Invertible Neural Networks and their Applications

Jun-Jie Huang and Pier Luigi Dragotti
Outline

1. Overview of Invertible Neural Networks
   - Origin of INN and Normalizing flows
   - INN for Inverse Problems

2. Wavelet-Inspired Invertible Neural Network

3. INN and diffusion models: INDigo

4. Other applications of INN
1. Overview of Invertible Neural Networks

Invertible Neural Networks (INNs) are bijective function approximators which have a forward mapping

\[ F_{\theta} : \mathbb{R}^d \rightarrow \mathbb{R}^l \]

\[ x \mapsto z \]

and inverse mapping

\[ F_{\theta}^{-1} : \mathbb{R}^l \rightarrow \mathbb{R}^d \]

\[ z \mapsto x \]
1. Overview of Invertible Neural Networks

How to Achieve Invertibility?

Invertible via lifting scheme like architectures

- Signal splitting
- Alternative prediction and update

Forward pass

\[
\begin{align*}
\text{Split} & \quad \begin{cases}
    d = x_o - P(x_e) \\
    s = x_e + U(d)
\end{cases}
\end{align*}
\]

Backward pass

\[
\begin{align*}
\begin{cases}
    x_o = d + P(x_e) \\
    x_e = s - U(d)
\end{cases}
\end{align*}
\]

Factoring wavelet transforms into lifting steps
I Daubechies, W Sweldens
Journal of Fourier analysis and applications 4 (3), 245-267
1. Overview of Invertible Neural Networks

Invertible Neural Networks are bijective function approximators which have a forward mapping

\[ F_\theta : \mathbb{R}^d \rightarrow \mathbb{R}^l \]

\[ x \mapsto z \]

and inverse mapping

\[ F_\theta^{-1} : \mathbb{R}^l \rightarrow \mathbb{R}^d \]

\[ z \mapsto x \]
1. Overview of Invertible Neural Networks

Also known as Normalizing Flow for generative modeling

- Tractable Jacobian, allows explicit computation of posterior probability

Inverse problems involve reconstructing unknown physical quantities from indirect measurements:

- denoising
- super-resolution
- deblurring
- inpainting
- …
1. Overview of Invertible Neural Networks

Invertible Neural Networks are ideal architectures to address inverse problems.

Outline

1. Overview of Invertible Neural Networks
   - Origin of INN and Normalizing flows
   - INN for Inverse Problems

2. Wavelet-Inspired Invertible Neural Network

3. INN and diffusion models: INDigo

4. Other applications of INN
2. Wavelet-inspired Invertible Neural Network

Image Denoising

- Recover a clean image from noisy observations
- Raw image data is usually noisy

Denoising is the “simplest” inverse problem yet plays an important role in many applications.

\[ y = x + e \]

- Measured
- Clean
- Noise
Deep Learning methods are effective while less interpretable and controllable.

DnCNN

U-Net

RED-Net
Wavelet Thresholding is a widely used denoising approach

- **Wavelets** provide invertible sparse representations of piecewise smooth images

Universal threshold

\[ T = \sqrt{2\sigma^2 \log N} \]

BayesShrink threshold

\[ T = \hat{\sigma}^2 / \hat{\sigma}_X \]
Motivation:

• Whether it is possible to **combine the merits of Wavelet Thresholding and DNNs** for image denoising and other image restoration tasks?

Idea:

• Learning a redundant transform with perfect reconstruction property using a **Wavelet-inspired INvertible Network (WINNet)**
2. Wavelet-inspired Invertible Neural Network

Overall architecture

Model-inspired Noise Est. Network

Noisy image

Denoised image

\( \sigma^2 \)

LINN\(_1\) \rightarrow \ldots \rightarrow \text{high-freq} \rightarrow \text{low-freq} \rightarrow \ldots \rightarrow \text{LINN}_k

Sparsity-driven Denoising Network

(forward)

(backward)

Sparsity-driven Denoising Network

(backward)
2. Wavelet-inspired Invertible Neural Network Network (LINN)

- **Forward pass**
- **Backward pass**

When no operation is applied on the representation, perfect reconstruction can be achieved using the backward pass.
2. Wavelet-inspired Invertible Neural Network

