BLG 561E Deep Learning
Lecture 10: Implicit Generative Deep Learning Models: GANs

Prof. Gozde Unal
Istanbul Technical University
Fall 2021

1

GANs

GANs: Algorithms that map white noise to random high-dimensional objects with structure:

$z \sim N(0, I_{100\times100}) \longrightarrow$ face GAN $\longrightarrow$

Trained using samples (e.g. faces) from true high-dimensional distribution with structure (e.g. natural face images)

Loss Function for vanilla (standard) GAN:

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

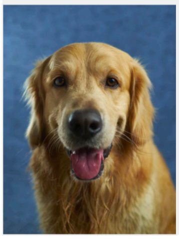
GANs, Goodfellow et al 2014

2

Generative adversarial networks

Discriminator

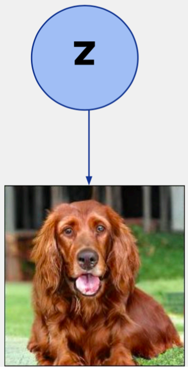
Learns to distinguish between real and generated data.



vs

Generator

Learns to generate data to “fool” the discriminator.



Deepmind DL Lectures

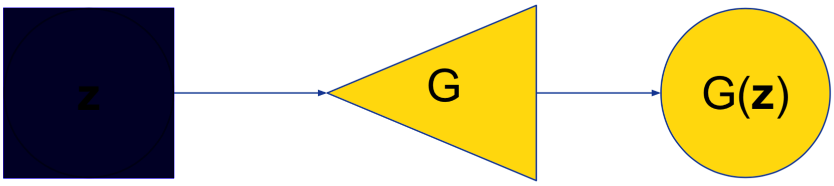
3

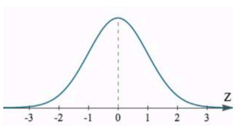
Generator

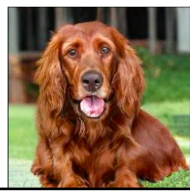
latent (“noise”) vector
z ~ P(**z**)

generator G:
a deep neural network

generated data
G(**z**)

