Deep Learning - COSC2779

Deep Unsupervised Learning

Dr. Iman Abbasnejad

RMIT UNIVERSITY

September 20, 2021

Reference: *Chapter 14, 20: Ian Goodfellow et. al., "Deep Learning", MIT Press, 2016.*

Outline

Data:

$$\mathcal{D} = \left\{ \left(\mathbf{x}^{(i)}, y^{(i)} \right) \right\}_{i=1}^{N}$$

Goal: Learn a function that $h : \mathbf{x} \rightarrow y$

Example: Classification, Regression, Object detection, Image segmentation, Sentiment analysis, Machine Translation, Image captioning, ...

Probabilistic interpretation:
$p\left(y^{(i)} \mid x_1^{(i)}, \cdots, x_d^{(i)}\right)$

Classification:

$\longrightarrow$ Dog

Supervised Learning

Data:

$$\mathcal{D} = \left\{ \left(\mathbf{x}^{(i)}, y^{(i)} \right) \right\}_{i=1}^{N}$$

Goal: Learn a function that $h : \mathbf{x} \rightarrow y$

Example: Classification, Regression, Object detection, Image segmentation, Sentiment analysis, Machine Translation, Image captioning, ...

Unsupervised Learning

Data:

$$\mathcal{D} = \left\{ \left(\mathbf{x}^{(i)}, \cdot \right) \right\}_{i=1}^{N}$$

Goal: Learn some underlying hidden structure of the data.

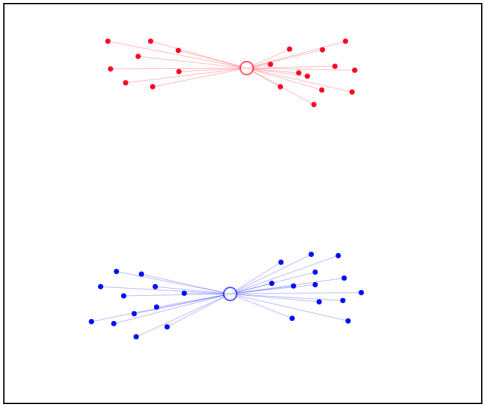
Example: Clustering, dimensionality reduction, feature learning, density estimation, ...

Example: K-means clustering

Data:

$$\mathcal{D} = \left\{ \left(\mathbf{x}^{(i)}, . \right) \right\}_{i=1}^{N}$$

Goal: To assign sample x^i to the k-th cluster.



A form of unsupervised learning.

Data:

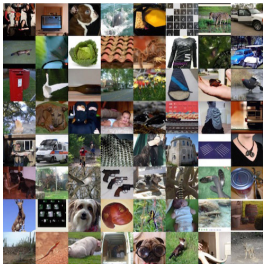
$$\mathcal{D} = \left\{ \mathbf{x}^{(i)} \right\}_{i=1}^{N}$$

Goal: Given training data, generate new samples from same distribution.

Example: Autoencoders, GAN, One-to-Many RNN, ...

Probabilistic interpretation:
$p\left(x_1^{(i)}, \cdots, x_d^{(i)}\right)$

Learning:

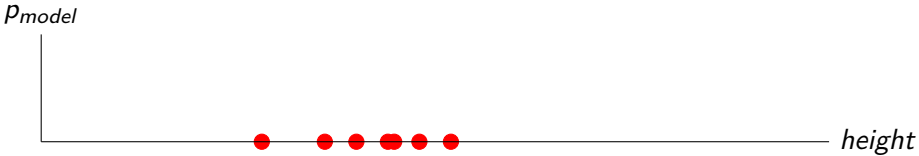


$\longrightarrow$ $p_{model}(x)$

Inference (testing):

$p_{model}(x)$ $\longrightarrow$ 

Given the heights of students in a class, guess the height of the next student entering the class.



Assumption: The heights of students are normally distributed

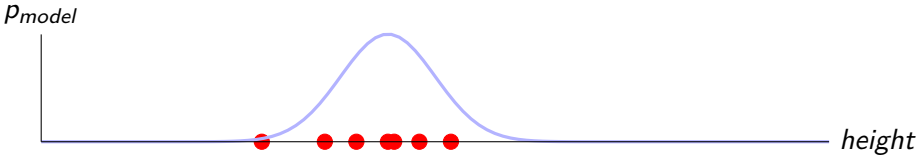
$p_{model} = p\left(x^{(i)}\right) = \mathcal{N}(x^i; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp - \frac{(x^i - \mu)^2}{2\sigma^2}$

We can fit the model to data using MLE - Learning.