Lifting inspired Invertible Neural Network (LINN)

- Predictor/Updater networks
  - Separable convolutional networks with soft-thresholding non-linearity
  - Noise adaptive soft-threshold

\[ S_{\sigma \lambda}(x) \]

Noise adaptive soft-threshold
2. Wavelet-inspired Invertible Neural Network

Sparsity-driven Denoising Network

- Non-invertible component
- A well-understood denoising network can lead to enhanced interpretability
2. Wavelet-inspired Invertible Neural Network

Sparsity-driven Denoising Network

- We model the denoising process as Convolutional Sparse Coding

\[
g = \arg\min_g \frac{1}{2} \left\| z_d^{(l)} - \sum_{i=1}^{M} D_i \otimes g_i \right\|_2^2 + \sum_{i=1}^{M} \lambda_i \|g_i\|_1
\]

- Unfold it into CLISTA Network \( G_t = \mathcal{T}_{\lambda_t} \left( G_{t-1} + W_a \otimes (D_M^k - W_s \otimes G_{t-1}) \right) \)
2. Wavelet-inspired Invertible Neural Network

Model-inspired Noise Estimation Network

Model-inspired Noise Estimation Network

Visualization of the selected patches for noise level estimation
2. Wavelet-inspired Invertible Neural Network

Experimental Settings:

- Training loss:
  - Mean squared error between restored image and clean image:
    \[ \mathcal{L}_r = \frac{1}{2N} \sum_{i=1}^{N} \| X_i - \hat{X}_i \|^2 \]
  - Spectral norm loss for LINN:
    \[ \mathcal{L}_s = \frac{1}{K \cdot M \cdot J} \sum_{k=1}^{K} \sum_{m=1}^{M} \sum_{j=1}^{J} \| P_{m,j}^k \|_s + \| U_{m,j}^k \|_s \]
  - Orthogonal loss for CLISTA Network:
    \[ \mathcal{L}_o = \| W_s \otimes W_a - \delta \|_F^2 \]

- Optimizer:
  - Adam with learning rate \( 1 \times 10^{-3} \) which decays to \( 1 \times 10^{-4} \) at the 30-th epoch

- Training data:
  - 400 images of size 180×180
Experimental Results — Non-blind denoising

Comparison of average PSNR (dB) and number of parameters of different methods. The testing dataset is Set12 with noise level $\sigma = [15, 25, 50]$. 

22
## 2. Wavelet-inspired Invertible Neural Network

### Experimental Results — Blind denoising

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Methods</th>
<th>(\sigma = 5)</th>
<th>(\sigma = 25)</th>
<th>(\sigma = 45)</th>
<th>(\sigma = 65)</th>
<th>(\sigma = 85)</th>
<th>(\sigma = 105)</th>
<th>(\sigma = 125)</th>
<th>(\sigma = 145)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BUIFD [29]</td>
<td>37.41</td>
<td>28.76</td>
<td>25.61</td>
<td>23.07</td>
<td>18.81</td>
<td>15.98</td>
<td>14.45</td>
<td>13.52</td>
</tr>
<tr>
<td></td>
<td>BF-CNN [28]</td>
<td>37.73</td>
<td>29.11</td>
<td>26.58</td>
<td>25.12</td>
<td>24.10</td>
<td>23.33</td>
<td>22.70</td>
<td>22.18</td>
</tr>
<tr>
<td></td>
<td>WINNet (1-scale)</td>
<td>37.82</td>
<td>29.13</td>
<td>26.66</td>
<td>25.23</td>
<td>24.23</td>
<td>23.46</td>
<td>22.81</td>
<td>22.23</td>
</tr>
<tr>
<td>Set12</td>
<td>DnCNN-B [21]</td>
<td>37.88</td>
<td>30.38</td>
<td>27.68</td>
<td>23.52</td>
<td>15.95</td>
<td>13.18</td>
<td>11.78</td>
<td>10.92</td>
</tr>
<tr>
<td></td>
<td>BUIFD [29]</td>
<td>37.34</td>
<td>30.18</td>
<td>27.01</td>
<td>24.27</td>
<td>19.41</td>
<td>16.28</td>
<td>14.66</td>
<td>13.73</td>
</tr>
<tr>
<td></td>
<td>BF-CNN [28]</td>
<td>37.81</td>
<td>30.33</td>
<td>27.58</td>
<td>25.83</td>
<td>24.54</td>
<td>23.55</td>
<td>22.74</td>
<td>22.07</td>
</tr>
<tr>
<td></td>
<td>WINNet (1-scale)</td>
<td>38.22</td>
<td>30.33</td>
<td>27.72</td>
<td>26.03</td>
<td>24.77</td>
<td>23.76</td>
<td>22.94</td>
<td>22.24</td>
</tr>
</tbody>
</table>

