CS-E4891 Deep Generative Models
Lecture 6: Generative Adversarial Networks

Lauri Juvela

Department of Information and Communications Engineering (DICE)
Aalto University

April 5, 2025

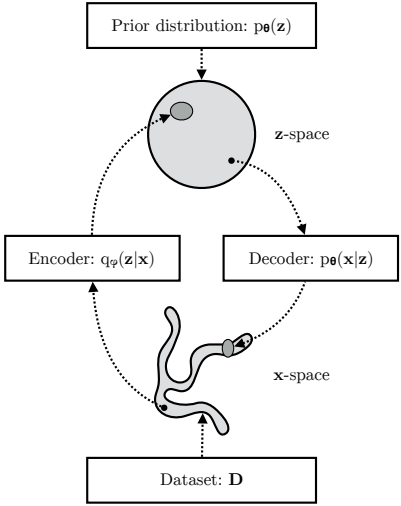
Outline

- Generative Adversarial Networks (GANs)
- Learning by comparison
- GAN loss functions
- GAN training process
- Theoretical properties: density ratios, quality, coverage
- Conditional GAN
- GAN architectures
- Cycle GAN

- Reading: Section 22 from Murphy, 2023; Chapter 15 from Prince 2024; Chapter 17 from Bishop, 2024)

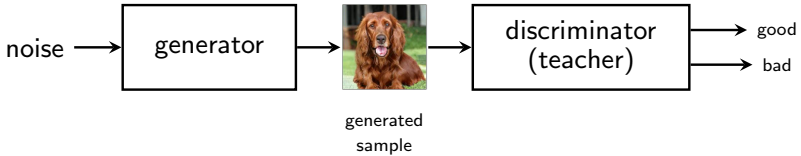
Variational autoencoders

- VAE reconstruction loss is defined in the pixel space, this is not a good metric perceptually
- Classifier neural networks are powerful for learning representations
- Could we use another neural net as a score function?



Generative Adversarial Networks (GANs) – Introduction

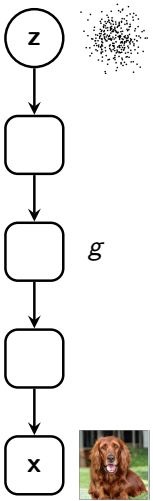
- GANs consist of two neural networks
- **Generator** transforms samples from a simple prior distribution to the data space
- **Discriminator** compares the generated samples to real data examples and scores them



- The Generator can be any parametric differentiable model (typically a deep neural network)
- Discriminator is a (deep neural network) classifier that classifies between real and generated samples

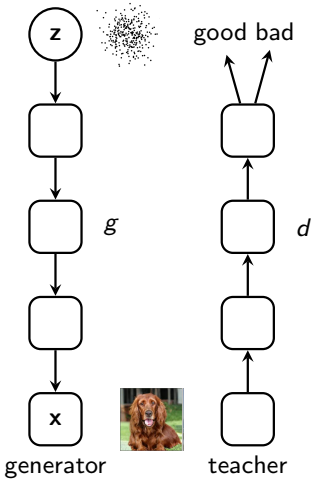
Generator

- Sample latent variable from isotropic Gaussian
  $\boldsymbol{z} \sim \mathcal{N}(0, I)$
- Transform z to the data space using a deep neural network
  $\boldsymbol{x} = G(\boldsymbol{z}; \boldsymbol{\theta})$
- Generator is an implicit generative model – model outputs generated samples but the explicit probability distribution is unknown



Generator and Discriminator

- Generator is guided by a teacher network (discriminator or critic) that evaluates the quality of generated samples
- How to train the teacher network $d(x; \phi)$?
- Classification task: classify training data examples as "good" and generator output samples as "bad"
- Generator attempts to improve its output by making its output more likely to classified as "good"



Learning by comparison

- GANs are implicit generative models
- Learning by comparison: sample from a generative models and compare generated samples to real data examples
- For more see, Murphy section 26.2