After Learning, we can use the model to generate new data - Sampling. E.g. Inverse CDF sampling

Generative Models: Simple Example

Given the heights of students in a class, guess the height of the next student entering the class.



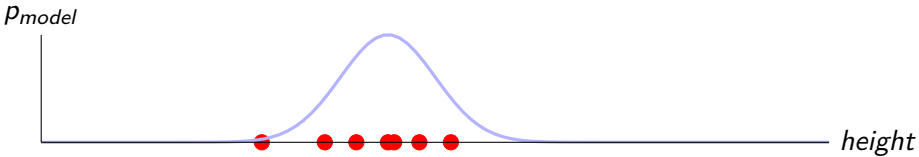
Assumption: The heights of students are normally distributed

$p_{model} = p\left(x^{(i)}\right) = \mathcal{N}(x^i; \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp - \frac{(x^i - \mu)^2}{2\sigma^2}$

We can fit the model to data using MLE - Learning.

After Learning, we can use the model to generate new data - Sampling. E.g. Inverse CDF sampling

Given the heights of students in a class, guess the height of the next student entering the class.



Assumption: The heights of students are normally distributed

$p_{model} = p\left(x^1, x^2, ..., x^n|\Theta\right) = \prod_{i=1}^{N} p\left(x^i|\Theta\right)$

$$\text{MLE:} \Rightarrow \frac{\partial}{\partial\Theta} \prod_{i=1}^{N} p\left(x^i|\Theta\right) = 0 \Rightarrow \sum_{i=1}^{N} \frac{\partial}{\partial\Theta} p\left(x^i|\Theta\right) = 0$$

$$\mu_{MLE} = \frac{1}{N} \sum_{i=1}^{i=N} x^i, \sigma_{MLE} = \sqrt{\frac{1}{N} \sum_{i=1}^{i=N} (x^i - \mu)^2}$$

- Learn useful features for downstream tasks such as classification.
- Getting insights from high-dimensional data (physics, medical imaging, etc.)
- Realistic samples for artwork, super-resolution, colorization, etc
- Modeling physical world for simulation and planning
- ...

18 Impressive Applications of Generative Adversarial Networks (GANs)

Objective of the Lecture

- Gain a basic understanding of the **deep generative models** applicable for image, text and speech generation.
- Will be a more **conceptual** lecture.
- Will provide an example for training a generative model.

Outline

Example Scenario

Airbus provides several services for the operation of the Columbus module and its payloads on the International Space Station (ISS).

To ensure the health of the crew as well as hundreds of systems onboard the Columbus module, engineers have to keep track of many telemetry data-streams, which are constantly beamed to earth.

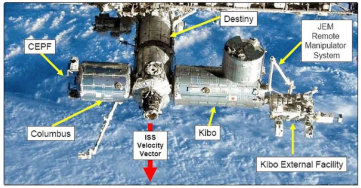
Airbus is interested in automated detection of anomalies in the telemetry data-streams.

Data: Telemetry data-streams of the last 10 years results in over 5 trillion data points. However there are very few (none for most systems) anomalies.

How Airbus Detects Anomalies in ISS Telemetry Data Using TFX

April 09, 2020

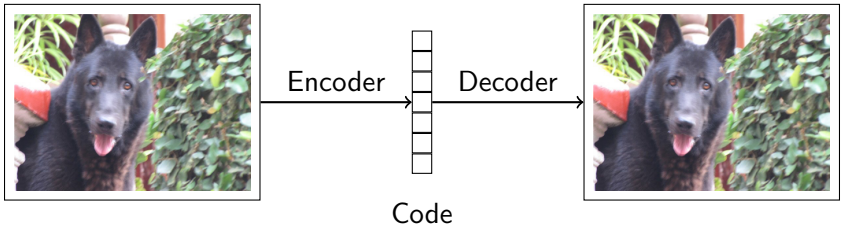
A guest post by Philipp Grashorn, Jonas Hansen and Marcel Rummens from Airbus



How Airbus Detects Anomalies in ISS Telemetry Data Using TFX

- Can this be approached as a supervised learning problem?

