Image Generation and GANs

CSE P576

Dr. Matthew Brown
Image Generation + GANs

- Can a neural net define the loss function?
- Loss functions for Super-Resolution: L2, VGG, Adversarial
- Generative Adversarial Nets and Image Generation
- Conditional GANs, Image Translation, pix2pix
Super-Resolution: SRCNN

- Small network (3 layers) generates reasonable results

![Diagram of SRCNN](image)

- Feature maps
- Non-linear mapping
- Reconstruction

Train using L2 loss vs ground truth

[ SRCNN, Dong et al 2014 ]
Super-Resolution: SRResNet

- Deeper networks generate better results, e.g., SRResNet

Trained with L2 loss, state-of-the-art PSNR in 2017

[ Ledig et al 2017 ]
A state of the art super-res network trained with L2 loss is good at sharpening edges, but results lack realistic texture.
SRResNet
(23.53dB/0.7832)
SRResNet
(23.53dB/0.7832)

More realistic, but L2
loss is worse
(21.15dB/0.6868)
Perceptual Metrics

• L2 loss does not match human perception in general

[ Zhang et al 2018 ]
Texture Synthesis

- Which are the real radishes?
Texture Synthesis

- Which are the real rocks?
\[
\text{loss} = E_L = \sum \left( \hat{G}^L - G^L \right)^2
\]

\(G = \text{Gram-matrix}\)

\[
\hat{G}_{ij}^{L} = \sum_{k} \hat{F}_{ik}^{L} \hat{F}_{jk}^{L}
\]

\[
\frac{\partial E_L}{\partial \hat{F}^L} \quad \frac{\partial E_L}{\partial \hat{F}^{L-1}}
\]

\[
\frac{\partial \mathcal{L}}{\partial \hat{x}} \quad \text{Gradient descent}
\]

\[
\mathcal{L}(\hat{x}, \hat{x}) = \sum_{l=0}^{L} w_l E_l
\]

Match \(\Sigma\) of outer products of feature activations
Portilla Simoncelli 1999 texture model also used correlation of filter responses (though shallow features / hand tuned)
Figure 2: Generated stimuli. Each row corresponds to a different processing stage in the network. When only constraining the texture representation on the lowest layer, the synthesised textures have little structure, similarly to spectrally matched noise (first row). With increasing number of layers on which we match the texture representation we find that we generate images with increasing degree of naturalness (rows 2–5; labels on the left indicate the top-most layer included). The source textures in the first three columns were previously used by Portilla and Simoncelli [21]. For better comparison we also show their results (last row). The last column shows textures generated from a non-texture image to give a better intuition about how the texture model represents image information.
Style Transfer

Re-render an image given the “style” of an artist [ Gatys et al 2015 ]
Content and Style Losses

\[ E(\text{content}) = \frac{1}{2} \sum_{i,j} (F_{ij}^l - P_{ij}^l)^2 \]

\[ E(\text{style}) = E_L = \sum \left( \hat{G}_L^L - G_L^L \right)^2 \]

Figure 1: Synthesis method. Texture analysis (left). The original texture is passed through the CNN and the Gram matrices \( G_l \) on the feature responses of a number of layers are computed. Texture synthesis (right). A white noise image \( \hat{x} \) is passed through the CNN and a loss function \( E_l \) is computed on every layer included in the texture model. The total loss function \( L \) is a weighted sum of the contributions \( E_l \) from each layer. Using gradient descent on the total loss with respect to the pixel values, a new image is found that produces the same Gram matrices \( \hat{G}_l \) as the original texture.

In this work, we propose a new parametric texture model to tackle this problem (Fig. 1). Instead of describing textures on the basis of a model for the early visual system [21, 10], we use a convolutional neural network – a functional model for the entire ventral stream – as the foundation for our texture model. We combine the conceptual framework of spatial summary statistics on feature responses with the powerful feature space of a convolutional neural network that has been trained on object recognition. In that way we obtain a texture model that is parameterised by spatially invariant representations built on the hierarchical processing architecture of the convolutional neural network.
Feedforward Style Transfer

Match neural network features (content and style)

12-layer, residual conn., fully conv

[ Johnson et al. 2016 ]
Feedforward Style Transfer

Comparable results to with much lower computational costs
(single feedforward pass vs 1000s of backprop iterations)
Super-Resolution

- Small networks are generally good at sharpening edges and can work well for small factor (e.g., 2) super-resolution.
- Better results can be achieved by using deeper networks, + more sophisticated loss functions (perceptual loss, GANs).