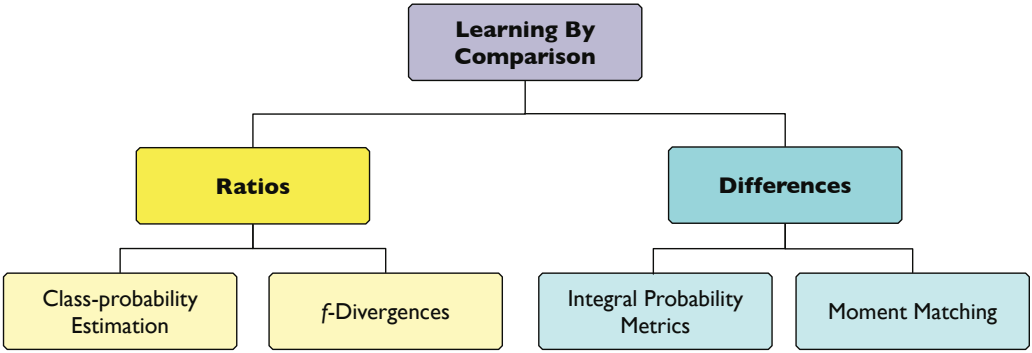


Figure: Overview of approaches for learning in implicit generative models (Murphy, Fig. 26.2)

GAN loss function

- Discriminator $D_{\boldsymbol{\phi}}(\boldsymbol{x}) \in [0, 1]$ is a neural network parameterised by $\boldsymbol{\phi}$
- Select the Bernoulli log-loss, also known as binary cross entropy loss, to obtain the "vanilla" GAN training objective for Discriminator

$$V(q_{\boldsymbol{\theta}}) = \arg\max_{\boldsymbol{\phi}} \mathbb{E}_{p(\boldsymbol{x}|y)p(y)} \left[y \log D_{\boldsymbol{\phi}}(\boldsymbol{x}) + (1-y) \log(1 - D_{\boldsymbol{\phi}}(\boldsymbol{x})) \right]$$

- Split the objective into two separate terms for real and fake classes

$$V(q_{\boldsymbol{\theta}}) = \arg\max_{\boldsymbol{\phi}} \left[\mathbb{E}_{p(\boldsymbol{x}|y=1)p(y=1)} \log D_{\boldsymbol{\phi}}(\boldsymbol{x}) + \mathbb{E}_{p(\boldsymbol{x}|y=0)p(y=0)} \log(1 - D_{\boldsymbol{\phi}}(\boldsymbol{x})) \right]$$

- Draw samples from real data distribution $p_{\text{data}}(\boldsymbol{x}) = p^*(\boldsymbol{x})$ and generated data distribution $p_G(\boldsymbol{x}) = q_{\boldsymbol{\theta}}(\boldsymbol{x})$

$$V(q_{\boldsymbol{\theta}}) = \arg\max_{\boldsymbol{\phi}} \left[\mathbb{E}_{p*(\boldsymbol{x})} \log D_{\boldsymbol{\phi}}(\boldsymbol{x}) + \mathbb{E}_{q_{\boldsymbol{\theta}}(x)} \log(1 - D_{\boldsymbol{\phi}}(\boldsymbol{x})) \right]$$

- More in Murphy, section 26.2

Non-saturating GAN loss

- Change the generator objective: instead of minimizing the probability of generator output classified as fake $\log(1 - D(G(z)))$, maximize the probability of generator output classified as real $-\log D(G(z))$
- The non-saturating loss provides stronger gradients when the discriminator is easily detecting that generated samples are fake.
- Generator loss as a function of discriminator score. (left)
- The gradients of the generator loss with respect to the discriminator score. (right)