### Diagram

- **Training noise levels**
  - Input Noise Level: 5, 25, 45, 65, 85, 105, 125, 145

- **Unseen noise levels**
  - Input Noise Level: 5, 25, 45, 65, 85, 105, 125, 145
2. Wavelet-inspired Invertible Neural Network

Application on Image Deblurring

\[ x = \arg \min_x \frac{1}{2\sigma^2} \| y - k \otimes x \|_2^2 + \lambda \Phi(x) \]

\[ x_k = \arg \min_x \| y - k \otimes x \|_2^2 + \frac{\lambda \sigma^2}{\beta^2} \| x - z_{k-1} \|_2^2 \]

\[ z_k = \arg \min_z \frac{1}{2\beta^2} \| z - x_k \|_2^2 + \Phi(z) \]
2. Wavelet-inspired Invertible Neural Network

Image Deblurring with WINNet

Algorithm 1: Plug-and-Play image deblurring with blind WINNet.

1. **Input:** Input image $y$, kernel $k$, parameter $\lambda$;
2. **Initialize:** $z_0 = y$, $\beta_0 = \text{NENet}(z_0)$, $\beta_1 = 10 \times \beta_0$, $k = 1$;
3. **while** $\beta_k > \beta_0$ **do**
   4. $x_k = \arg\min_x ||y - k \otimes x||_2^2 + \frac{\lambda \beta_0^2}{\beta_k^2} ||x - z_{k-1}||_2^2$;  
      %Auxiliary Update
   5. $\beta_{k+1} = \text{NENet}(x_k)$;
   6. $z_k = \text{WINNet}(x_k, 2\beta_{k+1})$;
   7. $k = k + 1$;
4. **end**
8. **Output:** Deblurred image $x = z_{k-1}$.

%Noise Estimation and Denoising
2. Wavelet-inspired Invertible Neural Network

Experimental Results on Image Deblurring
2. Wavelet-inspired Invertible Neural Network

Take home message:

• With proper nonlinear over-parameterization, Wavelet-inspired network architecture can achieve good performance, strong controllability, generalization ability and high interpretability


Outline

1. Overview of Invertible Neural Networks
   - Origin of INN and Normalizing flows
   - INN for Inverse Problems

2. Wavelet-Inspired Invertible Neural Network

3. INN and diffusion models: INDigo

4. Other applications of INN
3. INN and Diffusion Models

\[
\hat{x} = \min_x \| H(x) - y \|^2 + \lambda \rho(x)
\]

- consistency term  
- prior

- Impose consistency using the forward part of the INN
- Impose the prior using diffusion models
- Iterate
Diffusion Models are good for “unconditional” generation of new samples (e.g., Denoising Probabilistic Diffusion Models)

Motivation: Can we use a pretrained “unconditional” diffusion model for inverse problems?

