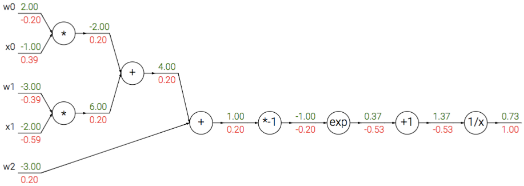
7. Generative Adversarial Networks

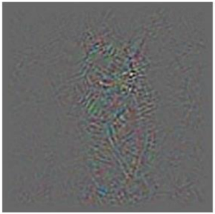
Adversarial examples

- Consider a case where an adversary knows some combination of
  - the training data
  - the trained mode weights
  - the trained model as a black box
- the goal of an adversary is to make the classifier fail (sometimes with emphasis on particular classes or examples)
- Timeline:
  - "Adversarial Classification" Dalvi et al 2004: fool spam filter
  - "Evasion Attacks Against Machine Learning at Test Time" Biggio 2013: fool neural nets
  - Szegedy et al 2013: fool ImageNet classifiers imperceptibly
  - Goodfellow et al 2014: cheap, closed form attack

Adversarial testing examples

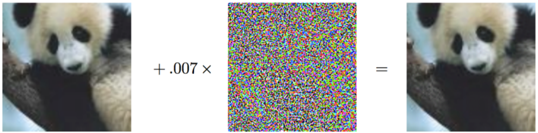
- Recall the back-propagation algorithm, which can be used to perform gradient descent over the **input example**, which itself is hard to interpret





Adversarial testing examples

- consider an experiment where we do gradient **ascent** on the cross-entropy loss to **minimize** the probability that it is correctly classified
- alternatively, we could also to gradient **descent** on a particular target class
- concretely, perturb the image slightly by taking the sign of the gradient with a small scaling constant



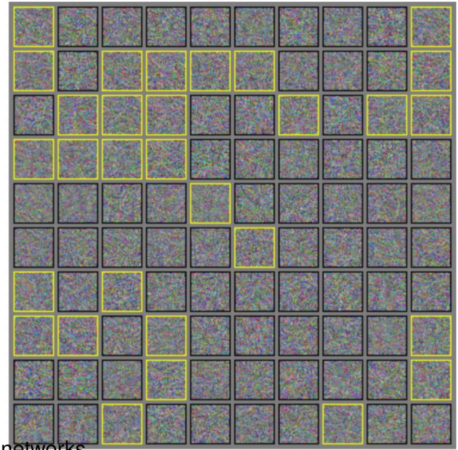
$\boldsymbol{x}$ "panda" 57.7% confidence

$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$ "nematode" 8.2% confidence

$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$ "gibbon" 99.3 % confidence

Adversarial testing examples

- In another experiment, you can start with a random noise and take **one** gradient step
- this often produces a confident classification
- the images outlined by yellow are classified as "airplane" with >50% confidence

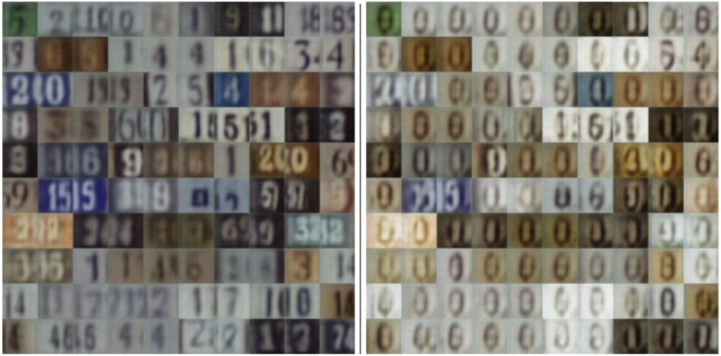


Adversarial testing examples

- In another experiment, you can have **targeted adversarial examples**, to misclassify examples to a specific target class
- the adversarial examples are misclassified as ostriches, and in the middle we show the perturbation times ten.

