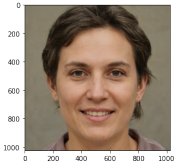
Generative adversarial neural networks for state-of-the-art image reconstruction.

Arthur Conmy and Dr Subhadip Mukherjee. *The following face does not exist: it was generated as the 'average' face from a pre-trained GAN. You can view your own synthetic faces (that no one will have seen before) at* `thispersondoesnotexist.com`*, and reload the page to generate a different face.*



Plan

1. Background context: machine learning for mathmos.
   - Neural nets.
   - Generative modelling.
   - GANs.

2. Our work.
   - Image reconstruction.
   - Our contribution.

Plan

1. Background context: machine learning for mathmos.
   - Neural nets.
   - Generative modelling.
   - GANs.
2. Our work.
   - Image reconstruction.
   - Our contribution.

Neural networks

For the purposes of this talk, it will be useful to think of neural networks as black-boxes that are very good at representing high-dimensional functions that lie in very large function classes.

- Dog / cat classifier:

$$f : [0,1]^{H\times W\times C} \to \{\mathrm{Dog}, \mathrm{Cat}\} \tag{1}$$

Neural networks

For the purposes of this talk, it will be useful to think of neural networks as black-boxes that are very good at representing high-dimensional functions that lie in very large function classes.

- Dog / cat classifier:

$$f : [0,1]^{H \times W \times C} \to \{\mathrm{Dog}, \mathrm{Cat}\} \tag{1}$$

- More exciting example to come:

$$\mathcal{F} = \{f : f : [0,1]^{H \times W \times C} \to \mathbb{R}, f \in \mathrm{Lip}(1)\} \tag{2}$$

Generative modelling.

- Consider a *fixed, but unknown* probability density function $p_d : \mathbb{R}^n \rightarrow \mathbb{R}$, of which we have access to a large number of i.i.d samples from: $x_1, x_2, ..., x_N \sim p_d$.

Generative modelling.

- Consider a *fixed, but unknown* probability density function $p_d : \mathbb{R}^n \to \mathbb{R}$, of which we have access to a large number of i.i.d samples from: $x_1, x_2, ..., x_N \sim p_d$.
- Open-ended problem: how can we generate samples $x'_1, x'_2, ...$ that are similar to the samples $x_1, x_2, ..., x_N$?

Generative modelling.

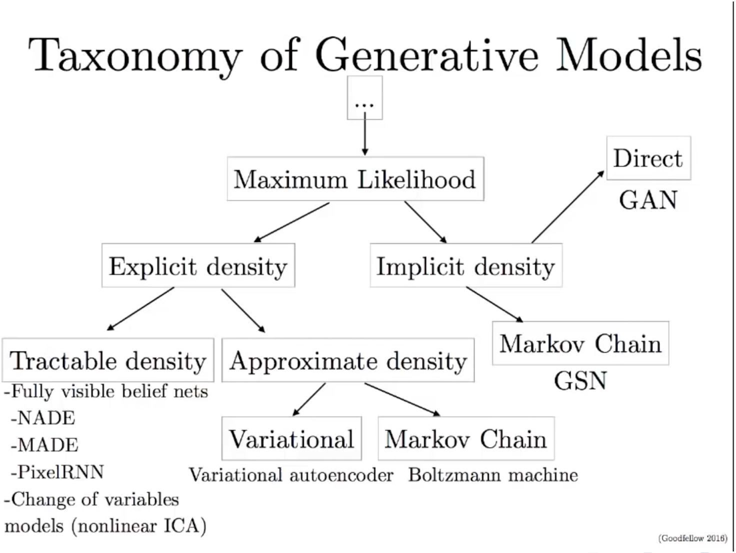
- Consider a *fixed, but unknown* probability density function $p_d : \mathbb{R}^n \to \mathbb{R}$, of which we have access to a large number of i.i.d samples from: $x_1, x_2, ..., x_N \sim p_d$.
- Open-ended problem: how can we generate samples $x_1', x_2', ...$ that are similar to the samples $x_1, x_2, ..., x_N$?
  - We assume that we derive such samples from some generated distribution p_g, but we are only required to have samples from p_g.

Generative modelling.

