Font Generation Trends In Deep Learning

Machine Learning Winter School 2020

2020. 12.17

숭실대학교

최 재 영

Soongsil University

Contents

Overview

Artificial Intelligence

Any technique that enables computers to mimic human behavior

Machine Learning

Ability to learn without explicitly being programed

Deep Learning

Extract patterns from data using neural networks

Why Deep Learning

Hand engineered features are time consuming and not scalable in practice

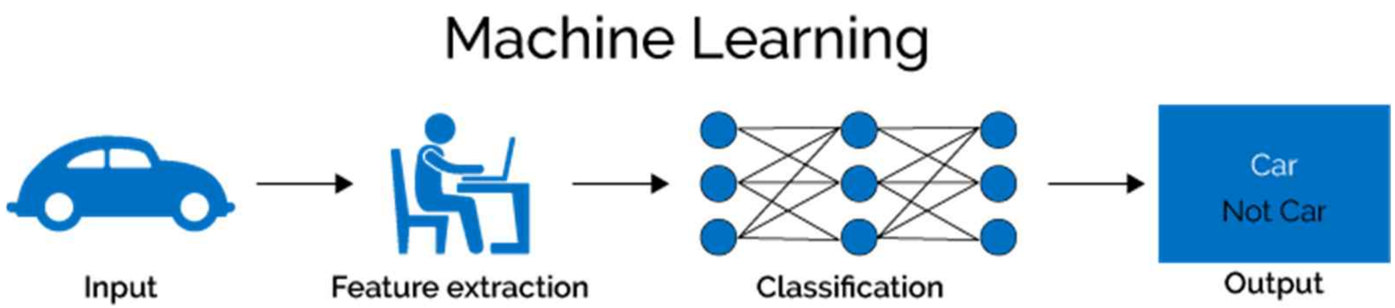
Can we learn the ***underlying features*** directly from data?

Why Deep Learning

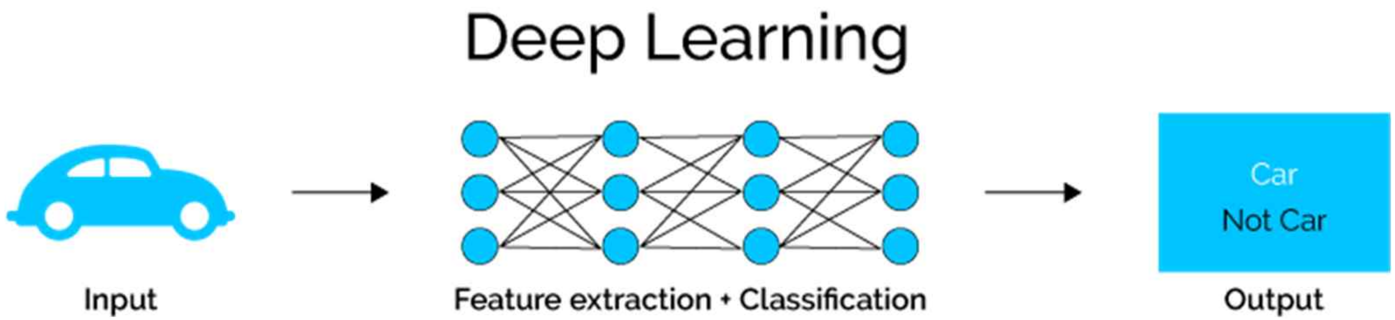
Traditional software development involves manually programming explicit rules.

Machine Learning is about learning rules from data.

Why Deep Learning



Traditional machine learning uses hand-crafted features, which is tedious and costly to develop.



Deep Learning learns hierarchical representations from the data itself, and scales with more data.

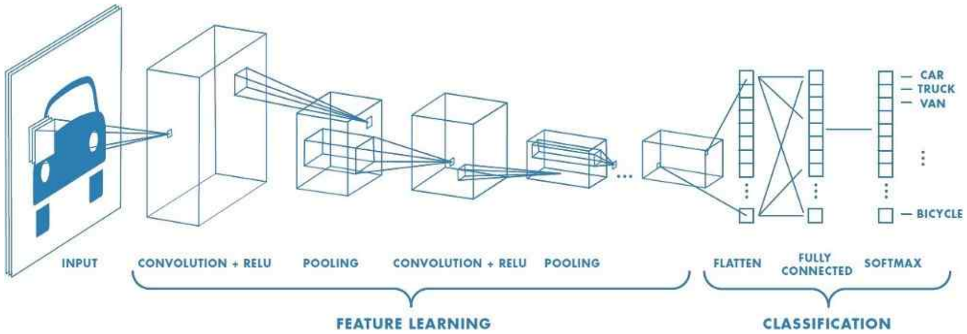
Computer Vision Tasks

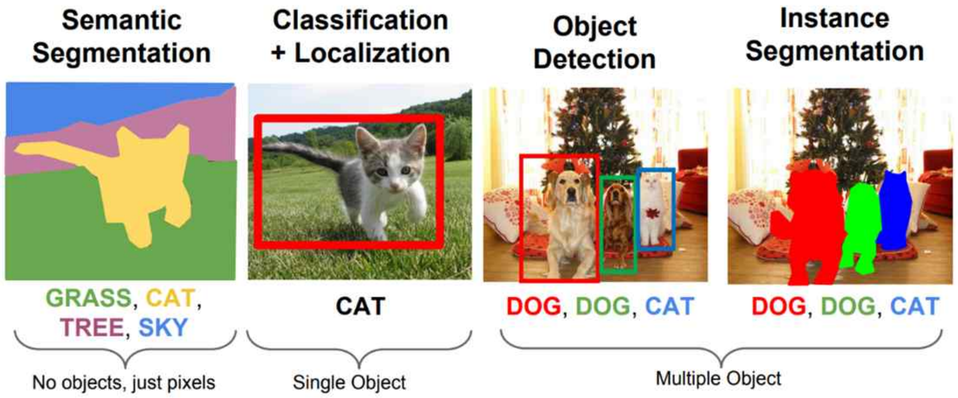
- Deep Learning has different architectures:
  - Multi-Layer Perceptron (MLP)
  - Convolutional neural net (CNN)