*12-layer, residual conn., fully conv, VGG loss [ Johnson et al. 2016 ]
Super-Resolution with VGG loss

Note: style loss not used for super-res

Low-res input

High-res target

Match neural net features (content loss)

[ Johnson et al. 2016 ]
Super-Resolution with VGG loss

- Results

Ground Truth  Bicubic  SRCNN  Johnson et al. VGG loss

[Johnson et al. 2016]
Super-Resolution with VGG loss

SRCNN

Johnson
Super-Resolution with VGG loss

<table>
<thead>
<tr>
<th>Ground Truth</th>
<th>Bicubic</th>
<th>Ours ($\ell_{pixel}$)</th>
<th>Ours ($\ell_{feat}$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>This image</td>
<td>22.75 / 0.5946</td>
<td>23.42 / 0.6168</td>
<td>21.90 / 0.6083</td>
</tr>
<tr>
<td>Set5 mean</td>
<td>23.80 / 0.6455</td>
<td>24.77 / 0.6864</td>
<td>23.26 / 0.7058</td>
</tr>
<tr>
<td>Set14 mean</td>
<td>22.37 / 0.5518</td>
<td>23.02 / 0.5787</td>
<td>21.64 / 0.5837</td>
</tr>
<tr>
<td>BSD100 mean</td>
<td>22.11 / 0.5322</td>
<td>22.54 / 0.5526</td>
<td>21.35 / 0.5474</td>
</tr>
</tbody>
</table>

[Johnson et al. 2016]
SRGAN

- Can we train a neural network to define the loss?

[ Ledig et al. 2017 ]
Figure 2: From left to right: bicubic interpolation, deep residual network optimized for MSE, deep residual generative adversarial network optimized for a loss more sensitive to human perception, original HR image. Corresponding PSNR and SSIM are shown in brackets.

In this work we propose a super-resolution generative adversarial network (SRGAN) for which we employ a deep residual network (ResNet) with skip-connection and diverge from MSE as the sole optimization target. Different from previous works, we define a novel perceptual loss using high-level feature maps of the VGG network combined with a discriminator that encourages solutions perceptually hard to distinguish from the HR reference images. An example photo-realistic image that was super-resolved with a $4 \times$ upscaling factor is shown in Figure 1.

## 1.1. Related work

### 1.1.1 Image super-resolution

Recent overview articles on image SR include Nasrollahi and Moeslund or Yang et al. Here we will focus on single image super-resolution (SISR) and will not further discuss approaches that recover HR images from multiple images.

Prediction-based methods were among the first methods to tackle SISR. While these filtering approaches, e.g., linear, bicubic or Lanczos filtering, can be very fast, they oversimplify the SISR problem and usually yield solutions with overly smooth textures. Methods that put particularly focus on edge-preservation have been proposed.

More powerful approaches aim to establish a complex mapping between low- and high-resolution image information and usually rely on training data. Many methods that are based on example-pairs rely on LR training patches for which the corresponding HR counterparts are known. Early work was presented by Freeman et al. Related approaches to the SR problem originate in compressed sensing. In Glasner et al. the authors exploit patch redundancies across scales within the image to drive the SR. This paradigm of self-similarity is also employed in Huang et al., where self dictionaries are extended by further allowing for small transformations and shape variations. Gu et al. proposed a convolutional sparse coding approach that improves consistency by processing the whole image rather than overlapping patches.

To reconstruct realistic texture detail while avoiding edge artifacts, Tai et al. combine an edge-directed SR algorithm based on a gradient profile prior with the benefits of learning-based detail synthesis. Zhang et al. propose a multi-scale dictionary to capture redundancies of similar image patches at different scales. To super-resolve landmark images, Yue et al. retrieve correlating HR images with similar content from the web and propose a structure-aware matching criterion for alignment.