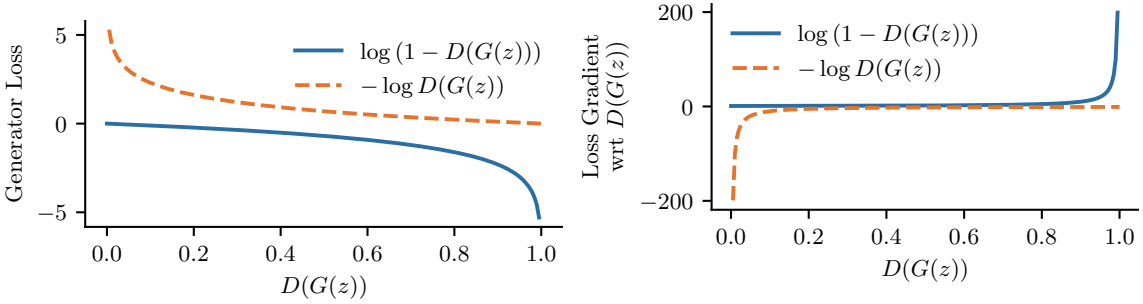


Figure: Murphy Fig. 26.5

GAN training procedure

1. Update the Discriminator
   - Sample N examples $\boldsymbol{x}_i$ from the training set
   - Generate N samples $G(\boldsymbol{z}_i)$
   - Compute the binary cross-entropy loss

$$\mathcal{L}_D = -\frac{1}{N}\sum_{i=1}^{N} \log D(\boldsymbol{x}_i) - \frac{1}{N}\sum_{i=1}^{N} \log\left(1 - D(G(\boldsymbol{z}_i))\right)$$

   - Update $\boldsymbol{\phi}$ by SGD (or Adam): $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} - \nabla_{\boldsymbol{\phi}} \mathcal{L}_D$
2. Update the Generator
   - Generate N samples $G(\boldsymbol{z}_i)$ using the Generator
   - Compute the (non-saturating) loss function

$$\mathcal{L}_G = -\frac{1}{N}\sum_{i=1}^{N} \log D(G(\boldsymbol{z}_i))$$

   - Update θ by SGD (or Adam): $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \nabla_{\boldsymbol{\theta}} \mathcal{L}_G$ (gradients flow through the discriminator)

Optimal discriminator (home exercise)

Write the expected values in the objective function in the integral form

$$V(G,D) = \int_x p^*(x) \log D(x) \mathrm{d}x + \int_z p(z) \log(1 - D(G(z))) \mathrm{d}z$$

Treat G as constant and apply "the law of unconscious statistician" to combine the expectations to a single integrand

$$= \int_x p^*(\boldsymbol{x}) \log D(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x}) \log(1 - D(\boldsymbol{x})) \mathrm{d}\boldsymbol{x}$$

Upper bound: the integral is less than or equal to an integral over the maximum of an integrand

$$V(G,D) \leq \int_x \max \left[p^*(\boldsymbol{x}) \log D(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x}) \log(1 - D(\boldsymbol{x})) \right] \mathrm{d}\boldsymbol{x}$$

Check the extreme values of the integrand, equality is fulfilled at

$$D^*(\boldsymbol{x}) = \frac{p^*(\boldsymbol{x})}{p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})}$$

This is the optimal discriminator

GANs implicitly minimise the Jensen-Shannon divergence

- For the optimal discriminator the GAN objective becomes

$$
\begin{aligned}
V^*(q_{\boldsymbol{\theta}}, p^*) &= \frac{1}{2}\mathbb{E}_{p^*(\boldsymbol{x})}\left[\log \frac{p^*(\boldsymbol{x})}{p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})}\right] + \frac{1}{2}\mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{x})}\left[\log\left(1 - \frac{p^*(\boldsymbol{x})}{p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})}\right)\right] \\
&= \frac{1}{2}\mathbb{E}_{p^*(\boldsymbol{x})}\left[\log \frac{p^*(\boldsymbol{x})}{\frac{p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})}{2}}\right] + \frac{1}{2}\mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{x})}\left[\log \frac{q_{\boldsymbol{\theta}}(\boldsymbol{x})}{\frac{p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})}{2}}\right] - \log 2 \\
&= \frac{1}{2} D_{\mathbb{KL}}\left(p^* \middle\| \frac{p^* + q_{\boldsymbol{\theta}}}{2}\right) + \frac{1}{2} D_{\mathbb{KL}}\left(q_{\boldsymbol{\theta}} \middle\| \frac{p^* + q_{\boldsymbol{\theta}}}{2}\right) - \log 2 \\
&= \mathsf{JSD}(p^*, q_{\boldsymbol{\theta}}) - \log 2
\end{aligned}
$$