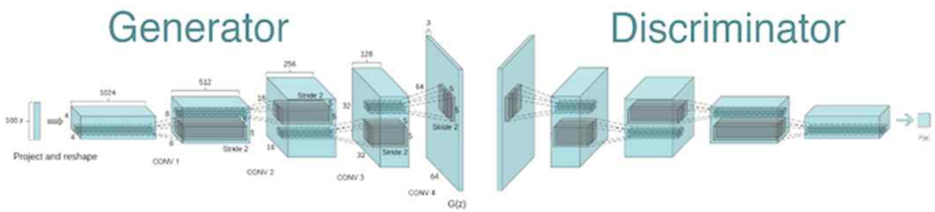
- Deep Learning introduces following computer vision application:
  - Image Classification
  - Semantic Segmentation
  - Object Classification + Localization
  - Object Detection
  - Instance Segmentation
  - Image Generation

- There are still many challenging problems to solve in computer vision.

"Single deep learning model can learn meaning from images and perform vision tasks, obviating the need for a pipeline of specialized and hand-crafted methods."







Computer Vision Tasks

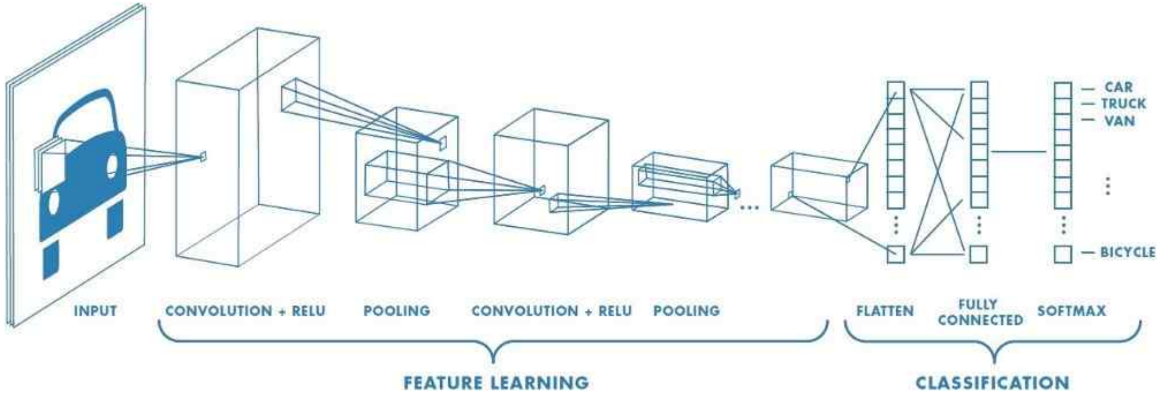
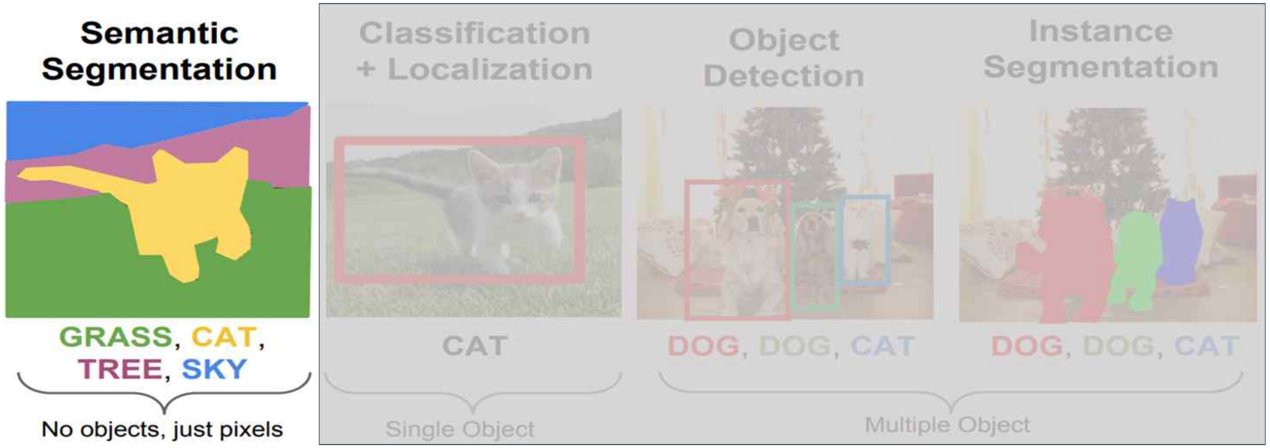


Image Classification:

- The convolutional neural network, or CNN for short, is a specialized type of neural network model used for image classification tasks
- Image classification is a supervised learning problem
- It refers to the task of
  - extracting information from the input image and
  - assigning a label to an entire image or photograph from a fixed set of defined categories
- This problem is also referred to as
  - object classification and
  - image recognition

Computer Vision Tasks

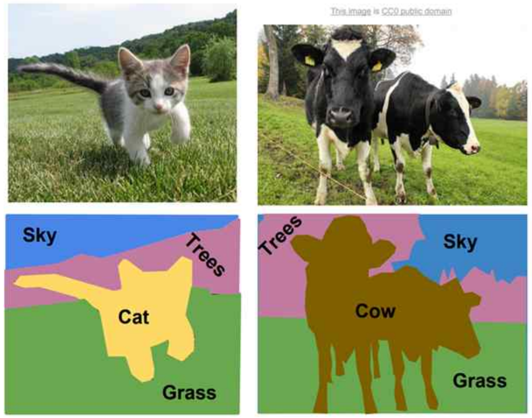


Semantic Segmentation:

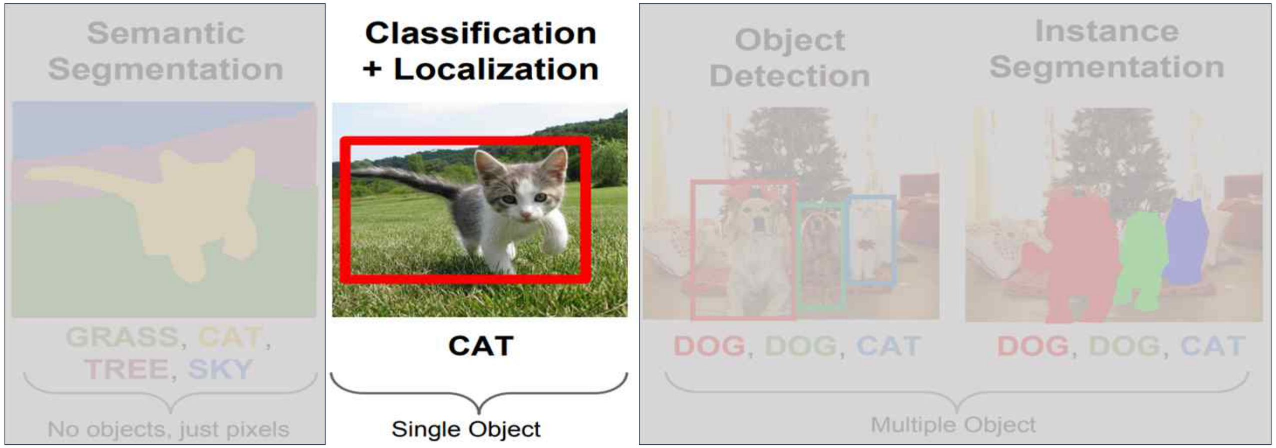
- There are two kinds of segmentation tasks in computer vision:
    - Semantic Segmentation
    - Instance Segmentation

- Label each pixel in the image with a category label
- Can't differentiate instance, only care about pixels

- This issue is fixed by Instance Segmentation



Computer Vision Tasks



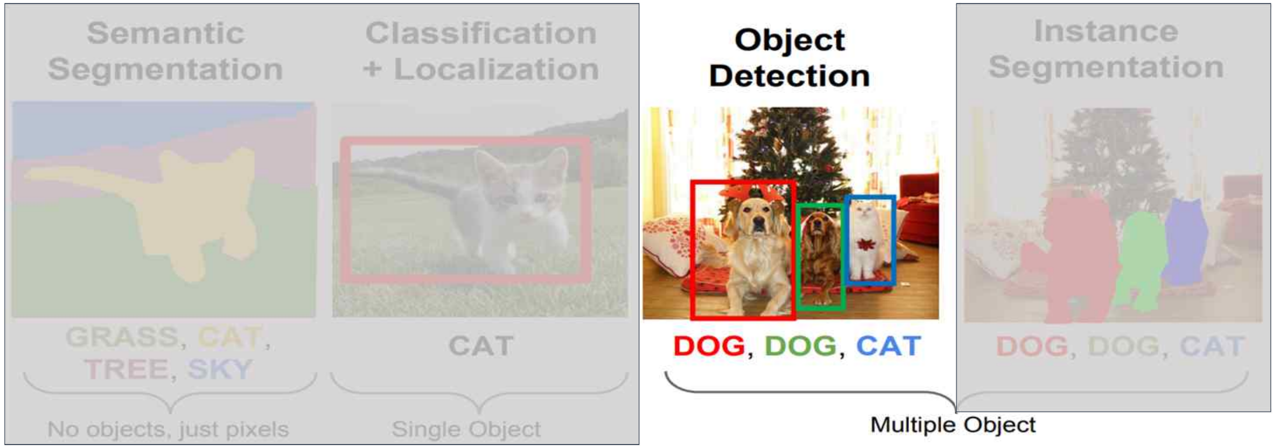
Object Classification + Localization

- Image classification with localization involves:
  - assigning a class label to an image and
  - showing the location of the object in the image by a bounding box (drawing a box around the object)

- There should be only single object in the image, it cannot classify and localize multiple objects presented in the image

- This task is also called "single-instance localization"

Computer Vision Tasks

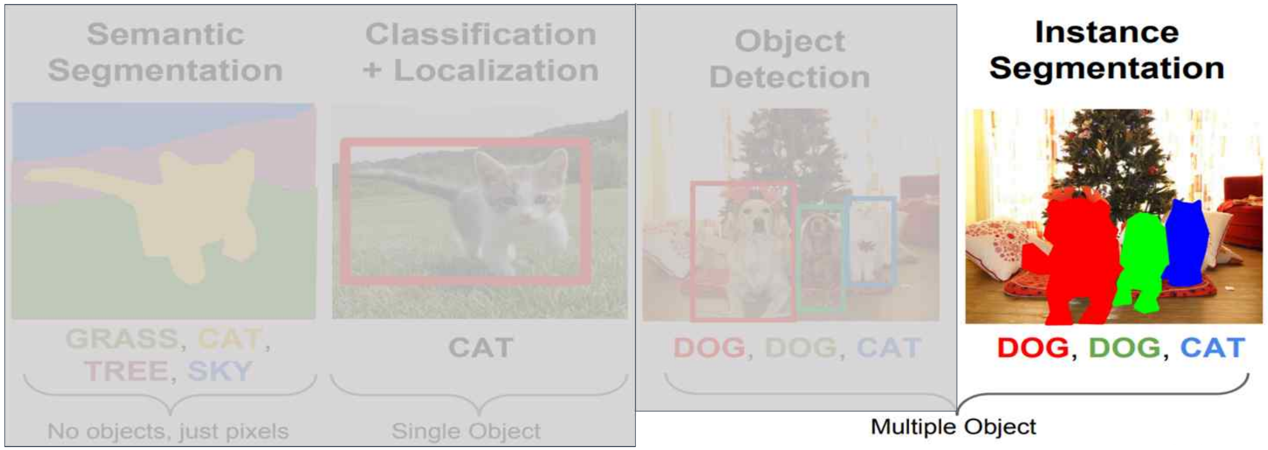


Object Detection:

- Object Detection is the ability
  - to detect or identify objects in any given image
  - along with their spatial position in the given image

- It is commonly used in applications such as
  - image retrieval
  - security
  - surveillance, and
  - automated vehicle parking systems

Computer Vision Tasks



Instance Segmentation:

- Instance segmentation is an important step to achieving a comprehensive image recognition and object detection algorithms

- It has the ability to
  - identifies each instance of each object featured in the image
  - instead of categorizing each pixel like in semantic segmentation

- For example, instead of classifying five dogs as one instance, it will identify each individual dog in the image.