- Consider a *fixed, but unknown* probability density function $p_d : \mathbb{R}^n \to \mathbb{R}$, of which we have access to a large number of i.i.d samples from: $x_1, x_2, ..., x_N \sim p_d$.
- Open-ended problem: how can we generate samples $x'_1, x'_2, ...$ that are similar to the samples $x_1, x_2, ..., x_N$?
    - We assume that we derive such samples from some generated distribution p_g, but we are only required to have samples from p_g.
- Big question: what does 'similar' mean?

Taxonomy

From Goodfellow's 2016 NIPS GANs tutorial:



GANs, part one.

By analogy

- The gang have a generator of fake money.

By formalism

By analogy

- The gang have a generator of fake money.

By formalism

- $G : \mathbb{R}^m \to \mathcal{X}$

GANs, part one.

By analogy

- The gang have a generator of fake money.
- The police have a discriminator that (imperfectly) distinguishes fake from real money.

By formalism

- $G : \mathbb{R}^m \to \mathcal{X}$

GANs, part one.

By analogy

- The gang have a generator of fake money.
- The police have a discriminator that (imperfectly) distinguishes fake from real money.

By formalism

- $G : \mathbb{R}^m \to \mathcal{X}$
- $D : \mathcal{X} \to (0, 1)$
  representing the probability an image is real.

GANs, part one.

By analogy

- The gang have a generator of fake money.
- The police have a discriminator that (imperfectly) distinguishes fake from real money.
- Have the gang try and outwit the police, while the police try and uncover the gang.

By formalism

- $G : \mathbb{R}^m \to \mathcal{X}$
- $D : \mathcal{X} \to (0, 1)$
  representing the probability an image is real.

GANs, part one.

By analogy

- The gang have a generator of fake money.
- The police have a discriminator that (imperfectly) distinguishes fake from real money.
- Have the gang try and outwit the police, while the police try and uncover the gang.

By formalism

- $G : \mathbb{R}^m \to \mathcal{X}$
- $D : \mathcal{X} \to (0, 1)$
  representing the probability an image is real.
- Jointly optimize G (to maximise the probability D misclassifies generated images), and D (to minimise the probability it misclassifies images).

GANs, part one.

By analogy

- The gang have a generator of fake money.
- The police have a discriminator that (imperfectly) distinguishes fake from real money.
- Have the gang try and outwit the police, while the police try and uncover the gang.

By formalism

- $G : \mathbb{R}^m \to \mathcal{X}$
- $D : \mathcal{X} \to (0,1)$ representing the probability an image is real.
- Jointly optimize G (to maximise the probability D misclassifies generated images), and D (to minimise the probability it misclassifies images).

- Given some loss function penalising G and D, we're already in the setting to apply back-prop, but sadly this is a FAR more delicate procedure than suggested.

GANs, part one.

By analogy

- The gang have a generator of fake money.
- The police have a discriminator that (imperfectly) distinguishes fake from real money.
- Have the gang try and outwit the police, while the police try and uncover the gang.

By formalism

- $G : \mathbb{R}^m \to \mathcal{X}$
- $D : \mathcal{X} \to (0, 1)$ representing the probability an image is real.
- Jointly optimize G (to maximise the probability D misclassifies generated images), and D (to minimise the probability it misclassifies images).

- Given some loss function penalising G and D, we're already in the setting to apply back-prop, but sadly this is a FAR more delicate procedure than suggested.

GANs, part one.

By analogy

- The gang have a generator of fake money.
- The police have a discriminator that (imperfectly) distinguishes fake from real money.
- Have the gang try and outwit the police, while the police try and uncover the gang.

By formalism

- $G : \mathbb{R}^m \to \mathcal{X}$
- $D : \mathcal{X} \to (0, 1)$ representing the probability an image is real.
- Jointly optimize G (to maximise the probability D misclassifies generated images), and D (to minimise the probability it misclassifies images).

- Given some loss function penalising G and D, we're already in the setting to apply back-prop, but sadly this is a FAR more delicate procedure than suggested.

GANs, part one.

By analogy

- The gang have a generator of fake money.
- The police have a discriminator that (imperfectly) distinguishes fake from real money.
- Have the gang try and outwit the police, while the police try and uncover the gang.