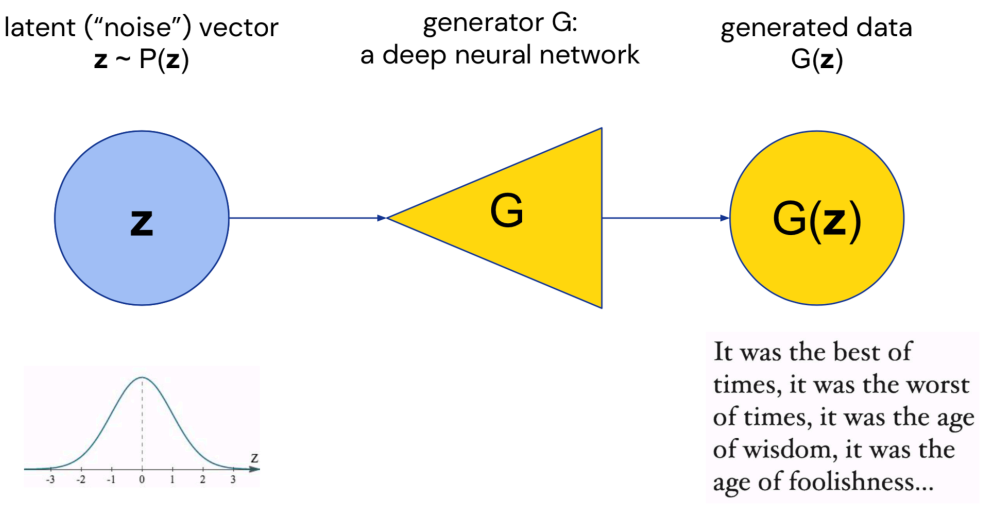






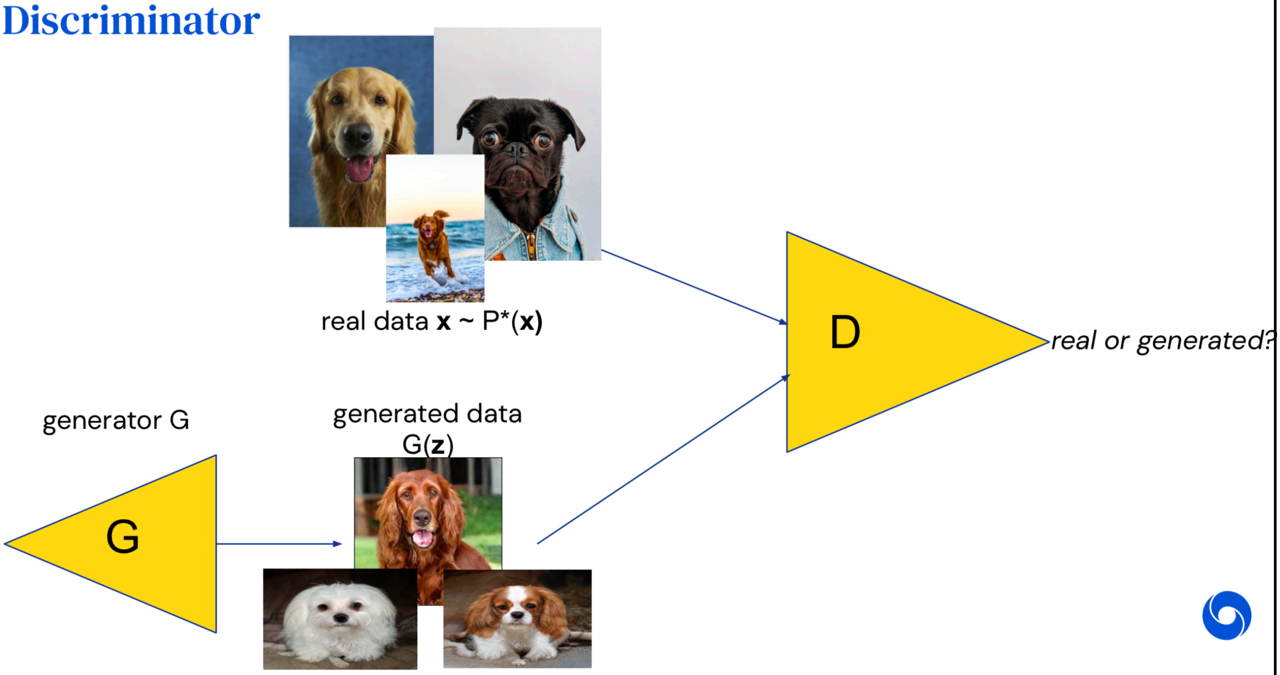
4

Generator



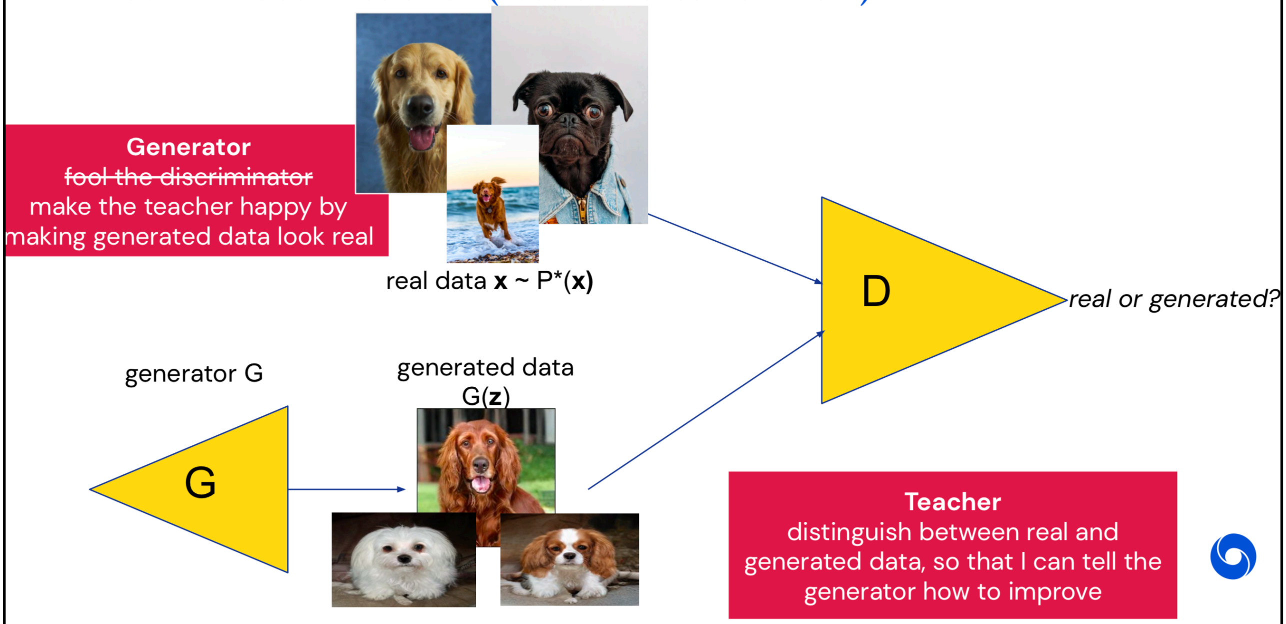
5

Discriminator



6

~~Discriminator~~ Teacher (less adversarial view)



7

Generative adversarial networks

Want to learn more?

Goodfellow, et al. **Generative adversarial networks.** Neural Information Processing Systems (**2014**)

$$\min_G \max_D V(D,G) = \underbrace{\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})]}_{\text{log-probability that D correctly predicts real data } \mathbf{x} \text{ are real}} + \underbrace{\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]}_{\text{log-probability that D correctly predicts generated data G}(\mathbf{z}) \text{ are generated}}$$

discriminator's (D) goal: **maximize** prediction accuracy

generator's (G) goal: **minimize** D's prediction accuracy, by **fooling** D into believing its outputs G(**z**) are real as often as possible

8

Generative adversarial networks

Want to learn more?

Goodfellow, et al. **Generative adversarial networks.** Neural Information Processing Systems (2014)

for number of training iterations **do**
  for k steps **do**
  - Sample minibatch of m noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
  - Sample minibatch of m examples $\{x^{(1)}, \dots, x^{(m)}\}$ from data generating distribution $p_{\text{data}}(x)$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(x^{(i)}\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right) \right].$$

  end for
  - Sample minibatch of m noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right).$$

end for

Algorithm from Goodfellow et al. (2014)

9

Generative adversarial networks as zero sum game

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

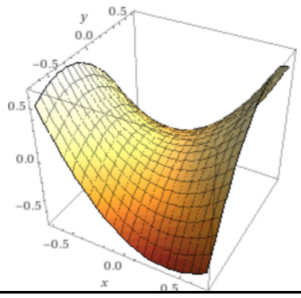
10

Min-Max theorem

- **[von Neumann 1928]:** If $X \subset \mathbb{R}^n, Y \subset \mathbb{R}^m$ are compact and convex, and $f: X \times Y \to \mathbb{R}$ is convex-concave (i.e. $f(x,y)$ is convex in x for all y and is concave in y for all x), then

$$\min_{x \in X} \max_{y \in Y} f(x,y) = \max_{y \in Y} \min_{x \in X} f(x,y)$$

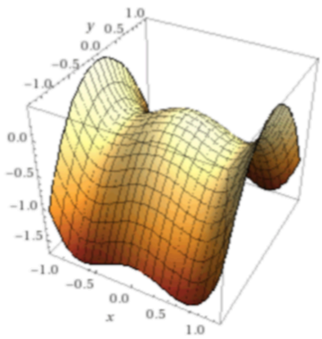
- Min-max optimal point (x,y) is essentially unique (unique if f is strictly convex-concave, o.w. a convex set of solutions); value always unique
- E.g. $f(x,y) = x^2 - y^2 + x \cdot y$