Neighborhood embedding approaches upsample a LR image patch by finding similar LR training patches in a low dimensional manifold and combining their corresponding HR patches for reconstruction. In Kim and Kwon the authors emphasize the tendency of neighborhood approaches to overfit and formulate a more general map of example pairs using kernel ridge regression. The regression problem can also be solved with Gaussian process regression, trees or Random Forests. In Dai et al. a multitude of patch-specific regressors is learned and the most appropriate regressors selected during testing. Recently convolutional neural network (CNN) based SR...
SRGAN

Generator performs super-resolution, tries to fool discriminator

Discriminator tries to detect real vs super-resolved images
Generative Adversarial Networks

Setup: Assume we have data $x_i$ drawn from distribution $p_{data}(x)$. Want to sample from $p_{data}$. 

Generative Adversarial Networks

Setup: Assume we have data $x_i$ drawn from distribution $p_{\text{data}}(x)$. Want to sample from $p_{\text{data}}$.

Idea: Introduce a latent variable $z$ with simple prior $p(z)$. Sample $z \sim p(z)$ and pass to a Generator Network $x = G(z)$. Then $x$ is a sample from the Generator distribution $p_G$. Want $p_G = p_{\text{data}}$. 

Generative Adversarial Networks

**Setup**: Assume we have data \( x_i \) drawn from distribution \( p_{\text{data}}(x) \). Want to sample from \( p_{\text{data}} \).

**Idea**: Introduce a latent variable \( z \) with simple prior \( p(z) \).
Sample \( z \sim p(z) \) and pass to a **Generator Network** \( x = G(z) \)
Then \( x \) is a sample from the **Generator distribution** \( p_G \). Want \( p_G = p_{\text{data}} \).

Sample \( z \) from \( p_z \)

Train **Generator Network** \( G \) to convert \( z \) into fake data \( x \) sampled from \( p_G \)

---

Generative Adversarial Networks

**Setup:** Assume we have data $x_i$ drawn from distribution $p_{\text{data}}(x)$. Want to sample from $p_{\text{data}}$.

**Idea:** Introduce a latent variable $z$ with simple prior $p(z)$. Sample $z \sim p(z)$ and pass to a **Generator Network** $x = G(z)$. Then $x$ is a sample from the **Generator distribution** $p_G$. Want $p_G = p_{\text{data}}$.

Jointly train $G$ and $D$. Hopefully $p_G$ converges to $p_{\text{data}}$!

---

**Diagram:**
- Sample $z$ from $p_z$
- Train **Generator Network** $G$ to convert $z$ into fake data $x$ sampled from $p_G$
- by "fooling" the **discriminator Network** $D$
- Train **Discriminator Network** $D$ to classify data as real or fake (1/0)

Generative Adversarial Networks: Training Objective

Jointly train generator $G$ and discriminator $D$ with a **minimax game**

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

Generative Adversarial Networks: Training Objective

Jointly train generator G and discriminator D with a **minimax game**

\[
\min_G \max_D \left( E_{x \sim p_{\text{data}}} \left[ \log D(x) \right] + E_{z \sim p(z)} \left[ \log \left( 1 - D(G(z)) \right) \right] \right)
\]

**Discriminator wants**

\( D(x) = 1 \) for real data

Sample \( z \) from \( p_z \)

Sample

G

Generated Sample

D

Discriminator Network

Fake

Real

Goodfellow et al, "Generative Adversarial Nets", NeurIPS 2014
Generative Adversarial Networks: Training Objective

Jointly train generator G and discriminator D with a minimax game

\[
\min_G \max_D \left( E_{x \sim p_{\text{data}}} [\log D(x)] + E_{z \sim p(z)} [\log (1 - D(G(z)))] \right)
\]

Discriminator wants
\( D(x) = 1 \) for real data

Discriminator wants
\( D(x) = 0 \) for fake data

Sample z from \( p_z \)

Generator Network

Generated Sample

Discriminator Network

Fake

Real

Generative Adversarial Networks: Training Objective

Jointly train generator G and discriminator D with a **minimax game**

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

- **Discriminator wants** 
  - \( D(x) = 1 \) for real data
  - \( D(x) = 0 \) for fake data

- **Generator wants** 
  - \( D(x) = 1 \) for fake data

Sample \( z \) from \( p_z \)