- In practice, we can only approximate the optimal discriminator

Quality and coverage

- **Coverage**: divergence is small where true density is high and the mixture $p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})$ has high probability.
- Penalize regions with real examples but no generated samples

$$D_{\mathbb{KL}}\left(p^* \middle\| \frac{p^* + q_{\boldsymbol{\theta}}}{2}\right) = \frac{1}{2}\int p^*(\boldsymbol{x}) \log \frac{2p^*(\boldsymbol{x})}{p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})} \mathrm{d}\boldsymbol{x}$$

- **Quality**: divergence is small where generated sample density $q_{\boldsymbol{\theta}}(\boldsymbol{x})$ is high and the mixture $p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})$ has high probability;
- Penalize regions with generated samples but no real samples

$$D_{\mathbb{KL}}\left(q_{\boldsymbol{\theta}}(\boldsymbol{x}) \middle\| \frac{p^* + q_{\boldsymbol{\theta}}}{2}\right) = \frac{1}{2}\int q_{\boldsymbol{\theta}}(\boldsymbol{x}) \log \frac{2q_{\boldsymbol{\theta}}(\boldsymbol{x})}{p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})} \mathrm{d}\boldsymbol{x}$$

- Balancing quality and diversity (coverage) are fundamental problems in generative modeling
- Mode collapse is a typical problem with GANs

Problems with density ratios

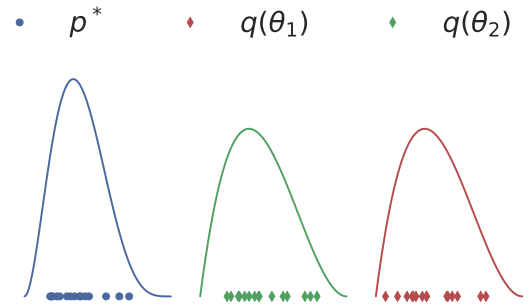


Figure: Murphy, Fig. 26.4a

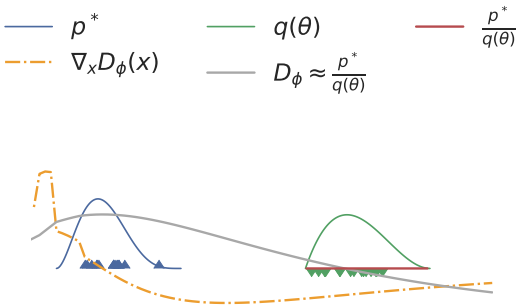


Figure: Murphy, Fig. 26.4b

- Failure of the KL divergence to distinguish between distributions with non-overlapping support
- $D_{\mathbb{KL}}(p^* \| q_{\theta_1}) = D_{\mathbb{KL}}(p^* \| q_{\theta_2}) = \inf$
- q_{θ_1} is clearly closer to p^* than q_{θ_2}

- The density ratio $\frac{p^*}{q_\theta}$ used by the KL divergence and a smooth estimate given by an MLP, together with the gradient it provides with respect to the input variable.