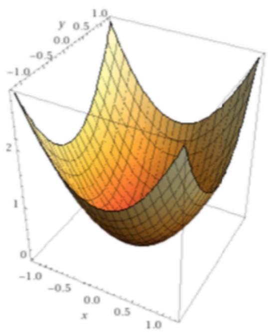


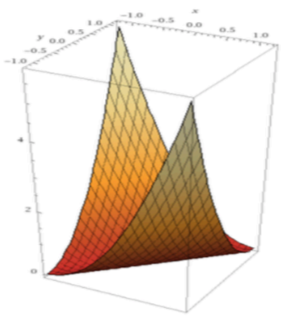
11

Min-Max theorem

- If f is not convex concave
  - $f(x,y) = x^4 - x^2 - y^2$ $\quad f(x,y) = x^2 + y^2$ $\quad f(x,y) = (x-y)^2$







12

GANs

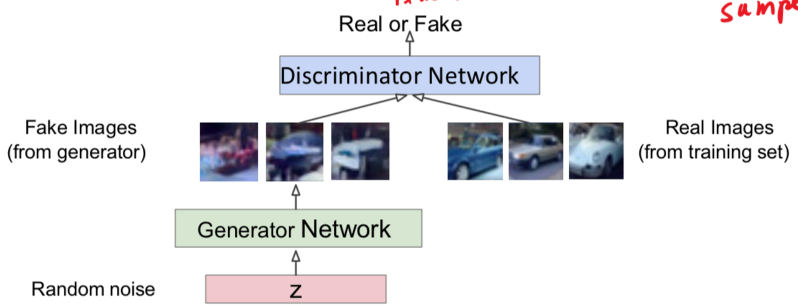
- A ***game*** between a *Generator* deep NN, w/ parameters θ_g, and a *Discriminator* deep NN, w/ parameters θ_d:

$$\inf_{\theta_g}\sup_{\theta_d}\left(\mathbb{E}_{X\sim F}[D_{\theta_d}(X)] - \mathbb{E}_{Z\sim N(0,I)}\left[D_{\theta_d}\left(G_{\theta_g}(z)\right)\right]\right)$$

Loss for Wasserstein GAN, Arjovski 2017

sample from real F

halucinated sample



- **Training**: generator and discriminator run gradient descent and ascent respectively to update their parameters θ_g, θ_d; expectations are approximated by finite sample averages
- even ignoring expectation approximation errors, will paired gradient descent/ascent dynamics converge? to what?

Picture: Costis Daskalakis

13

GANs

QUESTION: Is the GAN objective function convex-concave?

Answer: No!

What can be done?

1. study local saddles (don't necessarily exist), e.g. [Daskalakis-Panageas NeurIPS'18]
2. study local min of max $f(\cdot, y)$ function, e.g. [Jin-Netrapali-Jordan'19]
3. Go ahead and train your generator and discriminator **to find a low value for the loss function**
4. Evaluate your performance

Note: Your algorithm may fail to converge

14

Many variants of GANs

Early methods: architectural changes are advised, for instance DCGAN:

Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

DCGAN Unsupervised Representation Learning with Deep Convolutional GANs, Radford et al ICLR 2016

15

Other divergences and distances

Want to learn more?

Arjovsky, et al **Wasserstein GAN.** International Conference on Machine Learning (2017)

Wasserstein Distance

$$W(p, p^*) = \sup_{||f||_L \leq 1} \mathbb{E}_{p(x)} f(x) - \mathbb{E}_{p^*(x)} f(x)$$

Wasserstein GAN

$$\min_G \max_{||D||_L \leq 1} \mathbb{E}_{p^*(x)} D(x) - \mathbb{E}_{p(z)} D(G(z))$$

Try to make D is 1-Lipschitz via gradient penalties, spectral normalization, weight clipping.

16

Wasserstein GAN

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

Require: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.
Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

while θ has not converged **do**
  for $t = 0, ..., n_{\text{critic}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^m \sim \mathbb{P}_r$ a batch from the real data.
    Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
    $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z^{(i)}))\right]$
    $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
    $w \leftarrow \text{clip}(w, -c, c)$
  end for
  Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
  $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
end while

Wasserstein GANs, Arjovski et al 2017

17

Common Failure Modes of GANs

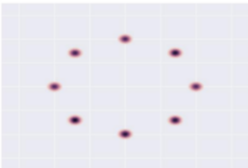
GANs have a number of common failure modes

All of these common problems are areas of active research

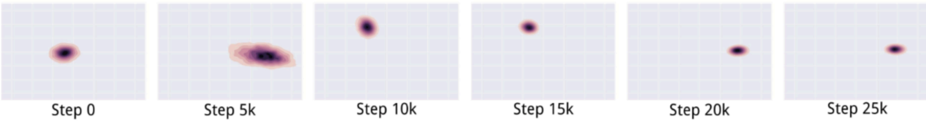
18

Oscillations in GAN training

Oscilations: e.g. Gaussian mixture



True Distribution: Mixture of 8 Gaussians on a circle



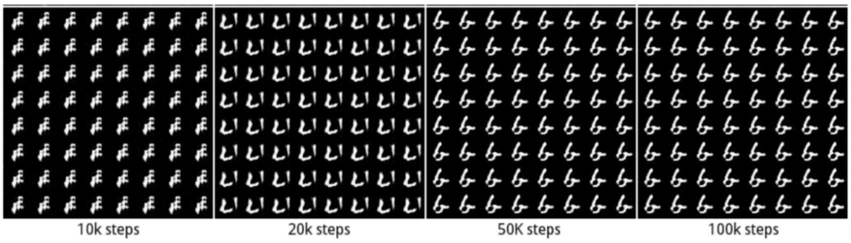
Output Distribution of vanilla GAN, trained via gradient descent/ascent dynamics: cycling thru modes at different steps of training

