Image Restoration with Deep Generative Models

ILLINOIS


Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign

ICASSP 2018
Overview

Image restoration refers to the task of recovering an image from a corrupted sample.
Image restoration refers to the task of recovering an image from a corrupted sample.
Examples:
- Inpainting
- Denoising
- etc.
Image restoration refers to the task of recovering an image from a corrupted sample.

Examples:
- Inpainting
- Denoising
- etc.

Task is generally ill-posed.
Task:
Let $y$ denote the observed image, $x^*$ be the original unobserved image, $A$ a known generative operator $A$, and noise $\epsilon$.

$$y = A(x^*) + \epsilon,$$

We seek to recover $\hat{x}$ with an objective of the form

$$\hat{x} = \arg\min_x d(y, A(x)) + \lambda R(x)$$

Where $R(\cdot)$ is some prior, and $d(\cdot)$ is some distance metric (e.g. $p$-norm).
Traditional Approach:

- Hand designed prior, $R$, (e.g. TV, Low-rank, sparsity, etc.)
- Solve the objective function with some solver
- **Disadvantage:** Priors tend to be simple, generally unable to capture complicated structures in data
Data-driven, direct:

- Train a deep network, $h(\cdot; \Theta)$ on clean and corrupted pairs in training set $\mathcal{D}$, that maps the corrupted measurements directly predict a clean version.

$$
\Theta^* = \arg\min_{\Theta} \| x_i - h(y_i; \Theta) \|_p + \lambda \| \Theta \|, \quad \forall (x_i, y_i) \in \mathcal{D}
$$

Output image:

$$\hat{x} = h(y; \Theta^*)$$

**Disadvantages:** New model needs to be trained for each new corruption
Formulated as a 2-player minimax game between a Generator $G$ and discriminator $D$ with value function $V(G, D)$ where,

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}}(x)[\log D(x)] + \mathbb{E}_{z \sim p_z}(z)[1 - D(G(z))]$$
Overview of Generative Adversarial Nets I

Formulated as a 2-player minimax game between a Generator $G$ and discriminator $D$ with value function $V(G, D)$ where,

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[1 - D(G(z))]$$

Intuitively,

- $D$ is a classifier that predicts if the given input belongs to the training dataset
- $G$ is a function that generate signals that are able to fool $D$ from a random latent variable $z$
Overview of Generative Adversarial Nets I

Formulated as a 2-player minimax game between a Generator $G$ and discriminator $D$ with value function $V(G, D)$ where,

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}} \left[ \log D(x) \right] + \mathbb{E}_{z \sim p_z} \left[ 1 - D(G(z)) \right]$$

Intuitively,

- $D$ is a classifier that predicts if the given input belongs to the training dataset
- $G$ is a function that generate signals that are able to fool $D$ from a random latent variable $z$

Note that GANs do not model $p_x$ explicitly.

Credit: Goodfellow et al. NIPS 2014
Convincing faces generated by fully convolutional GANs (DCGAN)

Credit: Radford et al. ICLR 2016
Leveraging the success of GANs, we combine the flexibility of traditional approaches together with the power of a data-driven prior.
Leveraging the success of GANs, we combine the flexibility of traditional approaches together with the power of a data-driven prior.

Ideally, we would like to solve the following MAP problem,

$$\arg\min_x \|y - Ax\|_p + \lambda \log p_X(x)$$
Leveraging the success of GANs, we combine the flexibility of traditional approaches together with the power of a data-driven prior.

Ideally, we would like to solve the following MAP problem,

$$\arg\min_x ||y - Ax||_p + \lambda \log p_X(x)$$

However, this cannot be done naively with GANs as $p_x$ is not modelled explicitly.
Objective function:

\[
\hat{z} = \arg \min_{\hat{z}} \| y - A(G(z)) \|_p \\
+ \lambda \left( \log(1 - D(G(z))) - \log(D(G(z)) + \log(p_z(z)) \right)
\]
Objective function:

\[ \hat{z} = \arg \min_{\hat{z}} \|y - A(G(\hat{z}))\|_p \]

\[ + \lambda \left( \log(1 - D(G(\hat{z}))) - \log(D(G(\hat{z}))) + \log(p_z(z)) \right) \]