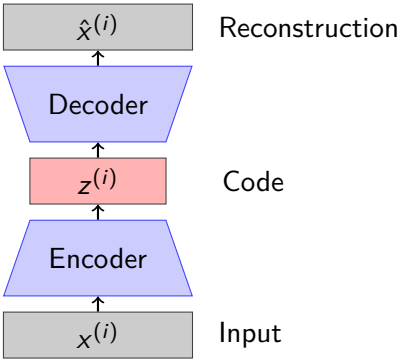
- Can this be approached as a supervised learning problem?
  - Heavily skewed data. 99.9% data maybe from Normal class. Very few anomalous examples (none for most systems).
  - Anomaly is not a pattern that we can learn. The pattern is only in the normal data.
- We are interested in modelling how normal data is distributed. $p_{normal}(\mathbf{x})$



Think about JPEG encoding and decoding. In that case both encoder and decoder are predetermined functions.

Can we learn an encoding function and decoding function using data.

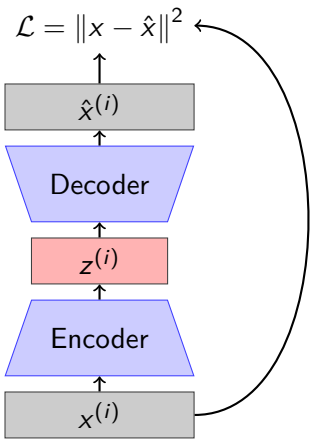
Parallel idea: PCA (Dimensionality reduction).



Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

- x is the **input**. Can be an image, sequence or other feature vector.
- $\hat{x}$ is the **prediction**. Same dimension as the input.
- z is the **Latent** representation (code). Usually has smaller dimensions than the input.
- Both encoder and decoder are neural networks (MLP, CNN, RNN).

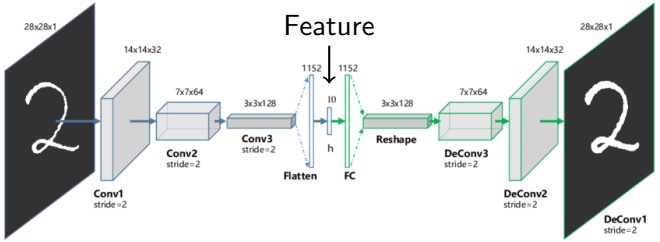
Autoencoder Training



Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

- Only need x to train the network.
- z has to capture information in the input image in order to be able to reconstruct that image.
- Therefore, this training process will generate a model that can encode image information into a compact vector (z)

Will only work well for test inputs that are "*similar*" to the training data.

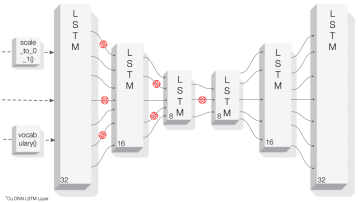


Input

Reconstruction

- Encoder consists of convolution layers. Some layers with Striding (or pooling) to reduce dimension.
- Decorder consists of convolution layers. Some layers with Transpose convolution (or un-pooling) to increase dimension.

For sequence data, both encoder and decoder can be RNN.



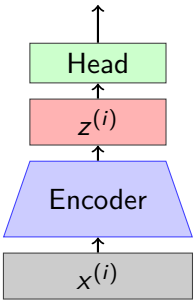
How Airbus Detects Anomalies in ISS Telemetry Data Using TFX

Train the AutoEncoder model **only using the normal data**.

The AutoEncoder will **learn to reconstruct the normal data well** - Low reconstruction error on normal data.

The hypothesis is that it will NOT do well on anomalous data.

Threshold the reconstruction error to detect anomalies in test data.



Can use the trained encoder as a feature extractor and do transfer learning for other similar tasks.

A form of self-supervision.

Can also use z to cluster the dataset.

Used to be a common pre-training technique for image classification. Not so popular anymore as transfer learning has taken over.

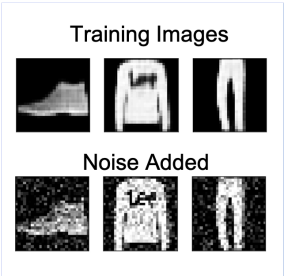
Example: De-Noising AutoEncoder

You are given a set of images from a clothing database. The task is to create a feature extractor that can be used for classifying the images to common clothing categories. E.g. Trousers, t-shirts, ...

An AutoEncoder with sufficient complexity may learn the so-called "Identity Function", meaning that the output equals the input, marking the AutoEncoder useless.