3. INN and Diffusion Models

- Given a training set \( \{x_i, y_i\} \) which contains N high-quality images and their low-quality counterparts, we learn the forward part of the INN using the following loss:

\[
L(\Theta) = \frac{1}{N} \sum_{i=1}^{N} \|c^i - y^i\|_2^2,
\]

- Consequently, \( d \) models the lost details that need to be recovered with the diffusion model
3. INN and Diffusion Models
3. INN and Diffusion Models

Algorithm 1 INDigo

1: \(x_T \sim \mathcal{N}(0, I)\)
2: for \(t = T, \ldots, 1\) do
3: \(z \sim \mathcal{N}(0, I)\) if \(t > 1\), else \(z = 0\)
4: \(x_{0,t} = \frac{1}{\sqrt{\alpha_t}}(x_t - \sqrt{1 - \bar{\alpha}_t}\epsilon_\theta(x_t, t))\)
5: \(\tilde{x}_{t-1} = \frac{\sqrt{\alpha_t(1-\bar{\alpha}_t-1)}}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_t-1}}{1-\bar{\alpha}_t}x_{0,t} + \sigma_t z\)
6: \(c_t, d_t = f_\phi(x_{0,t})\)
7: \(\hat{x}_{0,t} = f_\phi^{-1}(y, d_t)\)
8: \(x_{t-1} = \hat{x}_{t-1} - \zeta \nabla_x \|\hat{x}_{0,t} - x_{0,t}\|^2\)
9: end for
10: return \(x_0\)
● This approach is simple, flexible and effective

● No-need to know the degradation process

● The degradation process can be highly non-linear

● No need to retrain the diffusion model for every new degradation (just need to train the INN)
3. INN and Diffusion Models

Results for non-linear degradation models

![Image of results for non-linear degradation models]

- Input
- Bicubic
- Ours
- Ground Truth

Table 1: Resultson 4x Super-Resolution

<table>
<thead>
<tr>
<th>Method</th>
<th>Noise</th>
<th>PSNR</th>
<th>LPIPS</th>
<th>FID</th>
<th>NIQE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>0.10</td>
<td>26.00</td>
<td>0.14</td>
<td>3.90</td>
<td>0.73</td>
</tr>
<tr>
<td></td>
<td>0.05</td>
<td>26.25</td>
<td>0.13</td>
<td>3.96</td>
<td>0.73</td>
</tr>
<tr>
<td></td>
<td>0.00</td>
<td>28.15</td>
<td>0.09</td>
<td>4.15</td>
<td>0.83</td>
</tr>
<tr>
<td>DDRM</td>
<td>0.10</td>
<td>22.45</td>
<td>0.19</td>
<td>4.96</td>
<td>1.23</td>
</tr>
<tr>
<td></td>
<td>0.05</td>
<td>26.25</td>
<td>0.13</td>
<td>3.96</td>
<td>0.73</td>
</tr>
<tr>
<td></td>
<td>0.00</td>
<td>27.43</td>
<td>0.08</td>
<td>5.47</td>
<td>1.90</td>
</tr>
<tr>
<td>ILVR</td>
<td>0.10</td>
<td>24.60</td>
<td>0.48</td>
<td>4.49</td>
<td>1.70</td>
</tr>
<tr>
<td></td>
<td>0.05</td>
<td>26.42</td>
<td>0.30</td>
<td>4.65</td>
<td>1.70</td>
</tr>
<tr>
<td></td>
<td>0.00</td>
<td>27.43</td>
<td>0.21</td>
<td>5.47</td>
<td>1.90</td>
</tr>
<tr>
<td>DPS</td>
<td>0.10</td>
<td>24.73</td>
<td>0.17</td>
<td>4.24</td>
<td>1.56</td>
</tr>
<tr>
<td></td>
<td>0.05</td>
<td>25.92</td>
<td>0.15</td>
<td>4.37</td>
<td>1.70</td>
</tr>
<tr>
<td></td>
<td>0.00</td>
<td>26.67</td>
<td>0.14</td>
<td>4.49</td>
<td>1.90</td>
</tr>
</tbody>
</table>
3. INN and Diffusion Models