Computer Vision Tasks

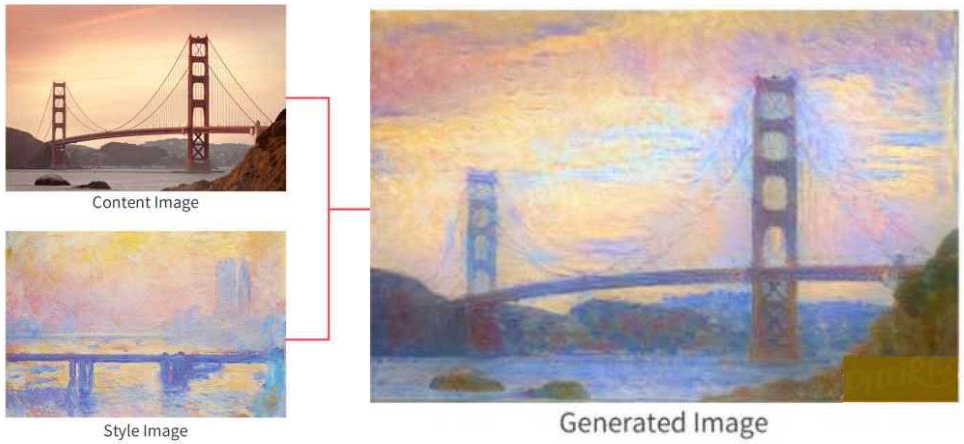


Image Generation:

- Image generation is the task of
  - generating targeted modifications of existing images
  - or entirely new images set

- It involves automatically discovering and learning the regularities or patterns in input data

- This is a very broad area that is rapidly advancing by the use of CNNs and Generative Adversarial Networks (GANs)

Generative Adversarial Nets – GAN

Proposed an architecture that can learn the generative model of any data distribution through adversarial methods with excellent performance (2014).

Generative Adversarial Nets

Ian J. Goodfellow, Jean Pouget-Abadie,[*] Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,[†] Aaron Courville, Yoshua Bengio[‡]
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7

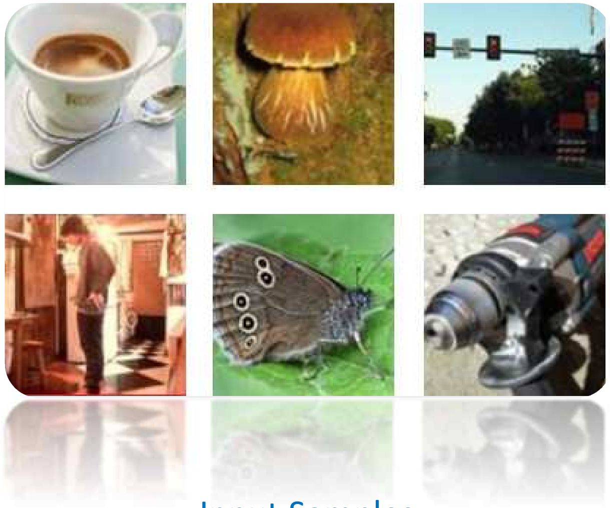
Yann LeCun described GANs as

“the most interesting idea in the last 10 years in Machine Learning.”

(Cited 25,700 at Dec 2020)

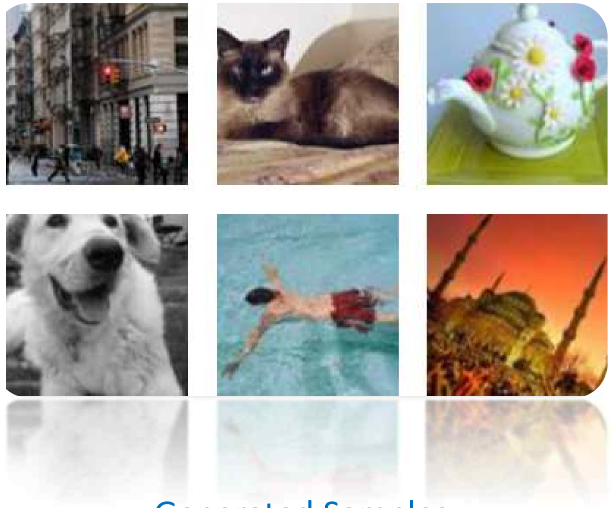
Generative Modeling

Goal: Take as input training samples from some distribution and learn a model that represents that distribution.