Figure from: Unrolled Generative Adversarial Networks, Metz ICLR 2017

19

Oscillations in GAN training

True Distribution: MNIST



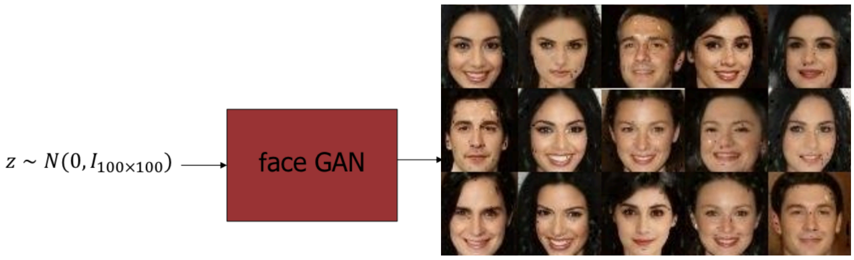
Output Distribution of vanilla GAN, trained via gradient descent/ascent dynamics: cycling thru “proto-digits” at different steps of training

Unrolled Generative Adversarial Networks, Metz ICLR 2017

20

Mode Collapse Problem in GANs

Normally, you want your GAN to produce a wide variety of outputs. You want, for example, a different face for every random input to your face generator.



Problem: Generator characterizes only a few images to keep fooling the discriminator in GAN, i.e. the input latent space is mapped to only a few points in the image space.

21

Mode Collapse Problem in GANs

Figure 2: Density plots of the true data and generator distributions from different GAN methods trained on mixtures of Gaussians arranged in a ring (top) or a grid (bottom).

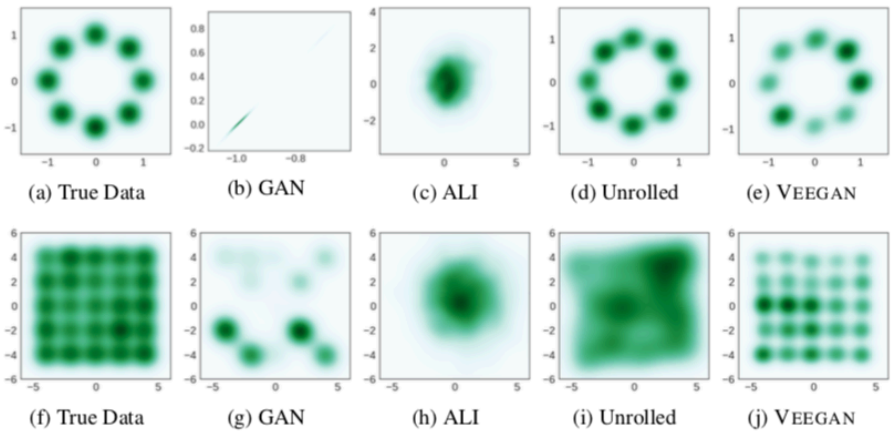


Figure: VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning, Srivastava 2017

22

GAN Variants: VeeGAN

An idea to address mode collapse

loss. More precisely, we minimize a loss function, like ℓ_2 loss between $z \sim p_0$ and $F_\theta(G_\gamma(z)))$. To quantify whether F_θ maps the true data distribution to a Gaussian, we use the cross entropy $H(Z, F_\theta(X))$ between Z and $F_\theta(x)$. This boils down to learning γ and θ by minimising the sum of these two objectives, namely

$$\mathcal{O}_{\text{entropy}}(\gamma, \theta) = E\left[\|z - F_\theta(G_\gamma(z))\|_2^2\right] + H(Z, F_\theta(X)). \quad (1)$$

While this objective captures the main idea of our paper, it cannot be easily computed and minimised. We next transform it into a computable version and derive theoretical guarantees.

The main idea of VEEGAN: introduce a second network Fθ, the *reconstructor network,* which is learned both to map the true data distribution p(x) to a Gaussian and to approximately invert the generator network.

VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning, Srivastava 2017

23

Catastrophic Forgetting problem of GAN

When a **GAN** suffers from catastrophic forgetting in its Discriminator, it exhibits undesired behaviors such as mode collapse and non-convergence.

24

AdaGAN

* Accepting that the GAN will cover only a subset of the modes in the dataset, train multiple GANs to cover different modes

* Use multiple generative models combined into a mixture

Q: what is the main downside of this approach?

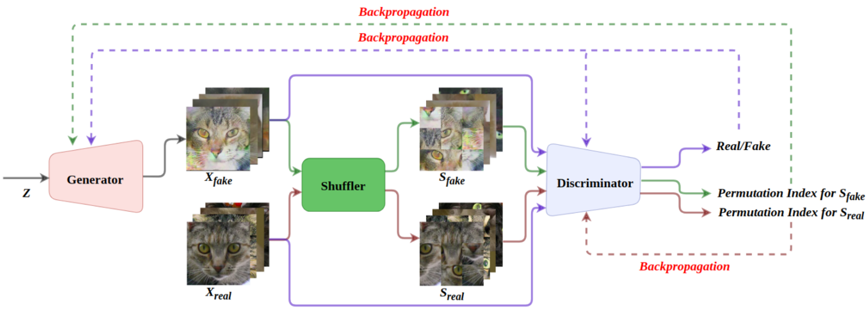
Algorithm 1: AdaGAN, a meta-algorithm to construct a "strong" mixture of T individual GANs, trained sequentially. The mixture weight schedule ChooseMixtureWeight and the training set reweighting schedule UpdateTrainingWeights should be provided by the user. Section 3 gives a complete instance of this family.