GAN min-max game

- Global minimum of $\mathcal{L}_G$ is achieved if and only if $q_{\boldsymbol{\theta}} = p^*$
- This is the Nash-equilibrium of GAN min-max game – no player can gain anything by changing just their own strategy
- Optimal discriminator is then

$$D^*(\boldsymbol{x}) = \frac{p^*(\boldsymbol{x})}{p^*(\boldsymbol{x}) + q_{\boldsymbol{\theta}}(\boldsymbol{x})} = \frac{p^*(\boldsymbol{x})}{p^*(\boldsymbol{x}) + p^*(\boldsymbol{x})} = \frac{1}{2}$$

- This means that Discriminator can not distinguish generated samples from real data examples

GAN mode collapse

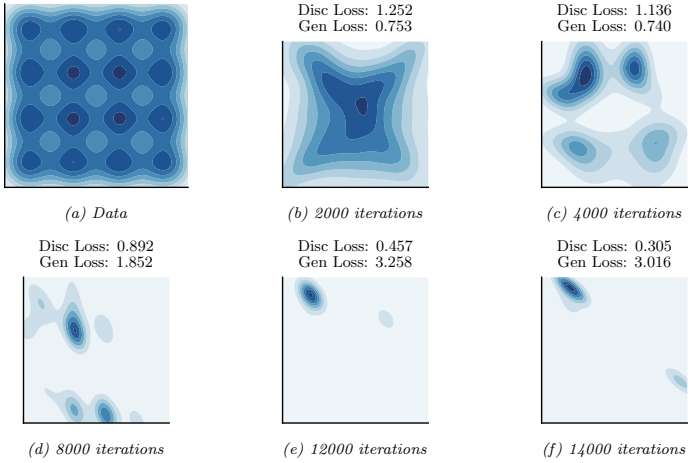


Figure: Mode collapse with GAN trying to fit a dataset of 16 Gaussians in 2D (Murphy, Fig. 26.6)

Least squares GAN and non-zero-sum GANs

- General GAN loss functions can have independent terms for Discriminator and Generator

$$\min_{\boldsymbol{\phi}} \mathcal{L}_D(\boldsymbol{\phi}, \boldsymbol{\theta}); \min_{\boldsymbol{\theta}} \mathcal{L}_G(\boldsymbol{\phi}, \boldsymbol{\theta})$$

- Zero-sum formulations $-L_D(\boldsymbol{\phi}, \boldsymbol{\theta}) = L_G(\boldsymbol{\phi}, \boldsymbol{\theta})$ are not strictly needed for GANs to work
- See Murphy section 26.3.2 for a wide range of various GAN loss functions and how they connect to f-divergences

Least-squares GAN

- Discriminator tries to output 1 (real data label) for the real examples and 0 (fake data label) for generated samples

$$\mathcal{L}_D = \mathbb{E}_{p^*(\boldsymbol{x})}(D(\boldsymbol{x}) - 1)^2 + \mathbb{E}_{q_\theta(\boldsymbol{x})} D(\boldsymbol{x})^2$$

- Generator tries to generated samples that get a discriminator output close to 1 (real data label)

$$\mathcal{L}_G = \mathbb{E}_{q_\theta(\boldsymbol{x})}(D(\boldsymbol{x}) - 1)^2$$

Discriminator Feature Matching and Moment Matching

- Often the Discriminator becomes too powerful and it's helpful to give the generator a sneak-peek into Discriminator activations
- Compute discriminator hidden activations $D(\boldsymbol{x})$ for real and generated samples and try to match those
- Moment matching tries to match the statistics of discriminator activations using a moment generating function $s(\cdot)$, for example, the mean and variance

$$\mathcal{L}_{\text{MM}} = \min_{\boldsymbol{\theta}} \|\mathbb{E}_{p^*(\boldsymbol{x})} s(D(\boldsymbol{x})) - \mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{x})} s(D(\boldsymbol{x}))\|_2^2$$

- Feature matching refers to trying to directly match the hidden activations without first computing statistics – this assumes paired data

$$\mathcal{L}_{\text{FM}} = \min_{\boldsymbol{\theta}} \mathbb{E}_{p^*(\boldsymbol{x}), q_{\boldsymbol{\theta}}(\boldsymbol{x})} \|D(\boldsymbol{x}) - D(G(\boldsymbol{z}))\|_2^2$$

- Feature matching is heavily used in speech synthesis, especially HiFi-GAN and related models
- Murphy 26.2.5

Deep Convolutional GAN (DCGAN)

- In the generator, a 100D latent variable z is mapped by a linear transformation to a 4×4 representation with 1024 channels.
- This is then passed through convolutional layers that gradually upsample the representation and decrease the number of channels.
- At the end a tanh function maps the representation to a fixed range so that it can represent an image.
- The discriminator consists of a standard convolutional net that classifies the input as either a real example or a generated sample.