Adversarial testing examples

- consider a variational autoencoder for images
- one can create adversarial images that is reconstructed (after compression) as an entirely different image

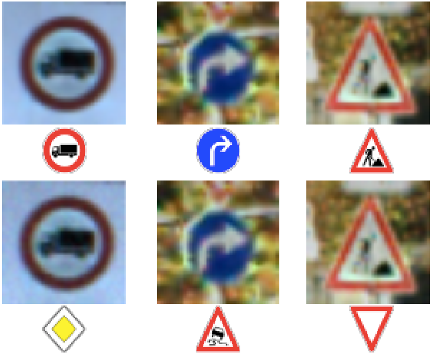


Adversarial testing examples

- First reported in ["Intriguing properties of neural networks", 2013, by Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, Rob Fergus]
- Led to serious concerns for security as, for example,
  - one can create road signs that fools a self-driving car to act in a certain way
- this is serious as
  - there is no reliable defense against adversarial examples
  - adversarial examples transfer to different networks, trained on disjoint subset of training data
  - you do not need the access to the model parameters; you can train your own model and create adversarial examples
  - you only need a black-box access via APIs (MetaMind, Amazon, Google)

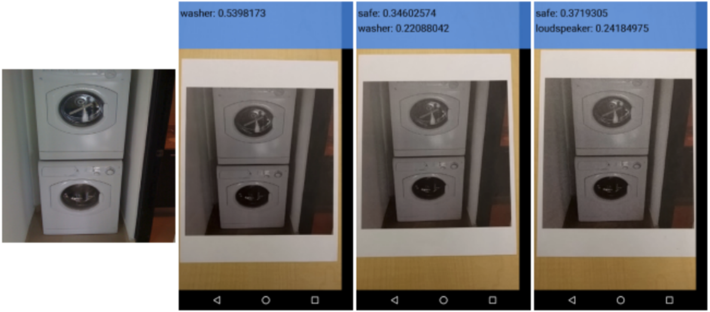
Adversarial testing examples

- ["Practical Black-Box Attacks against Machine Learning", 2016, Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, Ananthram Swami]
- no access to the actual classifier, only treat as a black-box



Adversarial testing examples

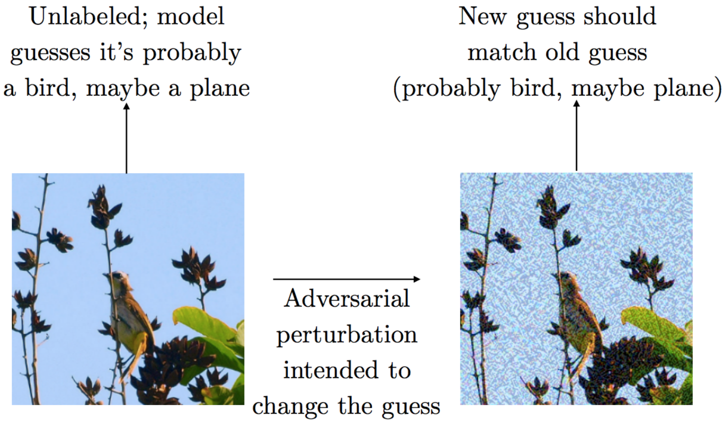
- You can fool a classifier by taking picture of a print-out.
- ["Adversarial examples in the physical world", 2016, Alexey Kurakin, Ian Goodfellow, Samy Bengio]
- one can potentially print over a stop sign to fool a self-driving car



(a) Image from dataset (b) Clean image (c) Adv. image, $\epsilon = 4$ (d) Adv. image, $\epsilon = 8$

Defense mechanism to adversarial testing examples

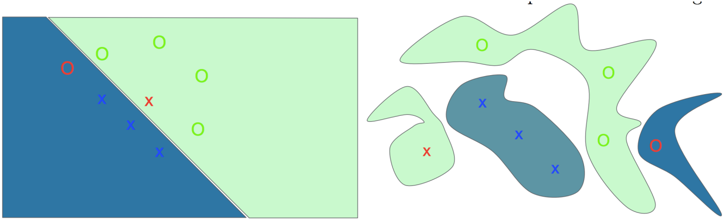
- Brute force: include adversarial testing examples (but with the correct classes) in the training data.