Results on 4x super-resolution

<table>
<thead>
<tr>
<th>Method</th>
<th>Noise $\sigma$</th>
<th>PSNR ↑</th>
<th>FID ↓</th>
<th>LPIPS ↓</th>
<th>NIQE ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILVR</td>
<td>0</td>
<td>27.43</td>
<td>44.04</td>
<td>0.2123</td>
<td>5.4689</td>
</tr>
<tr>
<td>DDRM</td>
<td>0</td>
<td>28.08</td>
<td>65.80</td>
<td>0.1722</td>
<td>4.4694</td>
</tr>
<tr>
<td>DPS</td>
<td>0</td>
<td>26.67</td>
<td>32.44</td>
<td>0.1370</td>
<td>4.4890</td>
</tr>
<tr>
<td>Ours</td>
<td>0</td>
<td><strong>28.15</strong></td>
<td><strong>22.33</strong></td>
<td><strong>0.0889</strong></td>
<td><strong>4.1564</strong></td>
</tr>
<tr>
<td>ILVR</td>
<td>0.05</td>
<td>26.42</td>
<td>60.27</td>
<td>0.3045</td>
<td>4.6527</td>
</tr>
<tr>
<td>DDRM</td>
<td>0.05</td>
<td>27.06</td>
<td>45.90</td>
<td>0.2028</td>
<td>4.8238</td>
</tr>
<tr>
<td>DPS</td>
<td>0.05</td>
<td>25.92</td>
<td>31.71</td>
<td>0.1475</td>
<td>4.3743</td>
</tr>
<tr>
<td>Ours</td>
<td>0.05</td>
<td><strong>27.16</strong></td>
<td><strong>26.64</strong></td>
<td><strong>0.1215</strong></td>
<td><strong>4.1004</strong></td>
</tr>
<tr>
<td>ILVR</td>
<td>0.10</td>
<td>24.60</td>
<td>88.88</td>
<td>0.4833</td>
<td>4.4888</td>
</tr>
<tr>
<td>DDRM</td>
<td>0.10</td>
<td>26.16</td>
<td>45.49</td>
<td>0.2273</td>
<td>4.9644</td>
</tr>
<tr>
<td>DPS</td>
<td>0.10</td>
<td>24.73</td>
<td>31.66</td>
<td>0.1698</td>
<td>4.2388</td>
</tr>
<tr>
<td>Ours</td>
<td>0.10</td>
<td><strong>26.25</strong></td>
<td><strong>28.89</strong></td>
<td><strong>0.1399</strong></td>
<td><strong>3.9659</strong></td>
</tr>
</tbody>
</table>
3. INN and Diffusion Models

Results on blind unsupervised deconvolution

![Results on blind unsupervised deconvolution](image)

1. Overview of Invertible Neural Networks
   - Origin of INN and Normalizing flows
   - INN for Inverse Problems

2. Wavelet-Inspired Invertible Neural Network

3. INN and diffusion models: INDigo

4. Other applications of INN
4. Other Applications of INN: Blind Source Separation

Deep Unfolded Reflection Removal Network

- Overparameterize the wavelet transform as a learnable INN

\[
\min_{z_T,z_R} \frac{1}{2} \left\| I - \sum_{i=1}^{N} D_T^i \otimes z_T^i - \sum_{i=1}^{N} D_R^i \otimes z_R^i \right\|_F^2 + \lambda_{TP}(z_T) \\
+ \lambda_{RP}(z_R) + \kappa \mathcal{E} \left( \sum_{i=1}^{N} D_T^i \otimes z_T^i, \sum_{i=1}^{N} D_R^i \otimes z_R^i \right)
\]

Exclusion Prior:
\[ \mathcal{E}(T, R) = \sum_{m=1}^{M} \left\| (W_m \otimes T) \odot (W_m \otimes R) \right\|_1 \]
where \( W \) denotes wavelet transform.

4. Other Applications of INN: Blind Source Separation

Deep Unfolded Reflection Removal Network

- Subjective comparisons

PSNR v.s. FLOPS and #Params

- Objective comparisons:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Real20 (20)</td>
<td>PSNR (↑)</td>
<td>18.45</td>
<td>22.55</td>
<td>18.41</td>
<td>21.86</td>
<td>21.57</td>
<td>22.89</td>
<td>23.26</td>
<td>23.80</td>
</tr>
<tr>
<td></td>
<td>SSIM (↑)</td>
<td>0.690</td>
<td>0.788</td>
<td>0.726</td>
<td>0.762</td>
<td>0.807</td>
<td>0.803</td>
<td>0.806</td>
<td>0.814</td>
</tr>
<tr>
<td>Nature (20)</td>
<td>PSNR (↑)</td>
<td>19.33</td>
<td>19.56</td>
<td>18.92</td>
<td>23.57</td>
<td>21.84</td>
<td>20.60</td>
<td>23.85</td>
<td>24.24</td>
</tr>
<tr>
<td></td>
<td>SSIM (↑)</td>
<td>0.745</td>
<td>0.736</td>
<td>0.737</td>
<td>0.783</td>
<td>0.805</td>
<td>0.755</td>
<td>0.810</td>
<td>0.812</td>
</tr>
</tbody>
</table>
4. Other Applications of INN: Adversarial Attacks