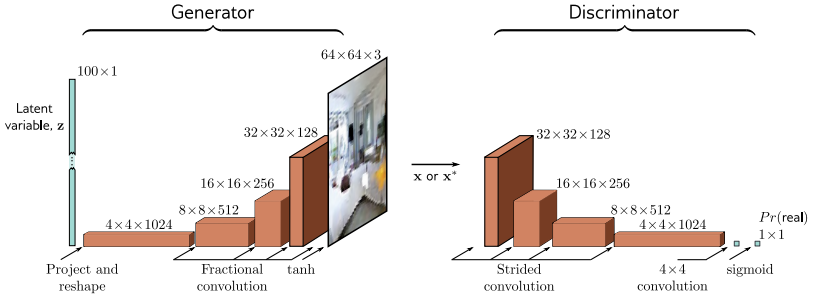
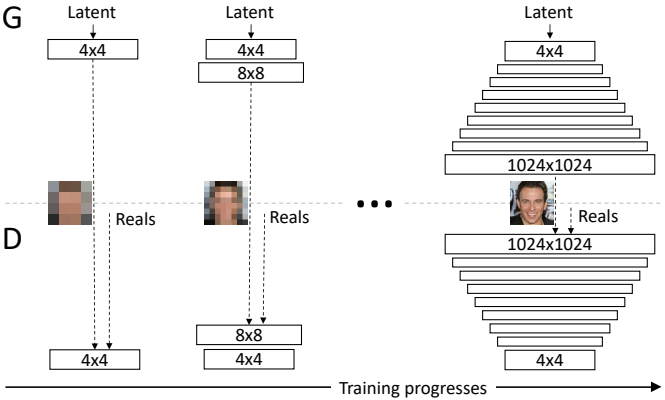


Figure: DCGAN architecture. Fig. 15.3 from Prince

Progressive growing of GAN

- Start by generating and discriminating images at low resolutions, progressively add more upsampling and downsampling layers
- Training is more stable for small images
- Training time is reduced compared to iterating at full resolution from the beginning



Images generated by progressive growing of GANs

Figure: From Karras et al., 2018

Conditional GAN

- The generator of the conditional GAN also receives an attribute vector c describing some aspect of the image.
- As usual, the discriminator receives either a real example or a generated sample, but now it also receives the attribute vector; this encourages the samples both to be realistic and compatible with the attribute.