- Defensive distillation:
  - Two models are trained
  - model 1: trained on the training data in as standard manner
  - model 2 (the robust model) : is trained on the same training data, but uses **soft classes** which is the probability provided by the first model
  - This creates a model whose surface is smoothed in the directions an adversary will typically try to exploit, making it difficult for them to discover adversarial input tweaks that lead to incorrect categorization
  - [Distilling the Knowledge in a Neural Network, 2015, Geoffrey Hinton, Oriol Vinyals, Jeff Dean]
  - original idea came from model compression
- both are vulnerable against high-power adversary

Why are modern classifiers vulnerable

- small margin due to overfitting or linear structures



Generative adversarial network

"There are many interesting recent development in deep learning. . .The most important one, in my opinion, is adversarial training (also called GAN for Generative Adversarial Networks). This, and the variations that are now being proposed is the most interesting idea in the last 10 years in ML, in my opinion." – Yann LeCun

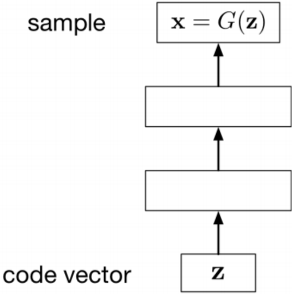
- discriminative model: given labelled samples $\{(X_i, Y_i)\}_{i=1}^n$ learn the conditional distribution $P(Y|X)$
- generative model: given unlabelled samples $\{X_i\}_{i=1}^n$ learn the distribution $P(X)$

Generative adversarial network

- Generative model
  - goal:
    1. sampling: given samples $X = \{X_1, \ldots, X_n\}$ find a model that generates samples that resembles X
    2. fit a parametric distribution to the data
    3. inference: compute likelihood
    4. inference: find latent variables (inference)
  - examples:
    - simple distributions - Gaussian, Bernoulli
    - mixture models
    - Boltzmann machines
    - variational autoencoders
- Generative Adversarial Networks mainly focus on goal 1.

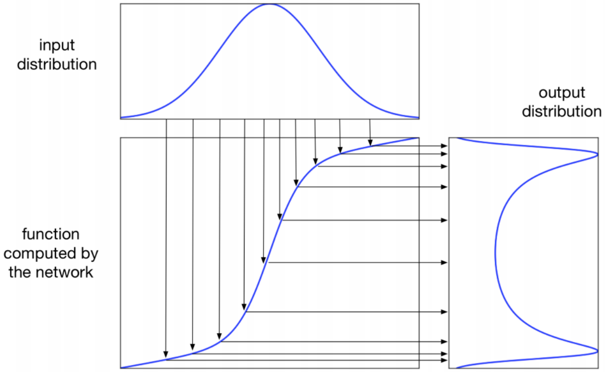
Generative Adversarial Networks

- a generative network implicitly encode a probability distribution via
- code vector Z from a simple fixed distribution (spherical Gaussian)
- and a network defining a mapping from Z to X, e.t. $X = G(Z)$ (and is differentiable)



Generative Adversarial Networks

- one dimensional example

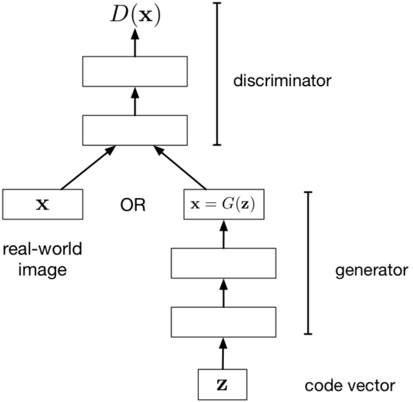


- the advantage of a generative network is that if we have a **appropriate loss** then one can train the generator via gradient descent

$$\mathbb{E}_Z[\ell(G_W(Z), \{X_j\}_{j=1}^n)] = \frac{1}{m}\sum_{i=1}^{m} \ell(G_W(Z_i), \{X_j\}_{j=1}^n)$$

