying layers

<table>
<thead>
<tr>
<th>Activation</th>
<th>Adam</th>
<th>DMoptimizer</th>
<th>RNNprop</th>
<th>HyperAdam</th>
</tr>
</thead>
<tbody>
<tr>
<td>sigmoid</td>
<td>0.35</td>
<td>0.38</td>
<td>0.34</td>
<td>0.33</td>
</tr>
<tr>
<td>ReLU</td>
<td>0.32</td>
<td>1.42</td>
<td>0.31</td>
<td>0.29</td>
</tr>
<tr>
<td>ELU</td>
<td>0.31</td>
<td>2.02</td>
<td>0.31</td>
<td>0.28</td>
</tr>
<tr>
<td>tanh</td>
<td>0.34</td>
<td>0.83</td>
<td>0.33</td>
<td>0.36</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Task</th>
<th>Adam</th>
<th>DMoptimizer</th>
<th>RNNprop</th>
<th>HyperAdam</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>0.65</td>
<td>3.10</td>
<td>0.49</td>
<td>0.42</td>
</tr>
<tr>
<td>Small noise</td>
<td>0.39</td>
<td>3.06</td>
<td>0.32</td>
<td>0.19</td>
</tr>
<tr>
<td>2-layer</td>
<td>0.51</td>
<td>2.05</td>
<td>0.27</td>
<td>0.26</td>
</tr>
</tbody>
</table>

Table 2: Performance on different sequence prediction tasks.
Generalization of the Learners

<table>
<thead>
<tr>
<th>Task</th>
<th>Measure</th>
<th>Adam</th>
<th>DMoptimizer</th>
<th>RNNprop</th>
<th>HyperAdam</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN-1</td>
<td>loss</td>
<td>0.10</td>
<td>2.30</td>
<td>0.36</td>
<td>0.05</td>
</tr>
<tr>
<td>(MNIST)</td>
<td>top-1</td>
<td>98.50%</td>
<td>10.10%</td>
<td>96.46%</td>
<td>98.48%</td>
</tr>
<tr>
<td></td>
<td>top-2</td>
<td>99.59%</td>
<td>20.38%</td>
<td>99.03%</td>
<td>99.63%</td>
</tr>
<tr>
<td>CNN-2</td>
<td>loss</td>
<td>0.09</td>
<td>2.30</td>
<td>2.30</td>
<td>0.07</td>
</tr>
<tr>
<td>(MNIST)</td>
<td>top-1</td>
<td>98.98%</td>
<td>11.35%</td>
<td>11.37%</td>
<td>99.02%</td>
</tr>
<tr>
<td></td>
<td>top-2</td>
<td>99.80%</td>
<td>21.45%</td>
<td>21.69%</td>
<td>99.78%</td>
</tr>
</tbody>
</table>

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 $\rightarrow$ deep CNN
  - ADMM algorithm $\rightarrow$ deep ADMM-net
  - Non-local approach $\rightarrow$ 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!