Example: De-Noising AutoEncoder

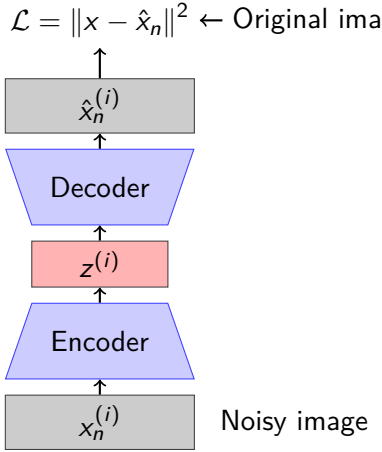
You are given a set of images from a clothing database. The task is to create a feature extractor that can be used for classifying the images to common clothing categories. E.g. Trousers, t-shirts, ...



Train an AutoEncoder with Noise added images as input and the corresponding training image as the target.

This is called a de-noising AutoEncoder.

Example: De-Noising AutoEncoder

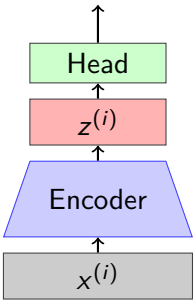


Train an AutoEncoder with Noise added images as input and the corresponding training image as the target.

At **test time**, Input corrupted test image and the network will generate a noise free version of it.

Now the network is forced to learn underlying structure. Cannot take shortcuts by memorizing inputs.

Example: De-Noising AutoEncoder



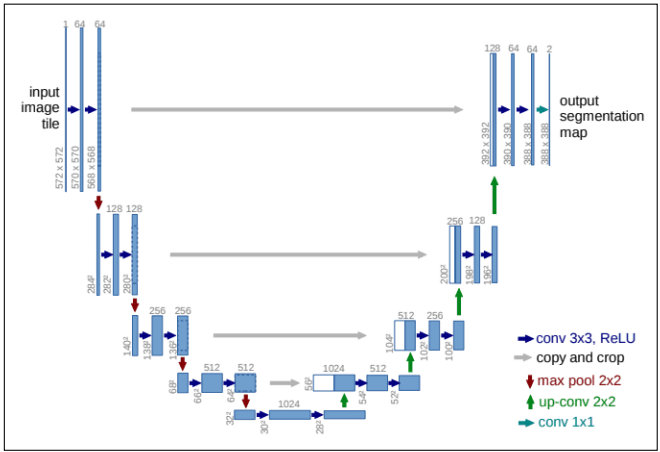
Can use the trained encoder as a feature extractor and do transfer learning for other similar tasks.

A form of self-supervision.

Can also use z to cluster the dataset.

Used to be a common pre-training technique for image classification. Not so popular anymore as transfer learning has taken over.

Example: Segmentation AutoEncoder



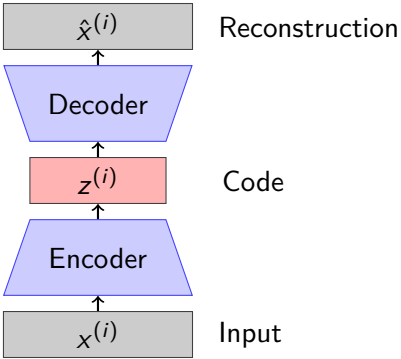
U-Net is an architecture used for segmentation.

With small training samples and augmentation they obtained state-of-the-art results on biomedical image data.

It consists of a contracting path (left side) and an expansive path (right side).

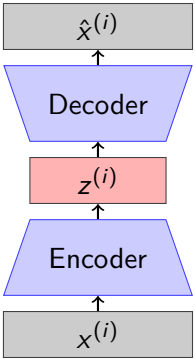
A concatenation with the correspondingly cropped feature map from the contracting path in the expansive path.

Issues in Basic Autoencoder

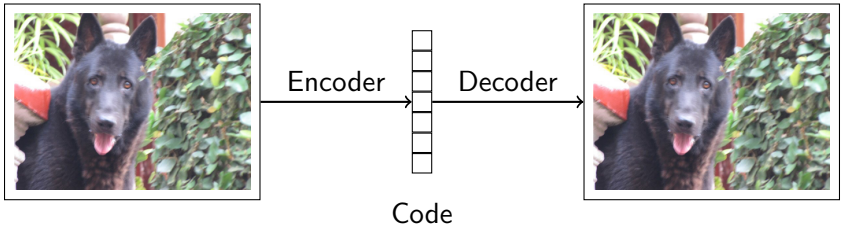


- The basic auto-encoder can memorize (over-fit) training data.
  - Adding noise to input or feature.
  - Inpainting
- Cannot generate novel images. Code is unknown.
  - Variational Auto-encoder.

Issues in Basic Autoencoder



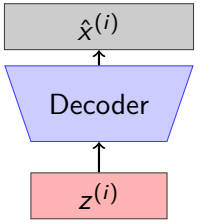
- The basic auto-encoder can memorize (over-fit) training data.
  - Adding noise to input or feature.
  - Inpainting
- Cannot generate novel images. Code is unknown.
  - Variational Auto-encoder.