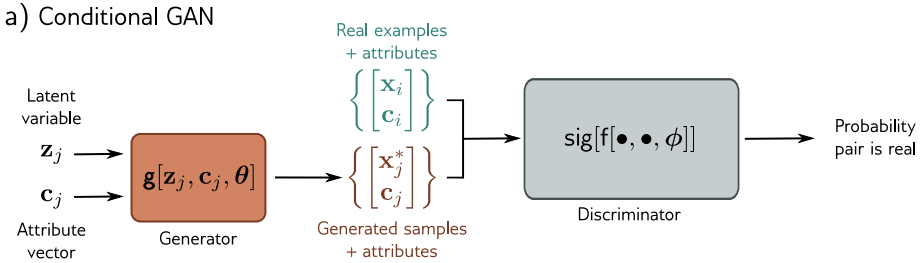


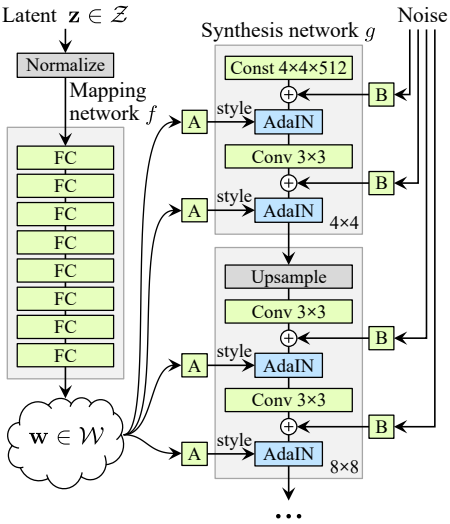
Figure: Fig. 15.13 from Prince, 2024

- Mapping network learns different conditioning features (styles) for different spatial resolutions in the network
- Additional noise inputs are provided for texture
- Conditioning combines FiLM and batch norm into Adaptive Instance Normalisation (AdaIN)

$$\text{AdaIN}(\boldsymbol{x}_i, \boldsymbol{y}) = \boldsymbol{y}_{s,i} \odot \frac{\boldsymbol{x}_i - \mu(\boldsymbol{x}_i)}{\sigma(\boldsymbol{x}_i)} + \boldsymbol{y}_{b,i}$$

where $\boldsymbol{x}_i$ is one feature map, $\mu(\boldsymbol{x}_i)$ and $\sigma(\boldsymbol{x}_i)$ are minibatch mean and standard deviation, respectively

- Style vectors $(\boldsymbol{y}_s, \boldsymbol{y}_b)$ are multiplicative and additive conditioning (similar to FiLM) derived from the latent $\boldsymbol{z}$



Cycle GAN – unpaired image-to-image translation

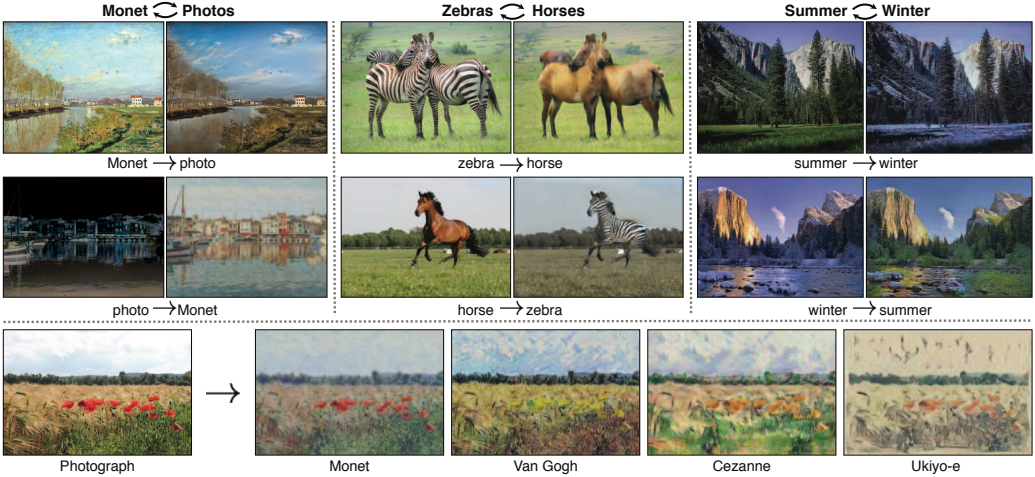


Figure: Examples of unpaired image-to-image translation. (Murphy Fig. 26.18, orig. Zhu et al., 2017)

Cycle GAN

- Cycle consistency loss: round-trip converted samples should be similar to the original sample

$$\mathcal{L}_{\text{cycle}} = \mathbb{E}_{p(\boldsymbol{x})}\|F(G(\boldsymbol{x})) - \boldsymbol{x})\|_1 + \mathbb{E}_{p(\boldsymbol{y})}\|G(F(\boldsymbol{y})) - \boldsymbol{y})\|_1$$

- Identity loss: converting images back to its own domain should not change the image

$$\mathcal{L}_{\text{identity}} = \mathbb{E}_{p(\boldsymbol{x})}\|\boldsymbol{x} - F(\boldsymbol{x})\|_1 + \mathbb{E}_{p(\boldsymbol{y})}\|\boldsymbol{y} - G(\boldsymbol{y})\|_1$$