Generative Adversarial Networks

- Generative adversarial network (GAN): innovative choice of the loss using two neural networks
  - **generator network**: produce realistic samples
  - **discriminator network**: figure out whether the sample came from the real data (training data) or the generator
  - the generator tries to fool the discriminator



Generative Adversarial Networks

- consider the discriminator with weights Θ, and denote the output by $D_\Theta(X_i) \simeq \mathbb{P}(X_i$ came from the training set $\mathcal{D})$
- the corresponding cross entropy loss for the discriminator to minimize is

$$\mathcal{L}_D(\Theta, W) \;=\; \frac{1}{n}\sum_{i=1}^{n}(-\log(D_\Theta(X_i))) + \frac{1}{m}\sum_{j=1}^{m}(-\log(1 - D_\Theta(G_W(Z_j))))$$

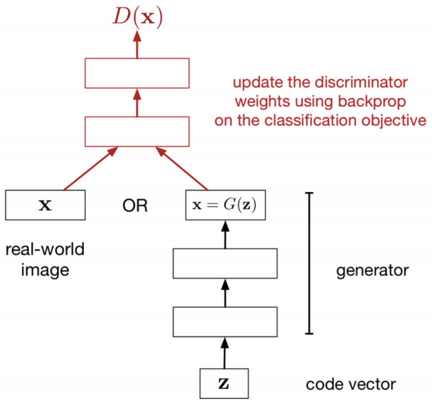
- the generator tries to fool the discriminator by maximizing the discriminator loss

$$\max_W \min_\Theta \quad \mathcal{L}_D(\Theta, W)$$

  this is a mini-max formulation of a zero sum game, and we try to find the optimal solution by iteratively fixing W and optimizing over Θ and then fixing Θ and optimizing over W

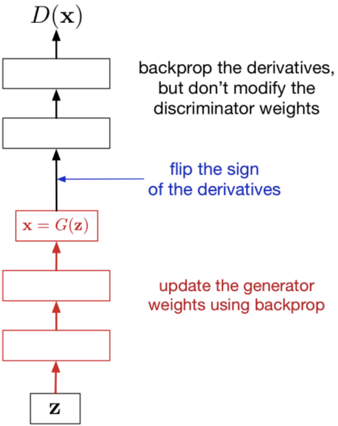
Generative Adversarial Networks

- Discriminator update is back-propagation on standard cross entropy loss for classification



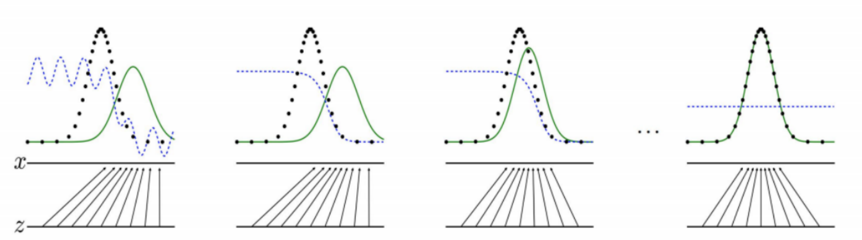
Generative Adversarial Networks

- Generator update is back-propagation on the discriminator with fixed Θ and over the example X
- which is further backpropagated to update generator weights W