Think about JPEG encoding and decoding. In that case both encoder and decoder are predetermined functions.

Can we alter the code to get new images?

Difficult. Would mostly be unrealistic even if we manage to generate an image.



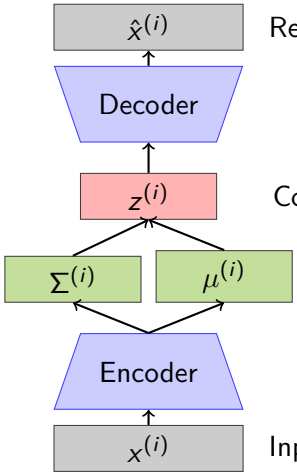
1. Sample z from prior: $z^{(i)} \sim p(z)$
2. Sample x from: $x^{(i)} \sim p_{model}\left(x \mid z^{(i)}\right)$

We want to learn $p(x_1, x_2, \cdots, x_d)$ given training data.

In autoencoder we learn $p(x_1, x_2, \cdots, x_d \mid \mathbf{z})$

What if z is from a known distribution for realistic images. E.g. $z^{(i)} \sim \mathcal{N}(0, 1)$ - Prior.

Then we can sample $z^{(i)}$ and use a decoder network to generate images.



VAE maps the input data into the parameters of a probability distribution, such as the mean and variance of a Gaussian. This approach produces a continuous, structured latent space, which is useful for image generation.

Probabilistic spin to traditional autoencoders which allows generating data

- Pros:
  - Principled approach to generative models.
  - Interpretable latent space.
  - Can be useful feature representation for other tasks.
- Cons:
  - Samples blurrier and lower quality compared to state-of-the-art (GANs)

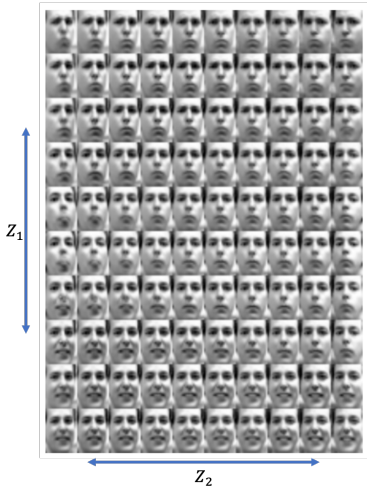


Image: Kingma and Welling, "Auto-Encoding Variational Bayes",

Outline

Example Scenario

Assume you are hired by a startup to design a system that takes a face image as input and generate aged versions of that face.

Image: businessinsider

Example Scenario

Assume you are hired by a startup to design a system that takes a face image as input and generate aged versions of that face.

It is not practical to generate a dataset that has the same person images at different times in history. This is required if we are going to use supervised learning.

However it is relatively easy to collect a dataset of faces of people with different ages. E.g. Dataset of 40 year old's, 50 year old's ...

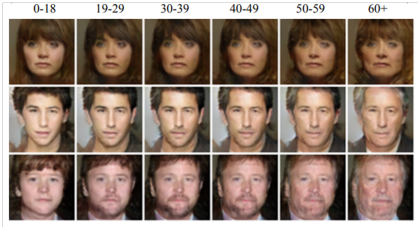
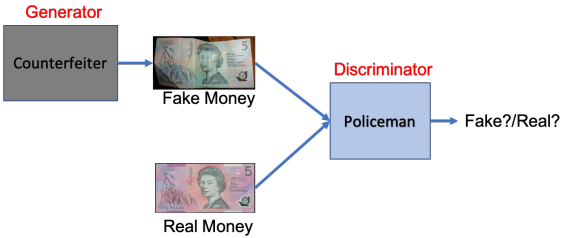


Image: Antipov, Grigory, et. al."Face aging with conditional generative adversarial networks." ICIP, 2017.

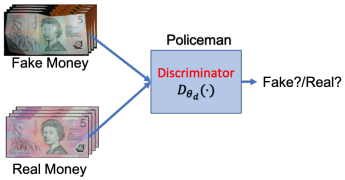


Counterfeiter: Becomes good at generating realistic looking money by learning about how he got caught.

Policeman: Becomes good at discriminating fake from real as the Counterfeiter becomes more sophisticated.

A **two player game** where both learn from each other. Players are neural networks for our use case.

Discriminator: Try to distinguish between real and fake images.