Input: Training sample $S_N := \{X_1, \dots, X_N\}$.
Output: Mixture generative model $G = G_T$.
Train vanilla GAN:
$W_1 = (1/N, \dots, 1/N)$
$G_1 = \mathrm{GAN}(S_N, W_1)$
for $t = 2, \dots, T$ **do**
#Choose a mixture weight for the next component
$\beta_t = \mathrm{ChooseMixtureWeight}(t)$
#Update weights of training examples
$W_t = \mathrm{UpdateTrainingWeights}(G_{t-1}, S_N, \beta_t)$
#Train t-th "weak" component generator G_t^c
$G_t^c = \mathrm{GAN}(S_N, W_t)$
#Update the overall generative model
#Notation below means forming a mixture of G_{t-1} *and* G_t^c.
$G_t = (1 - \beta_t)G_{t-1} + \beta_t G_t^c$
end for

AdaGAN: Boosting Generative Models, Tolstikhin et al 2017.

25

DeshuffleGAN: A Self-Supervised GAN: an approach to combat catastrophic forgetting



We deployed a commonly used self-supervised learning pretext of jigsaw puzzle solving as an auxiliary task for the Discriminator. We improved the structural representation quality of the generated images which are better to capture the original distribution compared to the baseline results.

Gulcin Baykal, Furkan Ozcelik, Gozde Unal. **Exploring DeshuffleGANs in Self-Supervised Generative Adversarial Networks. Pattern Recognition**, Volume 122, 2022. 108244, ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2021.108244.

26

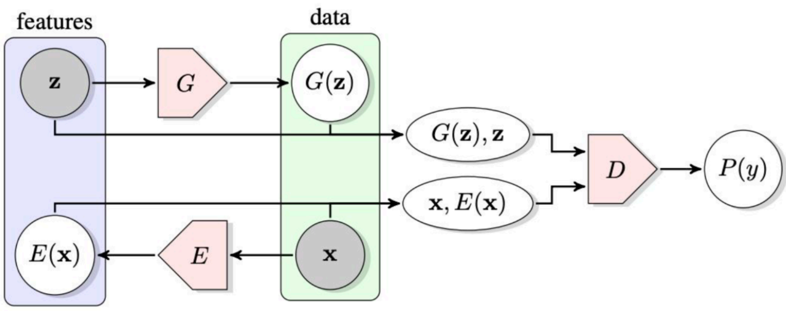
ALI / Bidirectional GANs (Dumoulin et al., Donahue et al.)

Want to learn more?

Dumoulin, et al. **Adversarially Learned Inference.** International Conference on Learning Representations (2017)

Donahue, et al. **Adversarial Feature Learning.** International Conference on Learning Representations (2017)

- Adversarial approach to feature representation learning and inference
- Adds an **encoder** network (E) which learns the inverse mapping from G, mapping from data **x** to latents **z**
- The **joint discriminator** sees tuples (**x**, **z**)



27

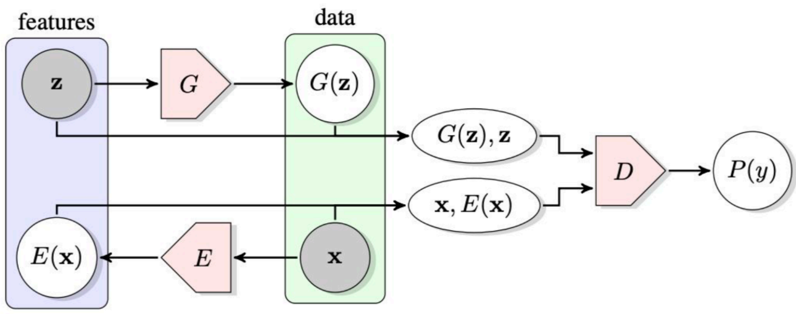
ALI / Bidirectional GANs (Dumoulin et al., Donahue et al.)

Want to learn more?

Dumoulin, et al. **Adversarially Learned Inference.** International Conference on Learning Representations (2017)

Donahue, et al. **Adversarial Feature Learning.** International Conference on Learning Representations (2017)

- The **joint discriminator** sees tuples (**x**, **z**)
  - $\mathbf{z} \sim P_z$, $\mathbf{x} = G(\mathbf{z})$
  - $\mathbf{x} \sim P_x$, $\mathbf{z} = E(\mathbf{x})$
- In the global optimum, E and G are inverses; for all **x** and **z** we have
  - $\mathbf{x} = G(E(\mathbf{x}))$
  - $\mathbf{z} = E(G(\mathbf{z}))$



28

ALI / Bidirectional GANs (Dumoulin et al., Donahue et al.)

Want to learn more?

Dumoulin, et al. **Adversarially Learned Inference.** International Conference on Learning Representations (2017)

Donahue, et al. **Adversarial Feature Learning.** International Conference on Learning Representations (2017)

- In the global optimum, E and G are inverses; for all **x** and **z** we have
  - **x** = G(E(**x**))
  - **z** = E(G(**z**))
- In practice, this inversion property does not hold perfectly
  - But reconstructions still often capture interesting semantics

(a) CelebA samples. (b) CelebA reconstructions.

29

BigBiGANs (Donahue et al.)

Want to learn more?

Donahue, et al. **Large Scale Adversarial Representation Learning.** Neural Information Processing Systems (2019)

- BiGANs at scale: **BigBiGANs** are BiGANs trained using the BigGAN G and D architectures
- ResNet-style encoders E
- Reconstructions exhibit clear high-level semantics of the input images (despite being unsupervised), while clearly not being memorised copies

real data **x** (128x128)

BigBiGAN reconstructions G(E(**x**))