- the first term is the reconstruction loss or the data fidelity term

We solve for \( \hat{z} \), initialized randomly, using gradient descent variants (e.g., ADAM). Finally \( \hat{x} = G(\hat{z}) \), and optional blending step can also be applied if desired.
Objective function:

\[ \hat{z} = \arg \min_{z} \|y - A(G(z))\|_p \]

\[ + \lambda \left( \log(1 - D(G(z))) - \log(D(G(z)) + \log(p_z(z)) \right) \]

- the first term is the reconstruction loss or the data fidelity term
- the second term is our proposed data-driven prior.
Our Proposed Method II

Objective function:

\[
\hat{z} = \arg \min_{\hat{z}} \| y - A(G(\hat{z})) \|_p \\
+ \lambda \left( \log(1 - D(G(\hat{z}))) - \log(D(G(\hat{z})) + \log(p_z(\hat{z}))) \right)
\]

- the first term is the reconstruction loss or the data fidelity term
- the second term is our proposed data-driven prior.
- We solve for \( \hat{z} \), initialized randomly, using gradient descent variants (e.g. ADAM).
Objective function:

\[ \hat{z} = \text{arg min}_{\hat{z}} \| y - A(G(\hat{z})) \|_p \]

\[ + \lambda \left( \log(1 - D(G(\hat{z}))) - \log(D(G(\hat{z})) + \log(p_{\mathbf{z}}(\hat{z}))) \right) \]

- the first term is the reconstruction loss or the data fidelity term
- the second term is our proposed data-driven prior.
- We solve for \( \hat{z} \), initialized randomly, using gradient descent variants (e.g. ADAM).
- Finally \( \hat{x} = G(\hat{z}) \), and optional blending step can also be applied if desired.
Our Proposed Method - Assumptions

Assumptions:

- we know the class of images we are restoring
- we have a corresponding well-trained generator $G$ and discriminator $D$ for this class of images
Ideally we would like to use $p_X(x)$ as the prior. However, this is not available for GANs. For a fixed $G$, the optimal discriminator $D$ for a given generator $G$ is

$$D^*(x) = \frac{p_X(x)}{p_X(x) + p_G(x)},$$
Ideally we would like to use $p_X(x)$ as the prior. However, this is not available for GANs. For a fixed $G$, the optimal discriminator $D$ for a given generator $G$ is

$$D^*(x) = \frac{p_X(x)}{p_X(x) + p_G(x)},$$

Rearranging terms,

$$\log(p_X(x)) = \log(D(x)) - \log(1 - D(x))$$

$$+ \log(p_Z(z)) + \log \left( \left| \frac{\partial z}{\partial x} \right| \right),$$

where $p_G(x) = p_Z(z) \left| \frac{\partial z}{\partial x} \right|$. Since $\left| \frac{\partial z}{\partial x} \right|$ is intractable to compute, we assume it to be constant.
Finally we need to choose an $A$ for the restoration task. $A$ should:

- reflect the *forward* operation that generates the corruption
- sub-differentiable
Choice of $A$

Finally we need to choose an $A$ for the restoration task. $A$ should:

- reflect the *forward* operation that generates the corruption
- sub-differentiable

For specific tasks:

- **Image Inpainting**: (weighted) masking function
- **Image Colorization**: RGB to HSV conversion, using only V (RGB to grayscale)
- **Image Super Resolution**: Down sampling operation
- **Image Denoising**: Identity
- **Image Quantization**: Identity. Ideally, a step function might make sense but it produces no meaningful gradients
Datasets and Corruption Process

Dataset:

- GAN trained on CelebA dataset
- Faces were aligned and cropped to $64 \times 64$
Datasets and Corruption Process

Dataset:

- GAN trained on CelebA dataset
- Faces were aligned and cropped to $64 \times 64$

Corruption process:

- **Semantic Inpainting**: The corruption method is a missing center patch of $32 \times 32$;
- **Colorization**: The corruption is the standard grayscale conversion;
- **Super Resolution**: The corruption corresponds to downsampling by a factor of 4;
- **Denoising**: The corruption applies additive Gaussian noise, with standard deviation of 0.1 (pixel intensities from 0 to 1);
- **Quantization**: The corruption quantizes with 5 discrete levels per channel.
Visualization of Optimization for Inpainting

Credit: Yeh et al. CVPR 2017
Table: Quantitative comparison on image restoration tasks using SSIM and PSNR(dB).