By formalism

- $G : \mathbb{R}^m \to \mathcal{X}$
- $D : \mathcal{X} \to (0, 1)$ representing the probability an image is real.
- Jointly optimize G (to maximise the probability D misclassifies generated images), and D (to minimise the probability it misclassifies images).

- Given some loss function penalising G and D, we're already in the setting to apply back-prop, but sadly this is a FAR more delicate procedure than suggested.
- HOWEVER, significant advances have been made to GAN theory since this initial work, citing Cedric Villani's optimal transport theory to redesign the loss function between the distributions!

GANs, part two

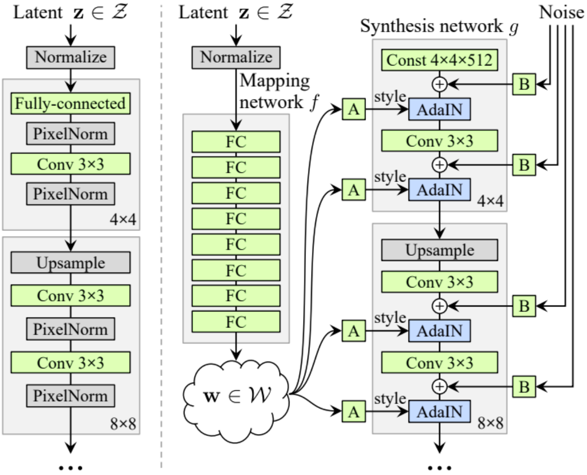
The most natural way to choose a loss function is the analogue of cross entropy loss in this case:

$$V(D, G) = \mathbb{E}_{p_d}[\log D(x)] + \mathbb{E}_{z \sim \mathcal{N}}[\log(1 - D(G(z)))].$$

A much better choice of loss function between distributions is the Wasserstein distance:

$$\mathrm{EMD}(p_d, p_g) = \sup_{\|f\|_{L \leq 1}} \mathbb{E}_{x \sim p_d} f(x) - \mathbb{E}_{x \sim p_g} f(x).$$

GANs, part three.



(a) Traditional (b) Style-based generator

Plan

1. Background context: machine learning for mathmos.
   - Neural nets.
   - Generative modelling.
   - GANs.
2. Our work.
   - Image reconstruction.
   - Our contribution.

Inverse problems

- Inverse problems are ill-posed problems where $y = A(x)$, and we need reconstruct x given y.

Inverse problems

- Inverse problems are ill-posed problems where $y = A(x)$, and we need reconstruct x given y.
- Non-uniqueness of solutions means that it turns out to be important to solve

$$\hat{x} \in \underset{x}{\operatorname{argmin}}\, L\left(y, A(x)\right) + \lambda\, \psi(x) \tag{3}$$

where L is the loss of the reconstruction, and ψ a *regularizer*.

Inverse problems

- Inverse problems are ill-posed problems where $y = A(x)$, and we need reconstruct x given y.
- Non-uniqueness of solutions means that it turns out to be important to solve

$$\hat{x} \in \operatorname*{argmin}_{x} L\left(y, A(x)\right) + \lambda\, \psi(x) \tag{3}$$

where L is the loss of the reconstruction, and ψ a *regularizer*.

- 

Our contribution.

- The previous SOTA for image reconstruction (where reconstruction could mean from down-sampling to lower resolution, masking parts of the image, or blurring (the last not considered in out work)) was *BRGM*:

Our contribution.

- The previous SOTA for image reconstruction (where reconstruction could mean from down-sampling to lower resolution, masking parts of the image, or blurring (the last not considered in out work)) was *BRGM*:

$$w^* = \arg\min_w \underbrace{\sum_i \left(\frac{w_i - \mu}{\sigma_i}\right)^2}_{\mathcal{L}_w} - 2\kappa \underbrace{\sum_{i,j} \frac{w_i w_j^T}{|w_i||w_j|}}_{\mathcal{L}_{colin}} + \sigma_{pixel}^{-2} \underbrace{\|I - f \circ G(w)\|_2^2}_{\mathcal{L}_{pixel}} + \sigma_{percept}^{-2} \underbrace{\|I - \phi \circ f \circ G(w)\|_2^2}_{\mathcal{L}_{percept}} \quad (6)$$

-

Our contribution.