Generative Adversarial Networks

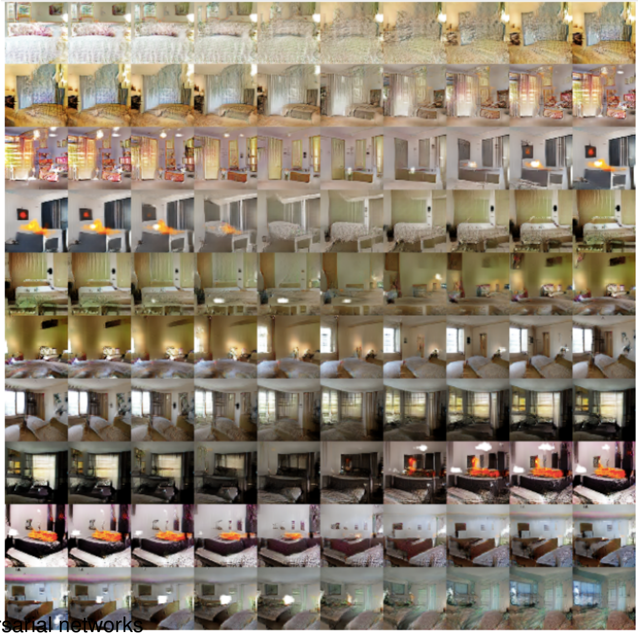
- evolution of the discriminator in blue and the generator in green, and training data in black



- Breakthrough: DCGAN ["Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", 11/2015, Alec Radford, Luke Metz, Soumith Chintala]

- generated images of bed rooms from LSUN dataset (3 million training data, $64 \times 64 \times 3$)
- evaluated via de-duplication with hashing (low collision)
- main contribution: use Convolutional Neural Network for discriminator and generator, with special guidelines

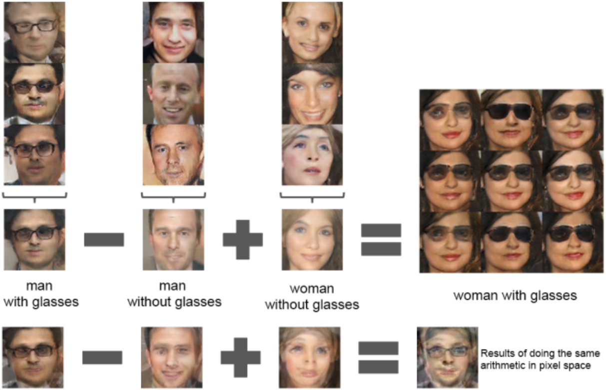
- 9 randomly chosen points in the latent space, and the interpolation shown in X space



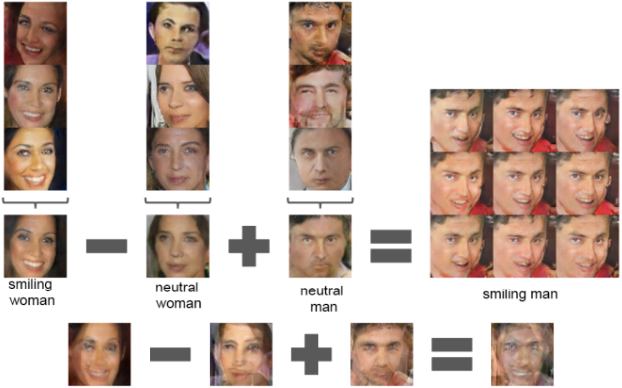
Generative Adversarial Networks

- evaluation: DBPEDIA dataset, 3M images filtered to 350,000 faces from 10K people
- the transition is smooth, each resembling a bed room