- Total loss

$$\mathcal{L} = \mathcal{L}_{GAN} + \lambda_1 \mathcal{L}_{\text{cycle}} + \lambda_2 \mathcal{L}_{\text{identity}}$$

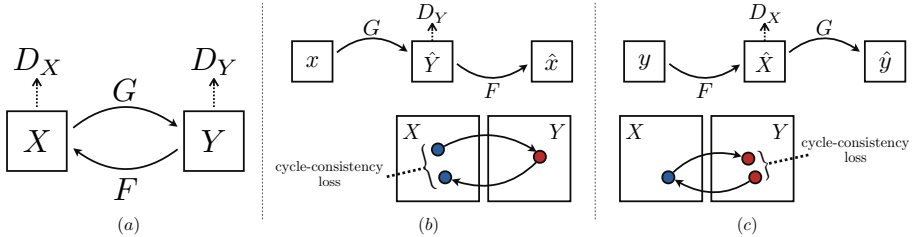


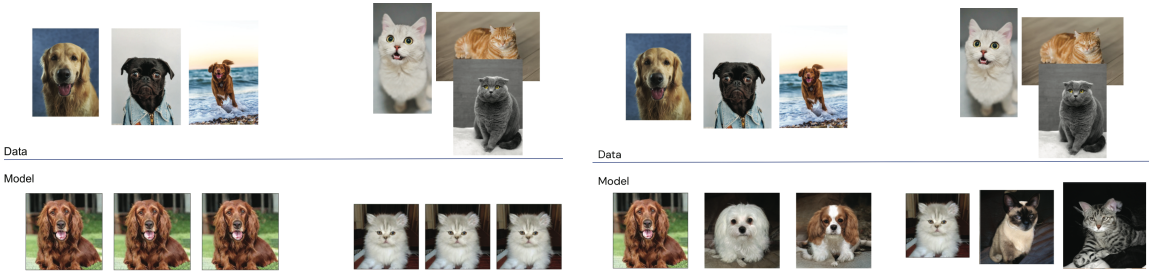
Figure: CycleGAN training scheme. (Murphy Fig. 26.18, orig. Zhu et al. 2017)

Evaluation of generated samples

(a) Generated samples are of high quality, but not diverse

(b) Generated samples are both high quality and diverse

- How to capture diversity in an evaluation metric?



(a) *(b)*

Figure: Murphy Fig. 20.11

Fréchet distances

- Fréchet Inception Distance (FID) is calculated from statistics of hidden activations in a pre-trained image classified (Inception Net in this case)
- Calculate mean $\boldsymbol{\mu}_m$ and covariance Σ_m from generated model output samples
- Calculate mean $\boldsymbol{\mu}_d$ and covariance Σ_d from real data examples

$$\text{FID} = \|\boldsymbol{\mu}_m - \boldsymbol{\mu}_d\|_2^2 - \text{tr}\left(\Sigma_d + \Sigma_m - 2(\Sigma_d\Sigma_m)^{1/2}\right)$$

- The idea of Fréchet distances is applicable to audio (FAD) and other domains – choose the suitable pretrained classifier for your application
- Similar to moment matching and discriminator feature matching, but typically only used for evaluation
- More in Murphy, Section 20.4.2

Summary

- Generative Adversarial Networks (GANs)
- Learning by comparison
- GAN loss functions
- GAN training process
- Theoretical properties: density ratios, quality, coverage
- Conditional GAN
- GAN architectures
- Cycle GAN

- Reading: Section 22 from Murphy, 2023; Chapter 15 from Prince 2024; Chapter 17 from Bishop, 2024)

References

- Murphy K, Probabilistic Machine Learning: Advanced Topics, 2023.
- Prince SJD, Understanding Deep Learning, The MIT Press, 2024.
- Bishop, Christopher M., and Hugh Bishop. Deep learning: Foundations and concepts. Springer Nature, 2024.