---

Generative Adversarial Networks: Training Objective

Jointly train generator $G$ and discriminator $D$ with a **minimax game**

Train $G$ and $D$ using alternating gradient updates

$$
\min_G \max_D \left( E_{x \sim p_{data}} \left[ \log D(x) \right] + E_{z \sim p(z)} \left[ \log \left( 1 - D(G(z)) \right) \right] \right)
$$

$$
= \min_G \max_D V(G, D)
$$

Generative Adversarial Networks: Training Objective

Jointly train generator \( G \) and discriminator \( D \) with a \textbf{minimax game}

Train \( G \) and \( D \) using alternating gradient updates

\[
\begin{align*}
\min_G \max_D \left( E_{x \sim p_{data}} [\log D(x)] + E_{z \sim p(z)} \left[ \log \left( 1 - D(G(z)) \right) \right] \right)
\end{align*}
\]

\[
= \min_G \max_D V(G, D)
\]

For \( t \) in 1, ... \( T \):

1. (Update \( D \)) \( D = D + \alpha_D \frac{\partial V}{\partial D} \)

2. (Update \( G \)) \( G = G - \alpha_G \frac{\partial V}{\partial G} \)

We are not minimizing any overall loss! No training curves to look at!

**SRGAN**

- Generator performs superres, discriminator attempts to detect real images vs super-resolved low res images

\[
\min_{\theta_G} \max_{\theta_D} \mathbb{E}_{I^{HR} \sim p_{train}(I^{HR})} \left[ \log D_{\theta_D}(I^{HR}) \right] + \mathbb{E}_{I^{LR} \sim p_G(I^{LR})} \left[ \log(1 - D_{\theta_D}(G_{\theta_G}(I^{LR}))) \right]
\]
Generative Adversarial Networks: Results

Generated samples

Original GAN paper, image generation, driven by noise input

Generative Adversarial Networks: DC-GAN

Generative Adversarial Networks: DC-GAN

Samples from the model look much better!

Radford et al, ICLR 2016
Generative Adversarial Networks: Interpolation

Interpolating between points in latent $z$ space

Radford et al, ICLR 2016
Generative Adversarial Networks: Vector Math

Samples from the model

Average Z vectors, do arithmetic

Smiling woman
Neutral woman
Neutral man

Radford et al, ICLR 2016
Generative Adversarial Networks: Vector Math

Samples from the model

Smiling woman
Neutral woman
Neutral man

Average Z vectors, do arithmetic

Smiling Man

Smiling woman
Neutral woman
Neutral man

Radford et al, ICLR 2016
Generative Adversarial Networks: Vector Math

Samples from the model

Man with glasses

Man w/o glasses

Woman w/o glasses

Average Z vectors, do arithmetic

Radford et al, ICLR 2016
Generative Adversarial Networks: Vector Math

Samples from the model

Average Z vectors, do arithmetic

Radford et al, ICLR 2016

Man with glasses

Man w/o glasses

Woman w/o glasses

Woman with glasses

Radford et al, ICLR 2016
GAN Questions

| Problem 1 | What are the trade-offs between GANs and other generative models? |
| Problem 2 | What sorts of distributions can GANs model? |
| Problem 3 | How can we Scale GANs beyond image synthesis? |
| Problem 4 | What can we say about the global convergence of the training dynamics? |
| Problem 5 | How should we evaluate GANs and when should we use them? |
| Problem 6 | How does GAN training scale with batch size? |
| Problem 7 | What is the relationship between GANs and adversarial examples? |

https://distill.pub/2019/gan-open-problems/
Image Translation

- Many problems in vision/graphics can be viewed as image translation problems

Can we build a general machine to translate images?

[ pix2pix, Isola et al. 2018 ]
Image Translation

- e.g., translation from grey to color should be indistinguishable from real

Note: pix2pix has an additional supervisory

$L1 \text{ loss } = |y - \hat{y}|$

This is a (conditional) Generative Adversarial Network
pix2pix: Segmentation → Image
Model trained using edge detection, but works for hand drawings:
#edges2cats by Christopher Hesse

sketch by Ivy Tsai

https://affinelayer.com/pixsrv/