Discriminator has a binary classification task, Objective:

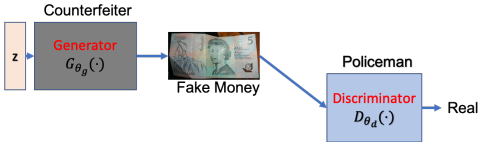
$$\arg\max_{\theta_d} \sum_i \left[\log D_{\theta_d}\left(x^{(i)}\right) + \log\left(1 - D_{\theta_d}\left(G_{\theta_g}\left(z^{(i)}\right)\right)\right)\right]$$

$D(x) = \frac{p_{data}}{p_{data}+p_g} \Rightarrow$ Discriminator cannot distinguish between real and fake therefore the optimum probability of $D^*(x) = \frac{1}{2}$

Real images: $x^{(i)}$
Fake images: $G_{\theta_g}\left(z^{(i)}\right)$

Latent feature (code): $z^{(i)}$

Generator: Try to fool the discriminator by generating real-looking images



Generator Objective:

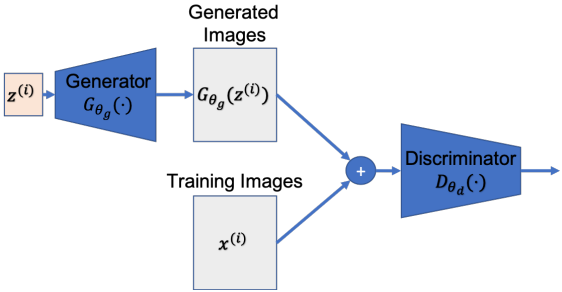
$$\arg\max_{\theta_g} \sum_i \left[\log\left(D_{\theta_d}\left(G_{\theta_g}\left(z^{(i)}\right)\right)\right)\right]$$

Generator wants to maximize objective such that D(G(z)) is close to 1 (discriminator is fooled into thinking generated G(z) is real)

Real images: $x^{(i)}$

Fake images: $G_{\theta_g}\left(z^{(i)}\right)$

Latent feature (code): $z^{(i)}$



Generator network takes a random vector and maps it to a image. For images it is a upsampling network (transpose convolution).

Discriminator is a classification CNN network that can distinguish between real and fake images.

Algorithm 1: Training GANs: Goodfellow, NIPS 2014

for *Number of training iterations* **do**
  for *k steps* **do**
    Sample m code samples $\{z^{(1)}, \cdots, z^{(m)}\}$;
    Sample m real images $\{x^{(1)}, \cdots, x^{(m)}\}$ from training data;
    Update the discriminator weights (θ_d) by maximizing discriminator objective.
  end
  Sample m code samples $\{z^{(1)}, \cdots, z^{(m)}\}$;
  Update the generator weights (θ_g) by maximizing generator objective.
end

After training, use generator network to generate new images

The original GAN networks were notoriously difficult to train.

Architecture guidelines for stable deep convolutional GANs:

- Replace any pooling layer in discriminator with strided convolutions and any pooling layer in genenrator with transpose convolutions.
- Use batch-norm in both generator and descriminator.
- Remove FC layer from deep architectures.
- Use ReLU activation in generator for all layers except for output where tanh is used.
- Use LeakyReLU activation in the descriminator.

Radford et al, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", ICLR 2016

GAN: Convolutional Architectures

Image: Radford et al, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", ICLR 2016

Developing better training algorithms for GANs is still a very active research area:

- LSGAN
- Wasserstein GAN
- Improved Wasserstein GAN
- Progressive GAN
- ...

List of some GAN papers

Example Scenario

Assume you are hired by a startup to design a system that takes a face image as input and generate aged versions of that face.

It is not practical to generate a dataset that has the same person images at different times in history. This is required if we are going to use supervised learning.

However it is relatively easy to collect a dataset of faces of people with different ages. E.g. Dataset of 40 year old's, 50 year old's ...

Can we solve this problem with GANs we discussed so far?