30

BigBiGAN

Large Scale Adversarial Representation Learning

Jeff Donahue
DeepMind
jeffdonahue@google.com

Karen Simonyan
DeepMind
simonyan@google.com

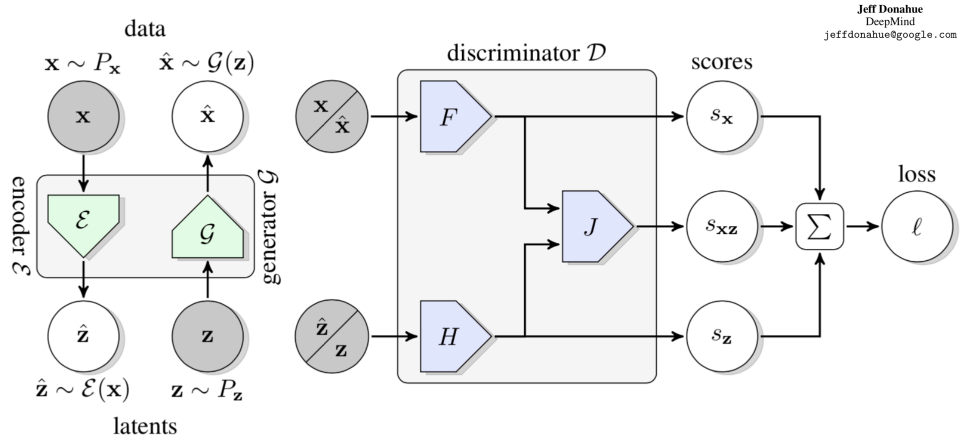


Figure 1: The structure of the BigBiGAN framework. The joint discriminator $\mathcal{D}$ is used to compute the loss ℓ. Its inputs are data-latent pairs, either $(\mathbf{x} \sim P_{\mathbf{x}}, \hat{\mathbf{z}} \sim \mathcal{E}(\mathbf{x}))$, sampled from the data distribution $P_{\mathbf{x}}$ and encoder $\mathcal{E}$ outputs, or $(\hat{\mathbf{x}} \sim \mathcal{G}(\mathbf{z}), \mathbf{z} \sim P_{\mathbf{z}})$, sampled from the generator $\mathcal{G}$ outputs and the latent distribution $P_{\mathbf{z}}$. The loss ℓ includes the unary data term $s_{\mathbf{x}}$ and the unary latent term $s_{\mathbf{z}}$, as well as the joint term $s_{\mathbf{xz}}$ which ties the data and latent distributions.

31

BigBiGANs (Donahue et al.)

Want to learn more?

Donahue, et al. **Large Scale Adversarial Representation Learning.** Neural Information Processing Systems **(2019)**

- BigBiGAN encoder learns ImageNet representations competitive with other unsupervised / self-supervised approaches

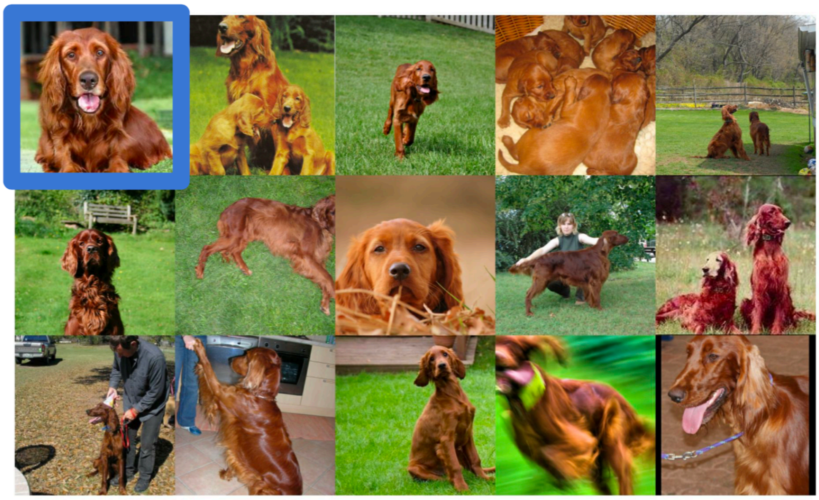
- Nearest neighbors (right) in BigBiGAN encoder feature space show the semantics present in the learnt representations

Nearest neighbors in BigBiGAN Encoder feature space

32

Checking overfitting: Nearest neighbours

Nearest neighbors: most similar (in feature space of a pretrained ImageNet classifier) images in the dataset.



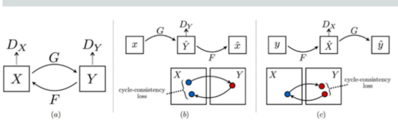
33

CycleGAN (Zhu et al.)

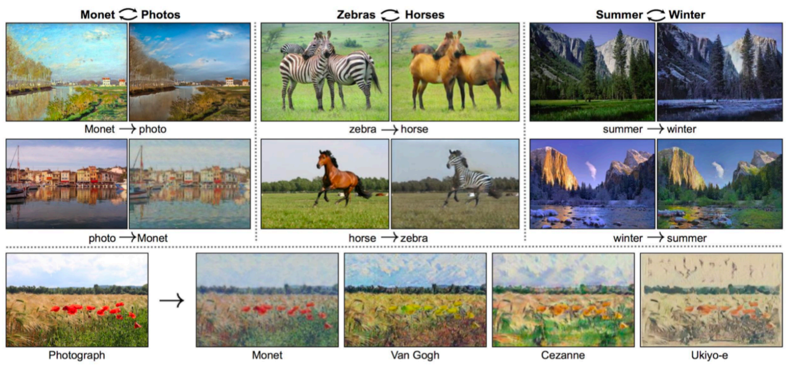
Want to learn more?

Zhu, et al. **Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks.** International Conference on Computer Vision (2017)

- Train a generator to **translate** between images of two different domains
- But **without any paired samples**!



- Enforces **cycle consistency**:
  - Image **x** in domain A
  - Translate to domain B
  - Back to domain A -> **x'**
  - Enforce **x** ≈ **x'**



34

Conditional GANs (cGANs)

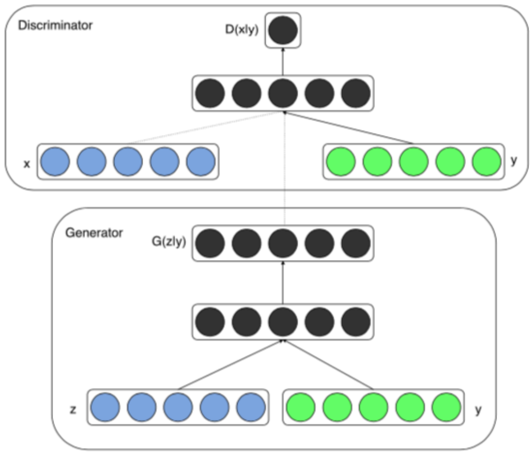


Figure 1: Conditional adversarial net

Helps prevent mode collapse

A conventional way of feeding y to D: to concatenate the vector y to the feature vector x, either at the input layer as in (Mirza-Osindero), or at some hidden layer (Reed et al.)