Input Samples

Training data ~ $P_{data}(x)$



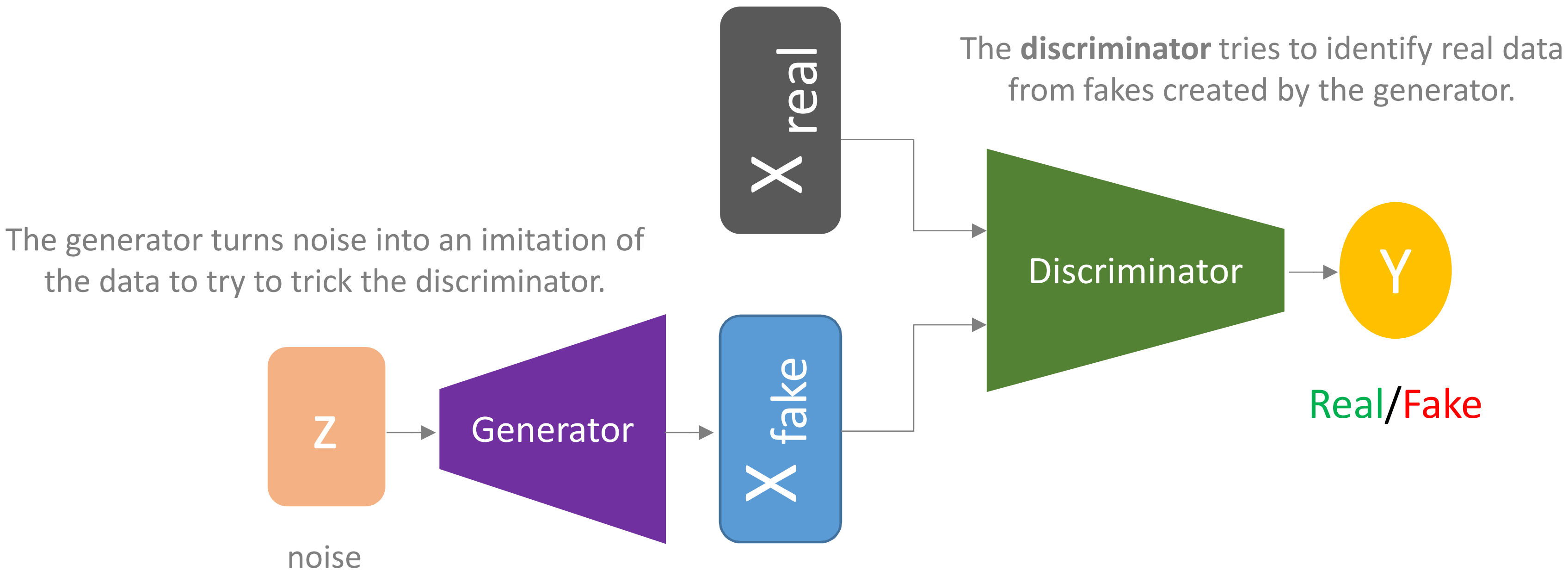
Generated Samples

Training data ~ $P_{model}(x)$

How can we learn $P_{model}(x)$ similar to $P_{data}(x)$?

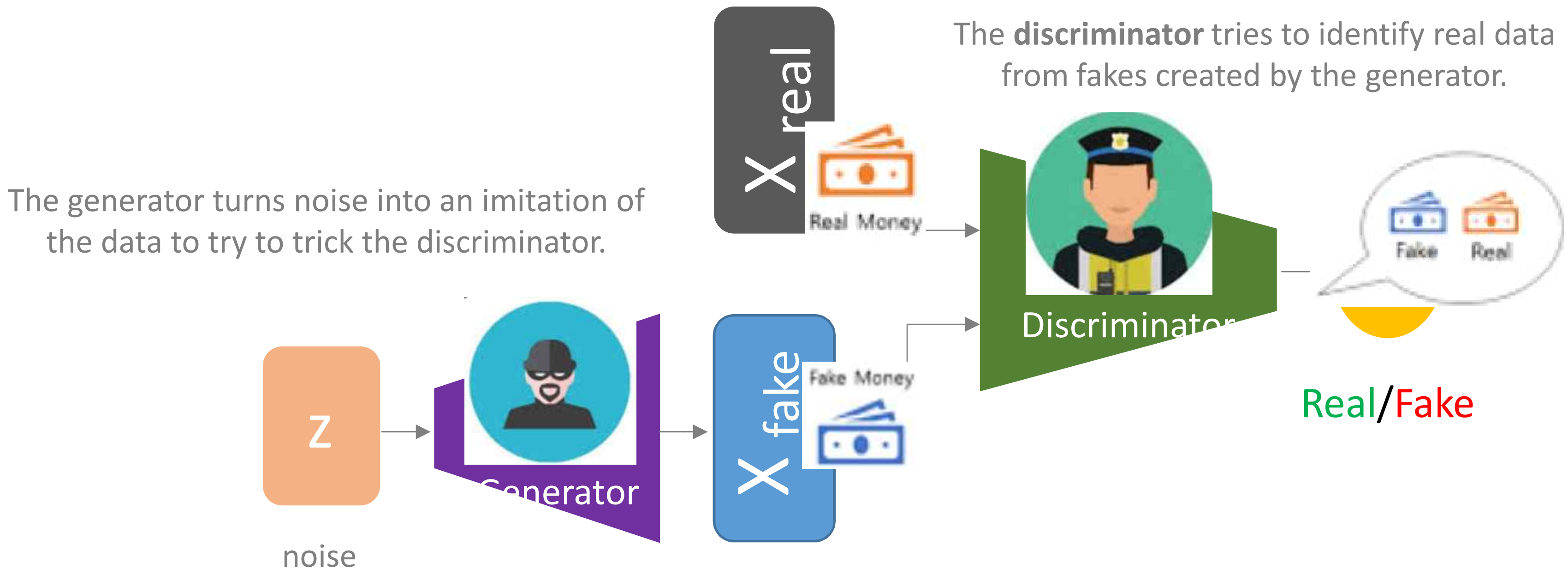
Generative Adversarial Networks (GANs)

GANs are a way to make a generative model
by having two neural networks compete with each other.



Generative Adversarial Networks (GANs)

GANs are a way to make a generative model
by having two neural networks compete with each other.

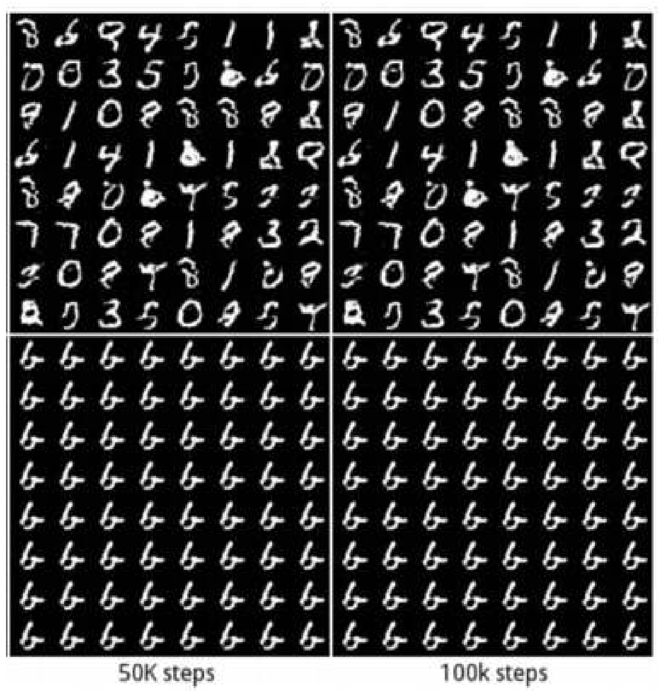


Issues in Vanilla GAN

- Compared with other generative models such as Variational Autoencoders (VAEs),
  - images generated by GANs are usually less blurred and more realistic