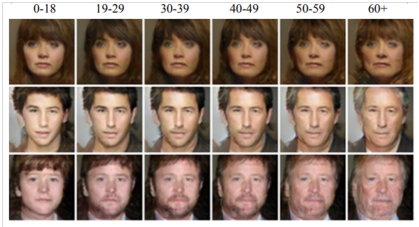
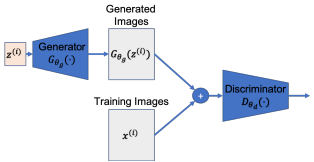
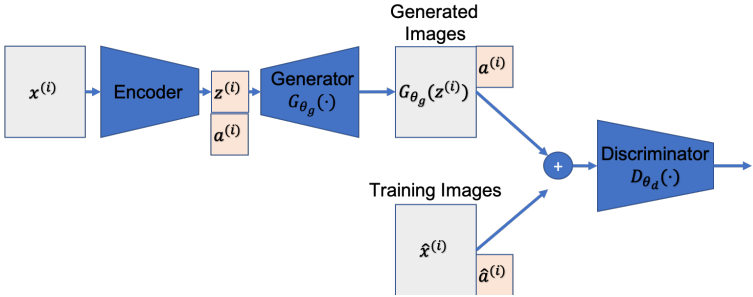


Image: Antipov, Grigory, et. al."Face aging with conditional generative adversarial networks." ICIP, 2017.



Conditional GANs



To convey the fundamental idea only. the actual network is slightly more complicated.

- Hidden representation (code), Z, is now generated by an encoder CNN.
- The generated uses the code and age related information (a) to generate the images.
- Discriminator takes in an image and age related information and decide weather the pair is fake or real.

Examples: Conditional GANs

Live Interactive Demos by NVIDIA Research

GAN: Image to Image translation

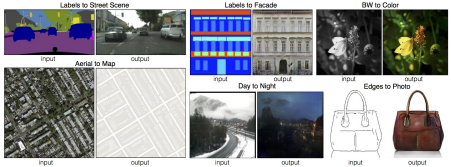


Image: Isola et al, "Image-to-image translation with conditional adversarial nets", CVPR 2017

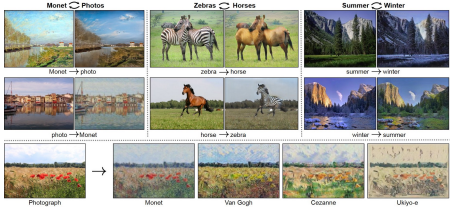
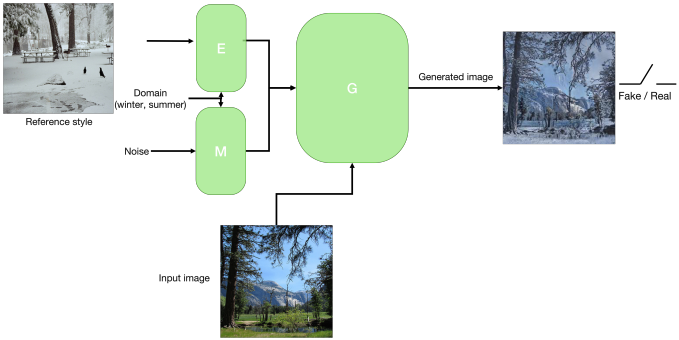
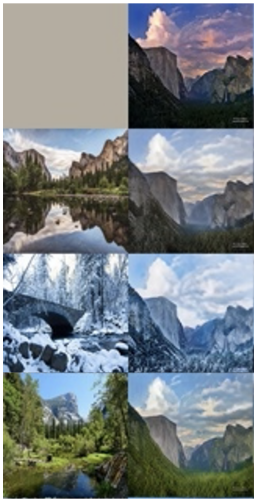


Image: Zhu et al, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks", ICCV 2017

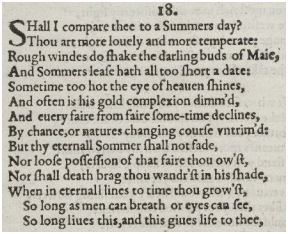
GAN: Style Transfer





Outline

Generating Text/Music.
E.g Generating text similar to Shakespeare's writing.



18.

SHall I compare thee to a Summers day?
Thou art more louely and more temperate:
Rough windes do ſhake the darling buds of Maie,
And Sommers leaſe hath all too ſhort a date:
Sometime too hot the eye of heauen ſhines,
And often is his gold complexion dimm'd,
And euery faire from faire ſome-time declines,
By chance,or natures changing courſe vntrim'd:
But thy eternall Sommer ſhall not fade,
Nor looſe poſſeſſion of that faire thou ow'ſt,
Nor ſhall death brag thou wandr'ſt in his ſhade,
When in eternall lines to time thou grow'ſt,
So long as men can breath or eyes can ſee,
So long liues this,and this giues life to thee,

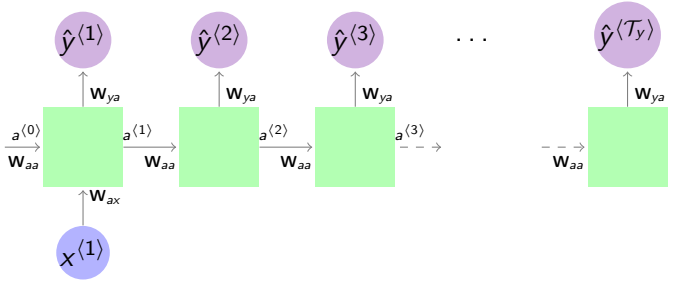
Image: Sonnet 18 in the 1609 Quarto of Shakespeare's sonnets.