Figure: Mirza, Osindero, Conditional Generative Adversarial Nets, 2014

35

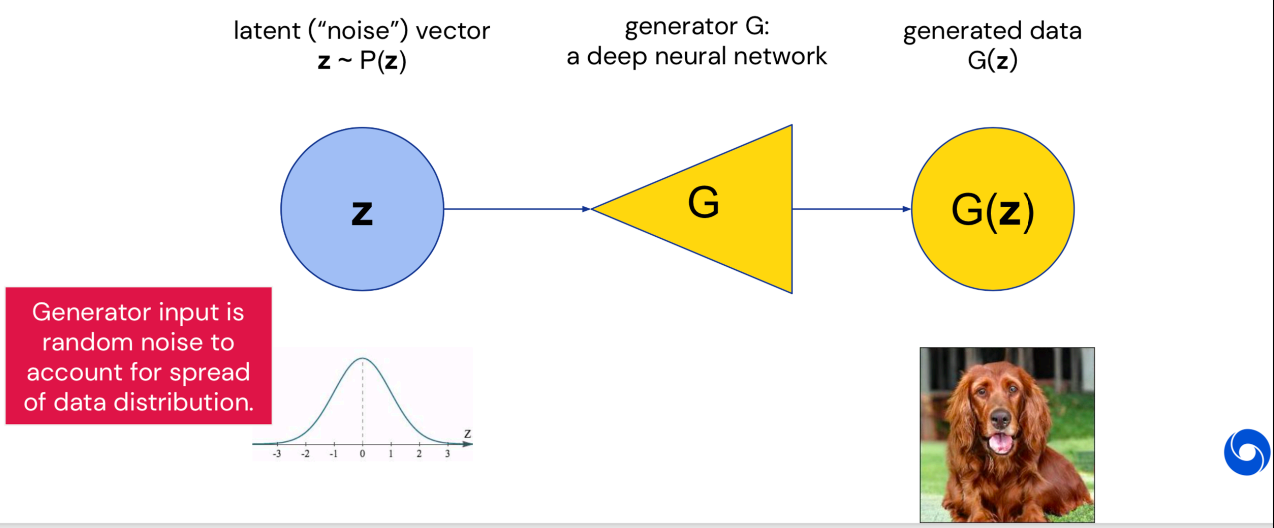
Unconditional vs Conditional GANs

provides a sample from the data distribution, but the user has no control over what kind of sample.