Adversarial Attack via Invertible Neural Networks:

\[ \ell_{adv} \]

\[ x_{adv} \]

\[ x_{tgt} \]

\[ x_{ctn} \]

Invertible Neural Networks: 

\[ x \]

\[ \ell \]

\[ x_{r} \]

\[ x_{adv} \]

\[ x_{ctn} \]

\[ x_{tgt} \]

Target Image Learning Module

Information Exchange Module

Visualization & Interpretation

\[ x_{clc} \]

\[ x_{tgt} \]

\[ x_{adv} \]

\[ x_{r} \]

\[ x_{drop} \]

\[ |x_{clc} - x_{adv}| \]

\[ \ell_{dev} \]

\[ \ell_{adv} \]

Table 1: Accuracy and evaluation metrics on different methods. All methods use \( \epsilon = 8/255 \) as the adversarial budget. ASR denotes the accuracy of adversarial attacks. 1 means the value is higher the better, and vice versa. (The best and the second best result in each column is in bold and underline.)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Methods</th>
<th>( \ell_2 )</th>
<th>( \ell_{\infty} )</th>
<th>SSIM</th>
<th>LPIPS</th>
<th>FID</th>
<th>ASR(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ImageNet-IK</td>
<td>StepLL</td>
<td>26.90</td>
<td>0.04</td>
<td>0.948</td>
<td>0.1443</td>
<td>25.176</td>
<td>98.5</td>
</tr>
<tr>
<td></td>
<td>C&amp;W</td>
<td>10.33</td>
<td>0.07</td>
<td>0.977</td>
<td>0.0617</td>
<td>11.515</td>
<td>91.7</td>
</tr>
<tr>
<td></td>
<td>PGD</td>
<td>64.42</td>
<td>0.04</td>
<td>0.881</td>
<td>0.2155</td>
<td>35.012</td>
<td>90.2</td>
</tr>
<tr>
<td></td>
<td>PerC-AL</td>
<td>1.85</td>
<td>0.10</td>
<td>0.995</td>
<td>0.0239</td>
<td>5.116</td>
<td>100.0</td>
</tr>
<tr>
<td></td>
<td>AdvDrop</td>
<td>18.47</td>
<td>0.07</td>
<td>0.977</td>
<td>0.0639</td>
<td>6.687</td>
<td>100.0</td>
</tr>
<tr>
<td></td>
<td>SSAD</td>
<td>5.97</td>
<td>0.03</td>
<td>0.991</td>
<td>0.0352</td>
<td>5.221</td>
<td>99.8</td>
</tr>
<tr>
<td></td>
<td>AdvINN-HCT</td>
<td>5.73</td>
<td>0.03</td>
<td>0.991</td>
<td>0.0206</td>
<td>5.361</td>
<td>100.0</td>
</tr>
<tr>
<td></td>
<td>AdvINN-UAP</td>
<td>5.84</td>
<td>0.03</td>
<td>0.990</td>
<td>0.0212</td>
<td>5.900</td>
<td>100.0</td>
</tr>
<tr>
<td></td>
<td>AdvINN-CGT</td>
<td>2.66</td>
<td>0.03</td>
<td>0.996</td>
<td>0.0118</td>
<td>1.594</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Less perceptible adversarial examples with 100% attacking success rate!

Conclusions

- The perfect reconstruction property of the Invertible Neural Networks is intriguing
- Designing INN using wavelets/lifting leads to more interpretable and simpler architectures
- Good generalization ability
- Invertible neural networks have the potential for many image/signal processing applications
Q&A

Thanks for listening!
Related Publications