- However, in practice, training GANs is difficult due to following reasons:
  - Do not have control on output
  - Training instability

- Various extensions of GANs have been proposed to improve training stability:
  - Conditional GANs (cGANs)
  - Deep Convolutional Generative Adversarial Networks (DCGANs)
  - Many More



Conditional Generative Adversarial Nets – cGAN

Introduced the conditional version of generative adversarial nets.

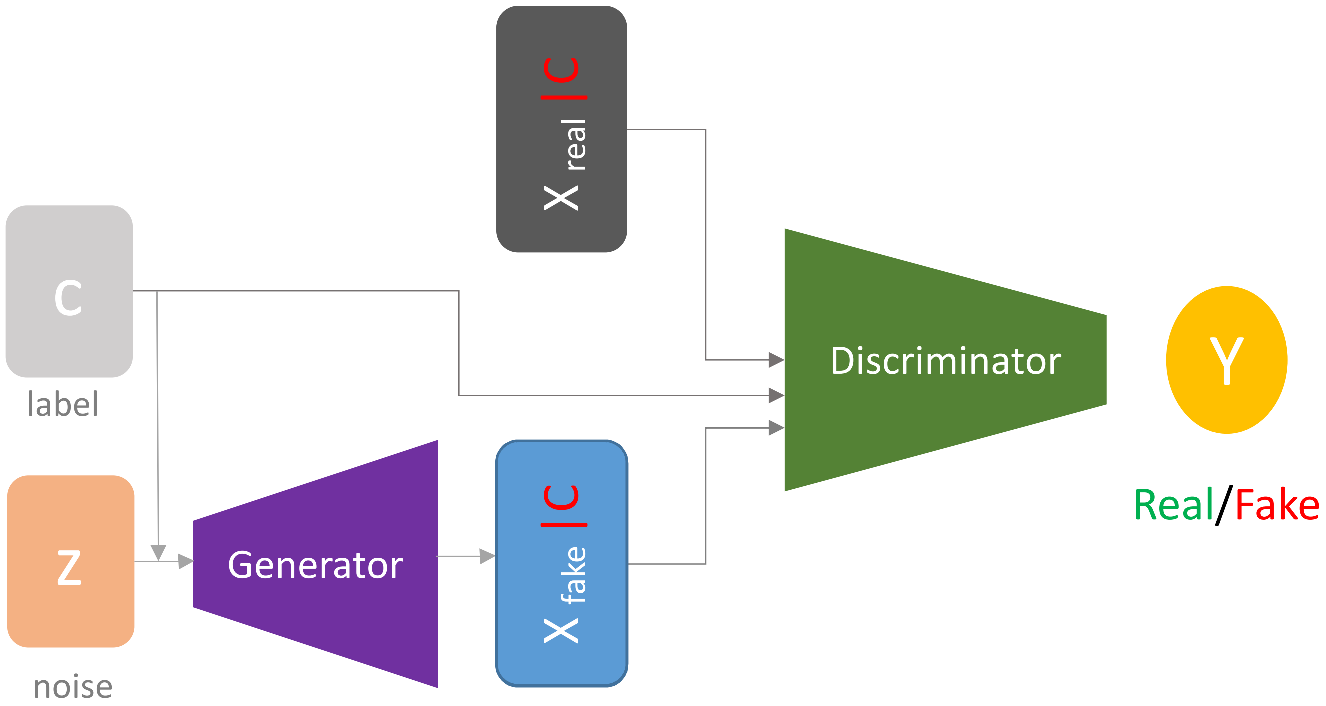
Conditional Generative Adversarial Nets

Mehdi Mirza
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7
mirzamom@iro.umontreal.ca

Simon Osindero
Flickr / Yahoo Inc.
San Francisco, CA 94103
osindero@yahoo-inc.com

Conditional Generative Adversarial Networks (cGANs)

- In cGANs, the generator G and the discriminator D are conditioned on
  - some extra information c
- The extra information could be class labels, text, or sketches



GANs VS cGANs Loss Functions

Generative Adversarial Loss – GANs

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

	real samples	generated samples
Discriminator	*correctly classify real data D(x) -> 1*	*correctly classify wrong data D(G(z)) -> 0*
Generator		*Generate sample similar to real ones D(G(z)) -> 1*

Conditional Generative Adversarial Loss – cGANs

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

Same objective function as GANs only condition y is added in both generator and discriminator to control the output

Noise-to-Image Translation – DCGAN

Proposed an architecture that's more stable in training GAN's and more likely to converge. (in 2015) (Without labels)

UNSUPERVISED REPRESENTATION LEARNING WITH DEEP CONVOLUTIONAL GENERATIVE ADVERSARIAL NETWORKS

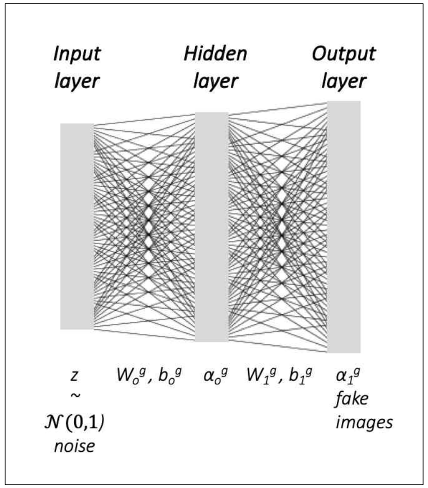
Alec Radford & Luke Metz
indico Research
Boston, MA
{alec,luke}@indico.io