- arithmetic over Z space for adding/subtracting features



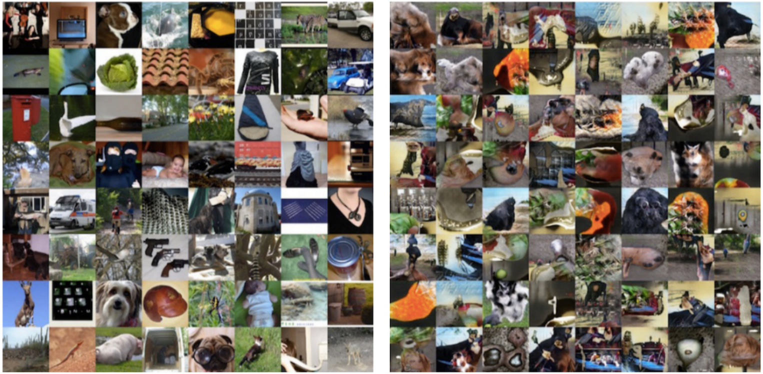
- adding a feature "glasses"



- adding a feature "look right"

- evaluation: Imagenet-1K dataset natural images, $32 \times 32 \times 3$
- achieve state-of-the-art in downstream classification tasks

ImageNet:



mixture of Bernoullis

RBM

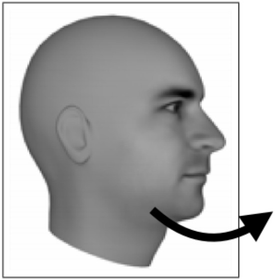
variational autoencoder

- Image to image translation with supervised examples:

- Image to image translation with supervised examples:

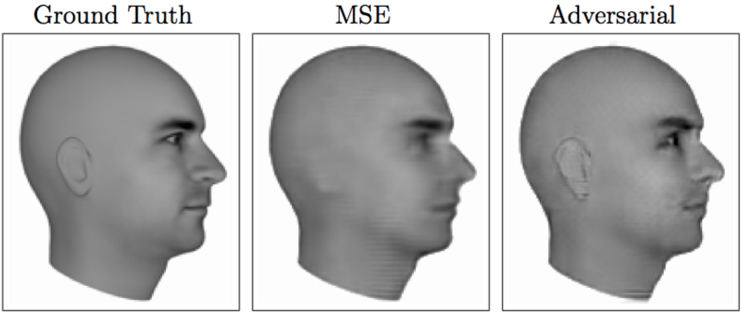
Next Video Frame Prediction

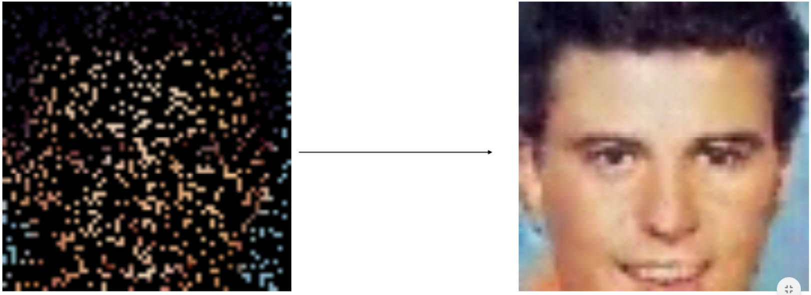
Ground Truth

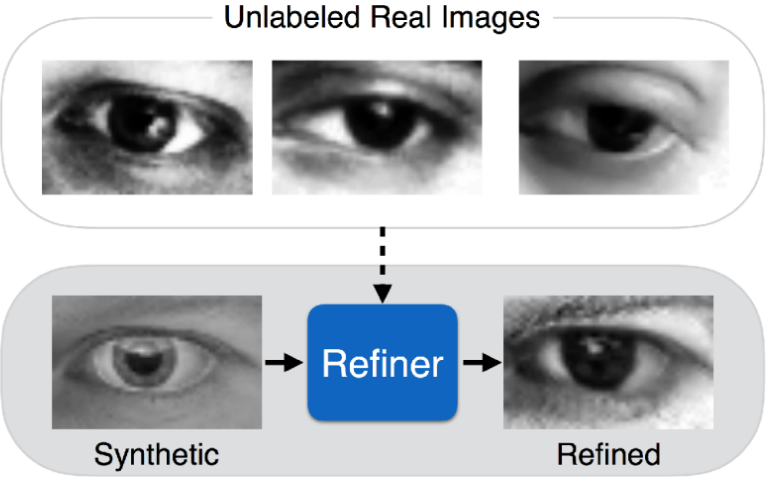


What happens next?