<table>
<thead>
<tr>
<th>Applications Metric</th>
<th>Inpainting</th>
<th>Colorization</th>
<th>Super Res</th>
<th>Denoising</th>
<th>Quantization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SSIM PSNR</td>
<td>SSIM PSNR</td>
<td>SSIM PSNR</td>
<td>SSIM PSNR</td>
<td>SSIM PSNR</td>
</tr>
<tr>
<td>TV$^a$</td>
<td>0.7647 23.10</td>
<td>- -</td>
<td>0.6648 21.05</td>
<td>0.7373 21.97</td>
<td>0.6312 20.77</td>
</tr>
<tr>
<td>LR$^b$</td>
<td>0.6644 16.98</td>
<td>- -</td>
<td><strong>0.6754 21.45</strong></td>
<td>0.6178 18.69</td>
<td>0.6754 20.65</td>
</tr>
<tr>
<td>Sparse$^c$</td>
<td>0.7528 20.67</td>
<td>- -</td>
<td>0.6075 20.82</td>
<td><strong>0.8092 23.63</strong></td>
<td><strong>0.7869 22.67</strong></td>
</tr>
<tr>
<td>Ours</td>
<td><strong>0.8121 23.60</strong></td>
<td>0.8876 20.85</td>
<td>0.5626 19.58</td>
<td>0.6161 19.31</td>
<td>0.6061 19.77</td>
</tr>
</tbody>
</table>

Other than inpainting, our method seems to perform poorly under these metrics. But is that the full story?

- Afonso et al. TIP 2011
- Hu et al. PAMI 2013
- Elad et al. CVPR 2006, Yang et al. TIP 2010
Table: Quantitative comparison on image restoration tasks using SSIM and PSNR(dB).

<table>
<thead>
<tr>
<th>Applications</th>
<th>Metric</th>
<th>Inpainting</th>
<th>Colorization</th>
<th>Super Res</th>
<th>Denoising</th>
<th>Quantization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SSIM</td>
<td>PSNR</td>
<td>SSIM</td>
<td>PSNR</td>
<td>SSIM</td>
<td>PSNR</td>
</tr>
<tr>
<td>TV\textsuperscript{a}</td>
<td>0.7647</td>
<td>23.10</td>
<td>-</td>
<td>-</td>
<td>0.6648</td>
<td>21.05</td>
</tr>
<tr>
<td>LR\textsuperscript{b}</td>
<td>0.6644</td>
<td>16.98</td>
<td>-</td>
<td>-</td>
<td>0.6754</td>
<td>21.45</td>
</tr>
<tr>
<td>Sparse\textsuperscript{c}</td>
<td>0.7528</td>
<td>20.67</td>
<td>-</td>
<td>-</td>
<td>0.6075</td>
<td>20.82</td>
</tr>
<tr>
<td>Ours</td>
<td>0.8121</td>
<td>23.60</td>
<td>0.8876</td>
<td>20.85</td>
<td>0.5626</td>
<td>19.58</td>
</tr>
</tbody>
</table>

Other than inpainting, our method seems to perform poorly under these metrics.
But is that the full story?

\textsuperscript{a} Afonso \textit{et al.} TIP 2011
\textsuperscript{b} Hu \textit{et al.} PAMI 2013
\textsuperscript{c} Elad \textit{et al.} CVPR 2006, Yang \textit{et al.} TIP 2010
<table>
<thead>
<tr>
<th></th>
<th>Real</th>
<th>Input</th>
<th>TV</th>
<th>LR</th>
<th>Sparse</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Inpainting</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Colorization</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Super Res</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Denoising</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Quantization</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Qualitative Results II

<table>
<thead>
<tr>
<th></th>
<th>Real</th>
<th>Input</th>
<th>TV</th>
<th>LR</th>
<th>Sparse</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inpainting</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Colorization</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Super Res</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Denoising</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quantization</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Conclusion

Contributions:

- Using GANs as a data-driven prior
- Same model can be used for different problems (no re-training!)
- Not restricted to a specific generative network
Conclusion

Contributions:

- Using GANs as a data-driven prior
- Same model can be used for different problems (no re-training!)
- Not restricted to a specific generative network

Limitations and potential improvements:

- Current GANs are not yet able to handle general images
- Better initial $z$, perhaps with a LUT or another deep network?
Questions?

Code and more examples at:

https://goo.gl/vNokXj