Soumith Chintala
Facebook AI Research
New York, NY
soumith@fb.com

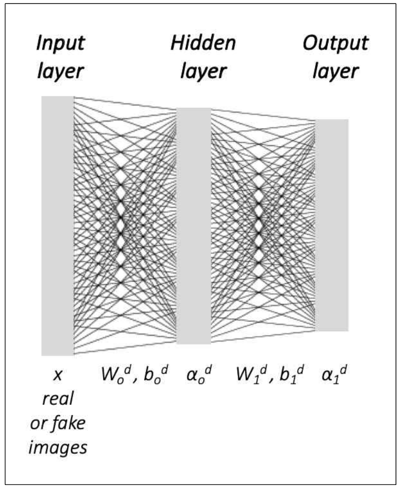
After extensive model exploration we identified a family of architectures that resulted stable training across a range of datasets and allowed for training higher resolution and deeper generative models.

DCGAN Contribution

DCGAN eliminates the need of fully connected layers in the network

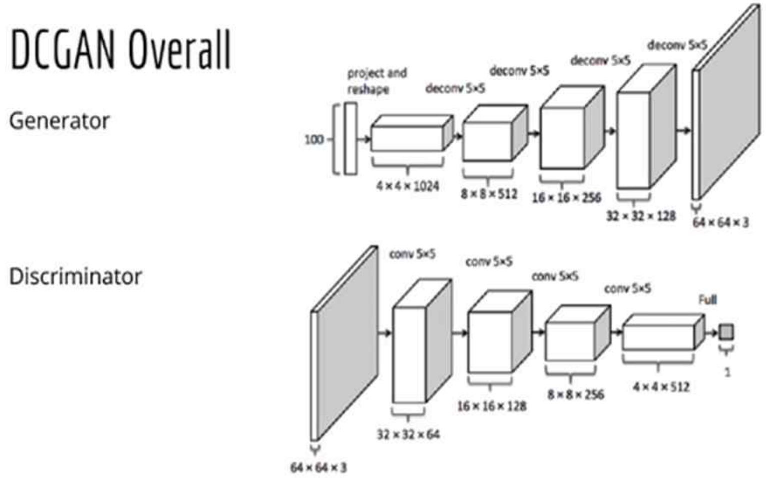


Generator

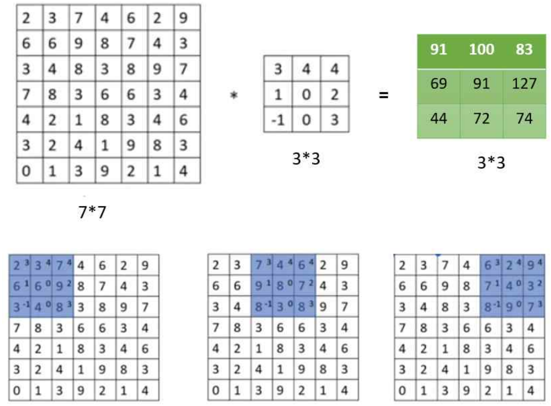


Discriminator

DCGAN architecture uses a standard CNN architecture on the discriminative model



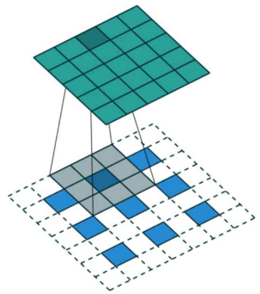
DCGAN Architectural Details



Architectural Changes:

1. *No Max Pooling:*
   - *Replace max pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).*

2. *Using Batch Normalization*
   - *Except the output layer for the generator and the input layer of the discriminator*
   - *This mainly tackles two problems in DCGAN and in deep neural networks in general:*
     - ✓ *It normalizes the input to each unit of a layer.*
     - ✓ *It also helps to deal with poor initialization that may cause problems in gradient flow.*

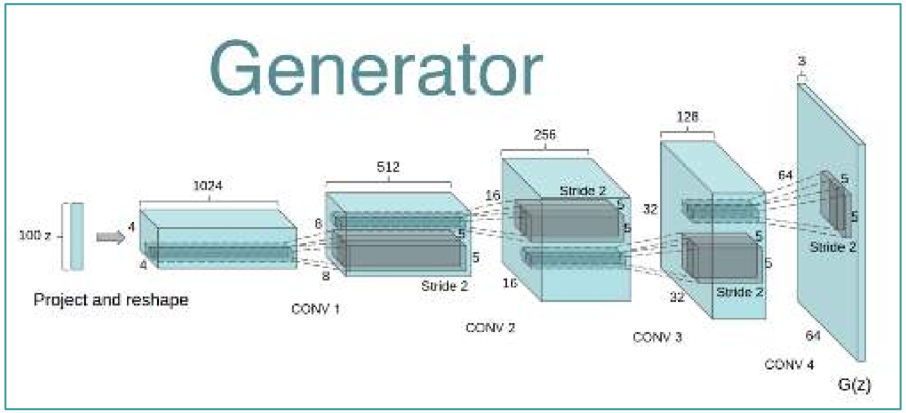


3. *Remove Fully Connected Layers*
   - *Remove fully connected hidden layers for deeper architecture*

4. *Use ReLU activation in Generator*
   - *In all the layers of the generator, except for the last one.*
   - *For the last convolutional layer, we will use Tanh activation function.*