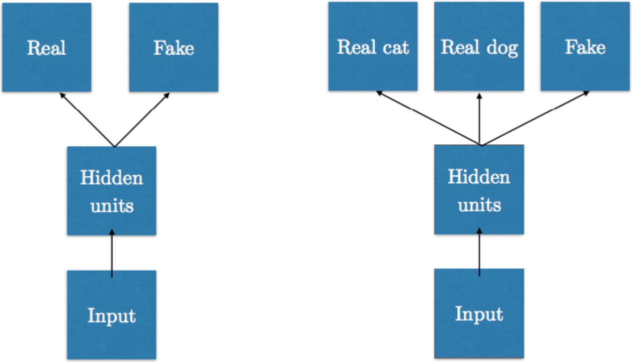
Next Video Frame Prediction





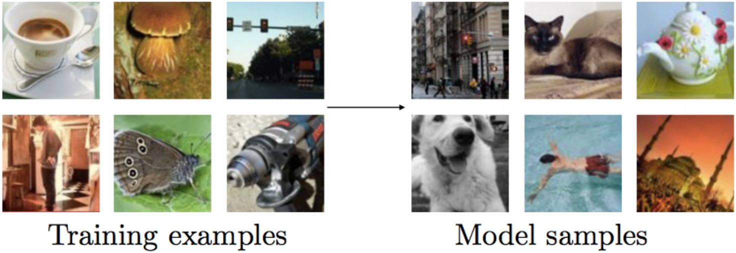

Unlabeled Real Images
Refiner
Synthetic
Refined

Supervised Discriminator



https://github.com/hindupuravinash/the-gan-zoo

- Challenges in GAN: visually realistic images
- [Improved Techniques for Training GANs, 6/2016 Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen]
- ultimate goal is

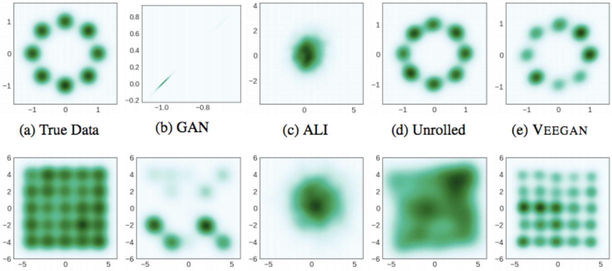


- Two challenges in GAN
  - mode collapse
    - folklore: the discriminator is constant in a input region, then the generator cannot replicate specific types of samples (no evidence)
  - unstable training [Improved Techniques]

  - Q. does mode collapse cause instability? [Veegan]
  - Q. does collapsed modes change over iterations?
  - Q. does solving mode collapse make GAN training more stable?
  - Q. how do we solve mode collapse?
  - Q. is packing compatible with all of them?

Mode collapse

- mode collapse observed in GANs



(a) True Data (b) GAN (c) ALI (d) Unrolled (e) VEEGAN

(f) True Data (g) GAN (h) ALI (i) Unrolled (j) VEEGAN

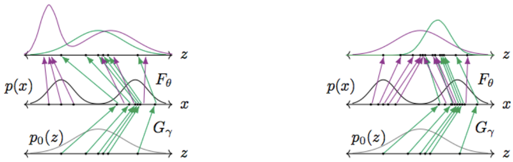
Mode collapse: new architectures - inverse network