Given some text from Shakespeare's writing generate novel sentences that look similar.

Generating Shakespeare's writing.

$x^{\langle t \rangle}$ is a one-hot with size equal to number of characters.

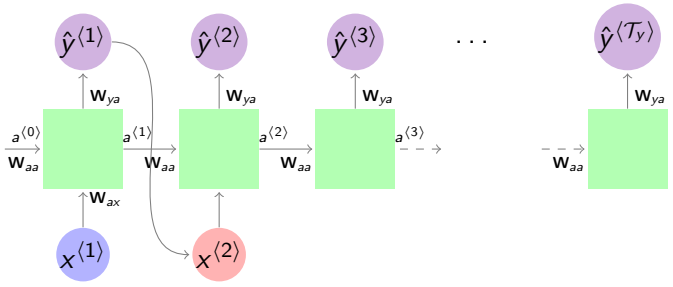
$\hat{y}^{\langle t \rangle}$ is a soft-max-out with size equal to number of characters.



All $\mathbf{W}_{aa}, \mathbf{W}_{ax}, \mathbf{W}_{ya}$ are shared across RNN cells

Text generation with an RNN

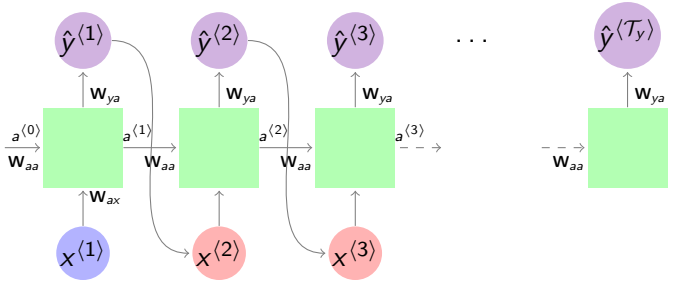
Inference: Convert $\hat{y}^{\langle t \rangle}$ to one-hot by sampling and input as $x^{\langle t+1 \rangle}$



All $\mathbf{W}_{aa}, \mathbf{W}_{ax}, \mathbf{W}_{ya}$ are shared across RNN cells

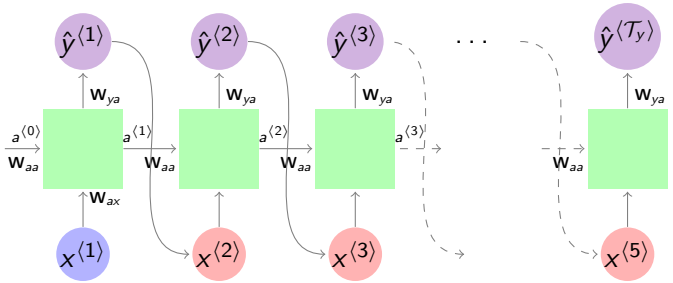
Text generation with an RNN

One-to-Many



All $\mathbf{W}_{aa}, \mathbf{W}_{ax}, \mathbf{W}_{ya}$ are shared across RNN cells

Text generation with an RNN



All $\mathbf{W}_{aa}, \mathbf{W}_{ax}, \mathbf{W}_{ya}$ are shared across RNN cells

Text generation with an RNN

Outline

WaveNet by google is one of the most popular deep networks for speech generation.

According to google: WaveNets are able to generate speech which mimics any human voice and which sounds more natural than the best existing Text-to-Speech systems, reducing the gap with human performance by over 50%.

Recommended Reading: WaveNet: A generative model for raw audio

Summary

Unsupervised learning: Deep generative models

- **AutoEncoders**: Interpretable latent space. Allows inference of $q(z|x)$, can be useful feature representation for other tasks. Samples blurrier and lower quality compared to state-of-the-art.
- **GANs**: Take game-theoretic approach. learn to generate from training distribution through 2-player game. Beautiful, state-of-the-art samples. Trickier & more unstable to train. Can't solve inference queries such as $p(x)$, $p(z|x)$.