- The previous SOTA for image reconstruction (where reconstruction could mean from down-sampling to lower resolution, masking parts of the image, or blurring (the last not considered in out work)) was *BRGM*:
- $$w^* = \arg\min_w \underbrace{\sum_i \left(\frac{w_i - \mu}{\sigma_i}\right)^2}_{\mathcal{L}_w} - 2\kappa \underbrace{\sum_{i,j} \frac{w_i w_j^T}{|w_i||w_j|}}_{\mathcal{L}_{colin}} + \sigma_{pixel}^{-2} \underbrace{\|I - f \circ G(w)\|_2^2}_{\mathcal{L}_{pixel}} + \sigma_{percept}^{-2} \underbrace{\|I - \phi \circ f \circ G(w)\|_2^2}_{\mathcal{L}_{percept}} \quad (6)$$
- We used a new 'guided' approach to the optimization over the two latent spaces.

Our contribution.

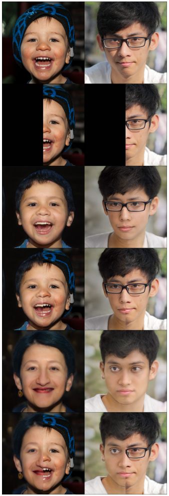
- The previous SOTA for image reconstruction (where reconstruction could mean from down-sampling to lower resolution, masking parts of the image, or blurring (the last not considered in out work)) was *BRGM*:

-
$$w^* = \arg\min_w \underbrace{\sum_i \left(\frac{w_i - \mu}{\sigma_i}\right)^2}_{\mathcal{L}_w} - 2\kappa \underbrace{\sum_{i,j} \frac{w_i w_j^T}{|w_i||w_j|}}_{\mathcal{L}_{colin}} + \sigma_{pixel}^{-2} \underbrace{\|I - f \circ G(w)\|_2^2}_{\mathcal{L}_{pixel}} + \sigma_{percept}^{-2} \underbrace{\|I - \phi \circ f \circ G(w)\|_2^2}_{\mathcal{L}_{percept}} \quad (6)$$

- We used a new 'guided' approach to the optimization over the two latent spaces.

-
The resulting optimization reads $\min_{(z,w^+)} J(w^+, z)$, where

$$J(w^+, z) := \lambda_{\text{pix}} \|y - A\,G(w^+)\|_2^2 + \lambda_{\text{map}} \sum_{i=1}^{L} \|w_i - M(z)\|_2^2 + \lambda_{\text{vgg}} \|\varphi(y) - \varphi(A\,G(w^+))\|_2^2 + \lambda_{\text{lat}} \|z\|_2^2. \quad (5)$$



We changed the regularization term used to reconstruct images, and obtained strong results.

	Input	Mask	Model	LPIPS				SSIM	
Inpainting	256^2	Half	DeepGIN	1	0.303	2	0.303	**59**	**0.625**
			BRGM	47	0.226	**57**	**0.231**	21	0.606
			L-BRGM	**51**	**0.224**	40	0.240	19	0.594
	1024^2	Half	BRGM	17	0.495				
			L-BRGM	**82**	**0.460**				
	1024^2	Eyepatch	BRGM	10	0.428				
			L-BRGM	**89**	**0.393**				
	Input	**Output**	**Model**	LPIPS				SSIM	
Super-resolution	64^2	256^2	GFPGAN	**72**	**0.231**	**77**	**0.231**	**89**	**0.534**
			BRGM	6	0.294	9	0.295	10	0.480
			L-BRGM	21	0.277	13	0.294	0	0.404
	64^2	1024^2	BRGM	32	0.426				
			L-BRGM	**67**	**0.414**				
	32^2	1024^2	BRGM	33	0.470				
			L-BRGM	**66**	**0.445**				

Table 1. Quantatitive evaluation of L-BRGM, BRGM, DeepGIN and GFPGAN. The two columns marked with an asterix refer to minimal *final* values of those metrics, i.e on the 2000th optimization step for (L-)BRGM.