- [Veegan: reducing mode collapse in GANs using implicit variational learning, 2017 NIPS]
  - insight: train inverse network $F_V(X) \simeq Z$ to give extra information on the joint (X, Z)

$$\mathcal{L}_D(\Theta, W, V) \quad = \quad \frac{1}{n}\sum_{i=1}^{n}(-\log(D_\Theta(X_i, F_V(X_i)))) + \frac{1}{m}\sum_{j=1}^{m}(-\log(1 - D_\Theta(G_W(Z_j), Z_j)))$$

$$\max_{W,V} \min_{\Theta} \quad \mathcal{L}_D(\Theta, W, V) - \frac{1}{m}\sum_{j=1}^{m} ||Z_j - F_V(G_X(Z_j))||^2$$

  - criticism: if sample in X is not enough for discriminator, why should the joint sample (X, Z) be better?
  - criticism: the regularization is claimed to provide gradient even when D_W is constant, which is not clear.



(a) Suppose F_θ is trained to approximately invert

(b) When F_θ is trained to map the data to a Gaus-

Mode collapse: new architectures

- [BiGAN]
  -

Mode collapse: new architectures - inverse network

- [Adversarially Learned Inference (ALI)]
  -

Mode collapse: new architectures - inverse network

- [Likelihood-free variational inference (LFVI)]
    -

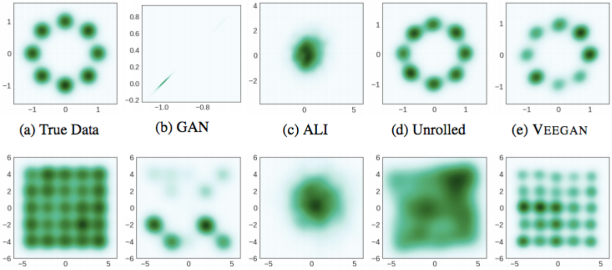
Mode collapse: new architectures

- [OMIE:On-line mutual information estimator, 2018 ICLR submission]
  -

Mode collapse: new architectures

- Quantitative comparisons
- proposed metrics:
  - high quality samples %: within x-std of nearest mode (x is three or ten)
  - modes captured: at least one high quality sample exists

	2D ring Modes (Max 8)	% high quality samples	2d Grid Modes (Max 25)	% high quality samples	1200D Synthetic Modes (Max 10)	% high quality samples
GAN []	1.0	99.30	3.3	0.5	1.6	2.00
ALI []	2.8	0.13	15.8	1.6	3.0	5.40
Unrolled GAN []	7.6	35.60	23.6	16.0	0.0	0.00
Veegan []	8.0	52.90	24.6	40.0	5.5	28.29

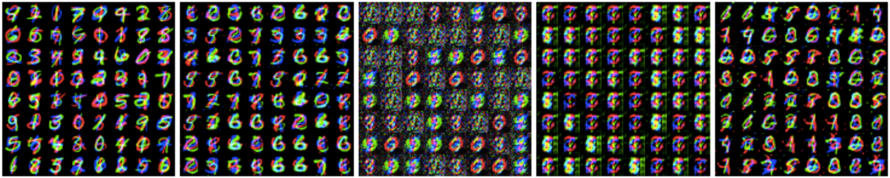


(a) True Data (b) GAN (c) ALI (d) Unrolled (e) VEEGAN

(f) True Data (g) GAN (h) ALI (i) Unrolled (j) VEEGAN

- metrics:
  - KL: KL divergence computed as in unrolled GAN paper
  - modes captured: use trained classifier
  - IvOM: for each real image, generate the closest image and report MSE

	Stacked MNIST Modes (Max 1000)	KL	CIFAR-10 IvOM
DCGAN []	99.0	3.40	0.00844 ± 0.002
ALI []	16.0	5.40	0.0067 ± 0.004
Unrolled GAN []	48.7	4.32	0.013 ± 0.0009
Veegan []	150	2.95	0.0068 ± 0.0001



(a) True Data (b) DCGAN (c) ALI (d) Unrolled (e) VEEGAN

(a) Generated samples nearest to real images from CIFAR-10. In each of the two panels, the first column are real images, followed