1. We can specify what sample we want, e.g. A cat
2. Input image can be provided.

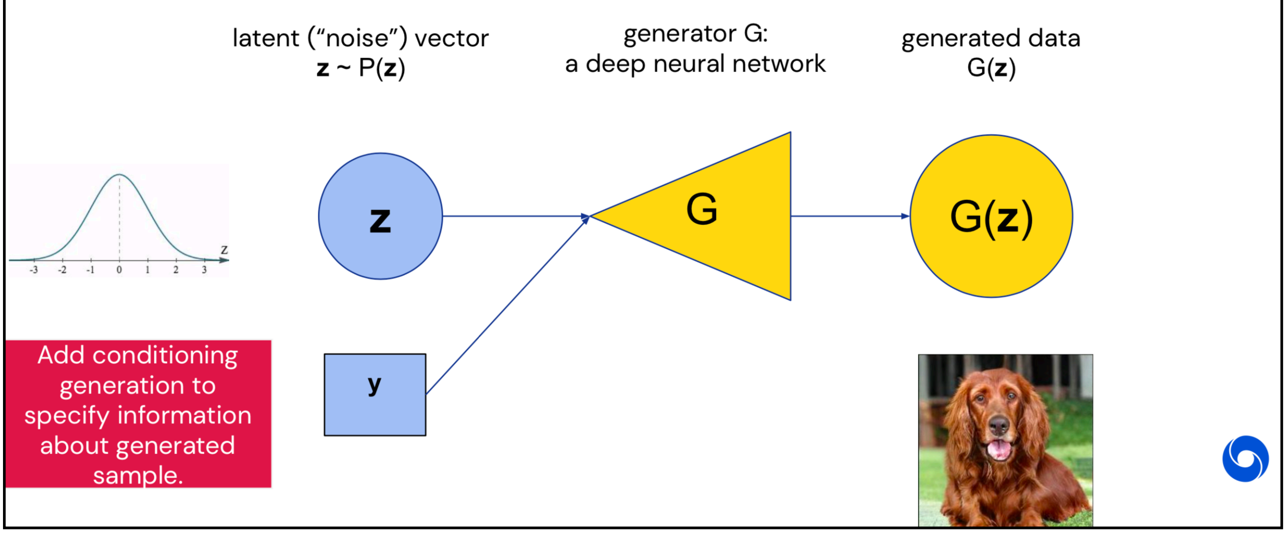
36

So far… unsupervised GANs



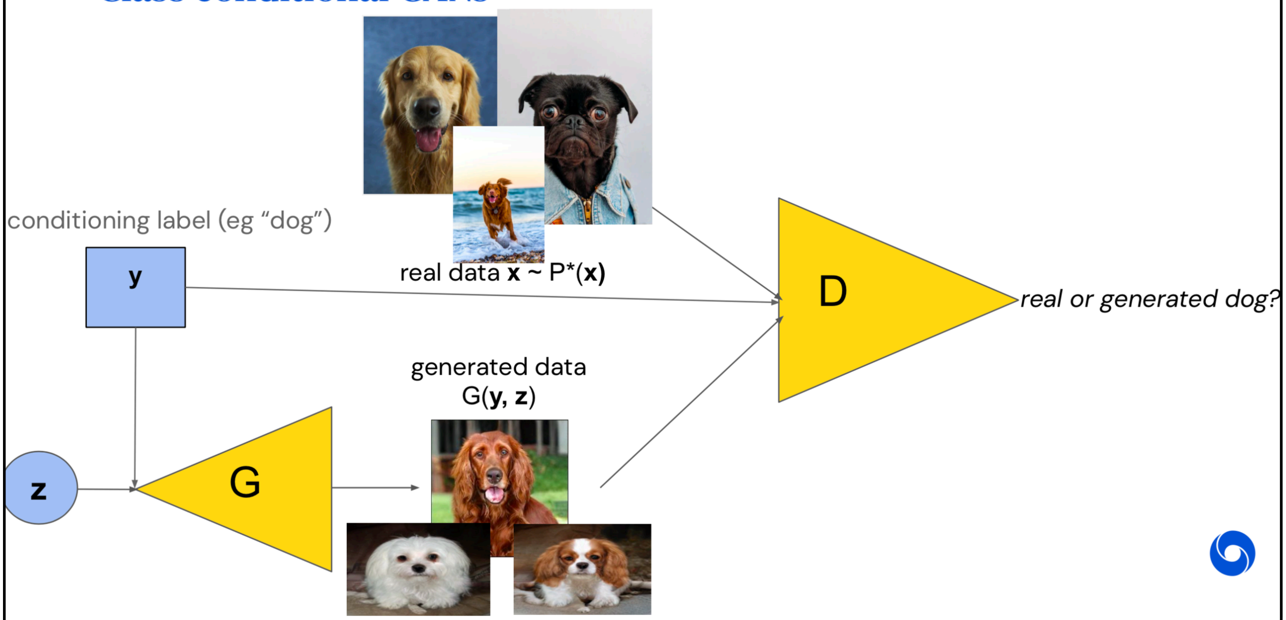
37

Conditioning information for training GANs



38

Class conditional GANs



39

Conditional GANs (cGANs)

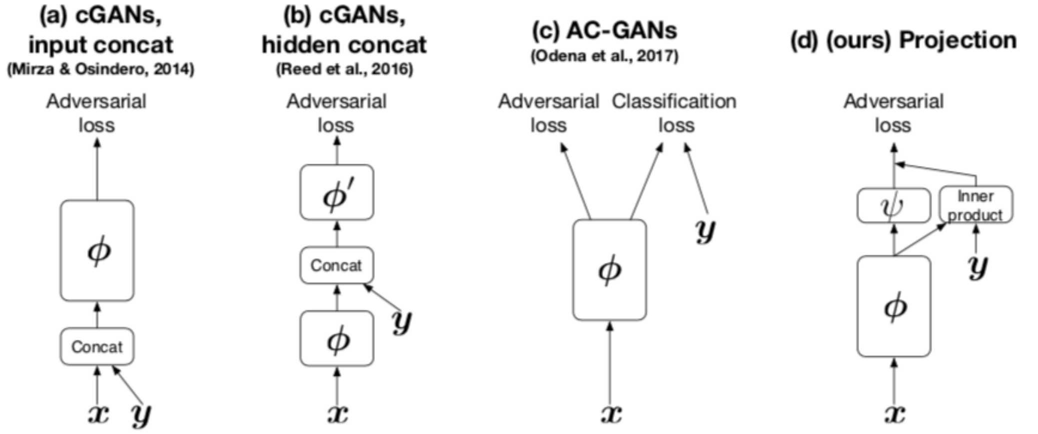


Figure 1: Discriminator models for conditional GANs

$$f(\boldsymbol{x}, \boldsymbol{y}) := \boldsymbol{y}^{\mathrm{T}} V \phi(\boldsymbol{x}) + \psi(\phi(\boldsymbol{x}))$$

Figure: cGANs with Projection Discriminator, Miyato and Koyama ICLR 2018

Typically: use y to denote a one-hot vector of the label y

40

Performance Evaluation

How to evaluate Generated Image Quality?

- No evaluation metric is able to capture all desired properties: sample quality, generalization, representation learning
- GANs are implicit models, there is no explicit log likelihood to evaluate

Some Ideas:

- Have humans evaluate the generation quality !
- Use synthetic/toy datasets such as mixture of Gaussians to see whether your model captures all modes

41

Inception Score

Want to learn more?

Salimans, et al **Improved techniques for training GANs**
Neural Information Processing Systems **(2016)**

Use a pretrained Imagenet classifier to compare (via KL divergence)
the distribution of labels obtained from the data
the distribution of labels coming from samples

Measures:
- sample quality
- dropping classes (no cats)
- correlates with human evaluation
- does not measure differences beyond class labels
- requires pretrained classifier

Higher is better.

42

Frechet Inception Distance

Want to learn more?

Heusel, et al **GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium**

Neural Information Processing Systems (2017)

Use a pretrained Imagenet classifier to compare (via Frechet distance)
the distribution of layer features obtained from the data
the distribution of layer features coming from samples

Measures:
- sample quality
- dropping classes (no cats)
- captures feature level statistics (not just classes)
- correlates with human evaluation
- requires pretrained classifier
- biased for a small number of samples and KID for a fix (see Binkowski, et al., ICLR 2018)

Lower is better.

43

Performance Evaluation

…??? Open research problem!

Other ideas:

* Calculate a density estimator ?? and compare the generated data distribution to the training data distribution

With some of the Conditional GANs, more apparent ways: e.g compare to the paired images through reconstruction Loss: e.g. Pix2Pix

44

GANs: The most active current area of research in Deep Unsupervised Learning

Please read material I referred to in these slides, as well as others that may be of interest to your own work

THE END

45