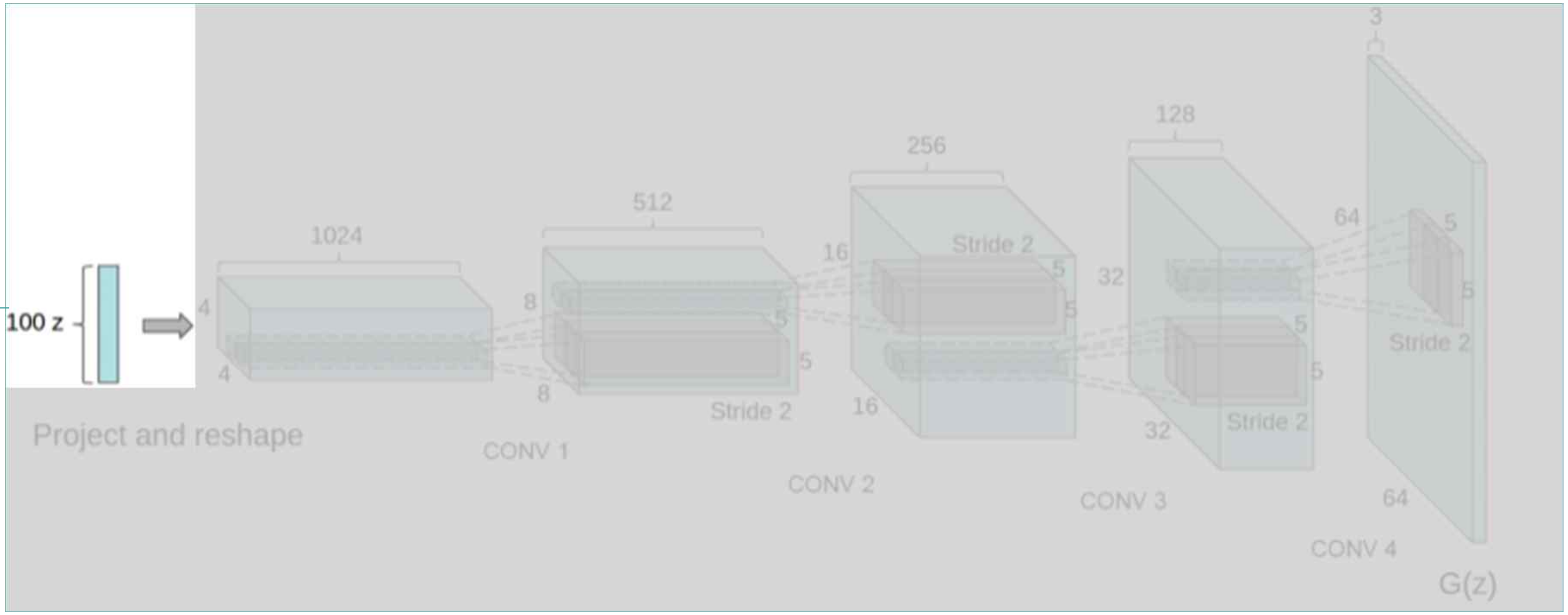
5. *Use LeakyReLU activation in Discriminator*
   - *Use LeakyReLU for all the convolutional layer after applying batch normalization.*

DCGAN Model Architecture



DCGAN Model Architecture - Generator

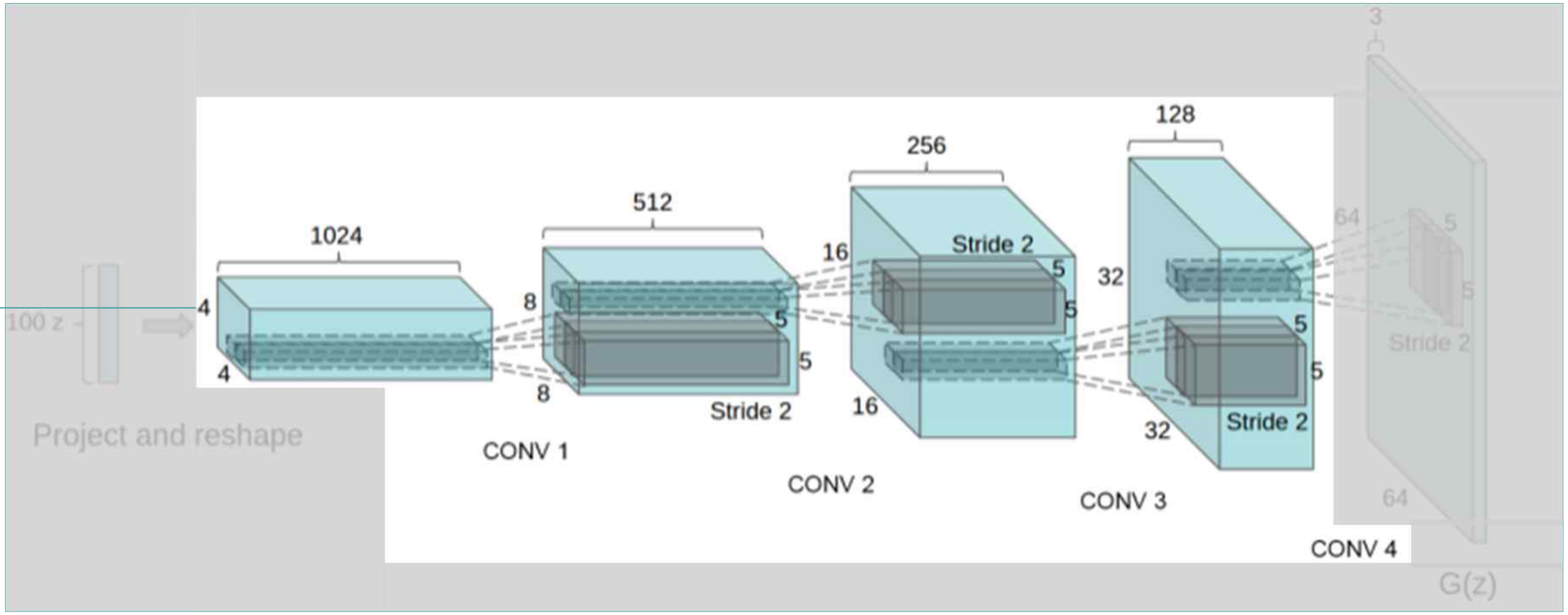
DCGAN generator workflow – Step 1



- Provide the generator a 100-dimensional noise vector as the input.
- After that, we project and reshape the input.

DCGAN Model Architecture - Generator

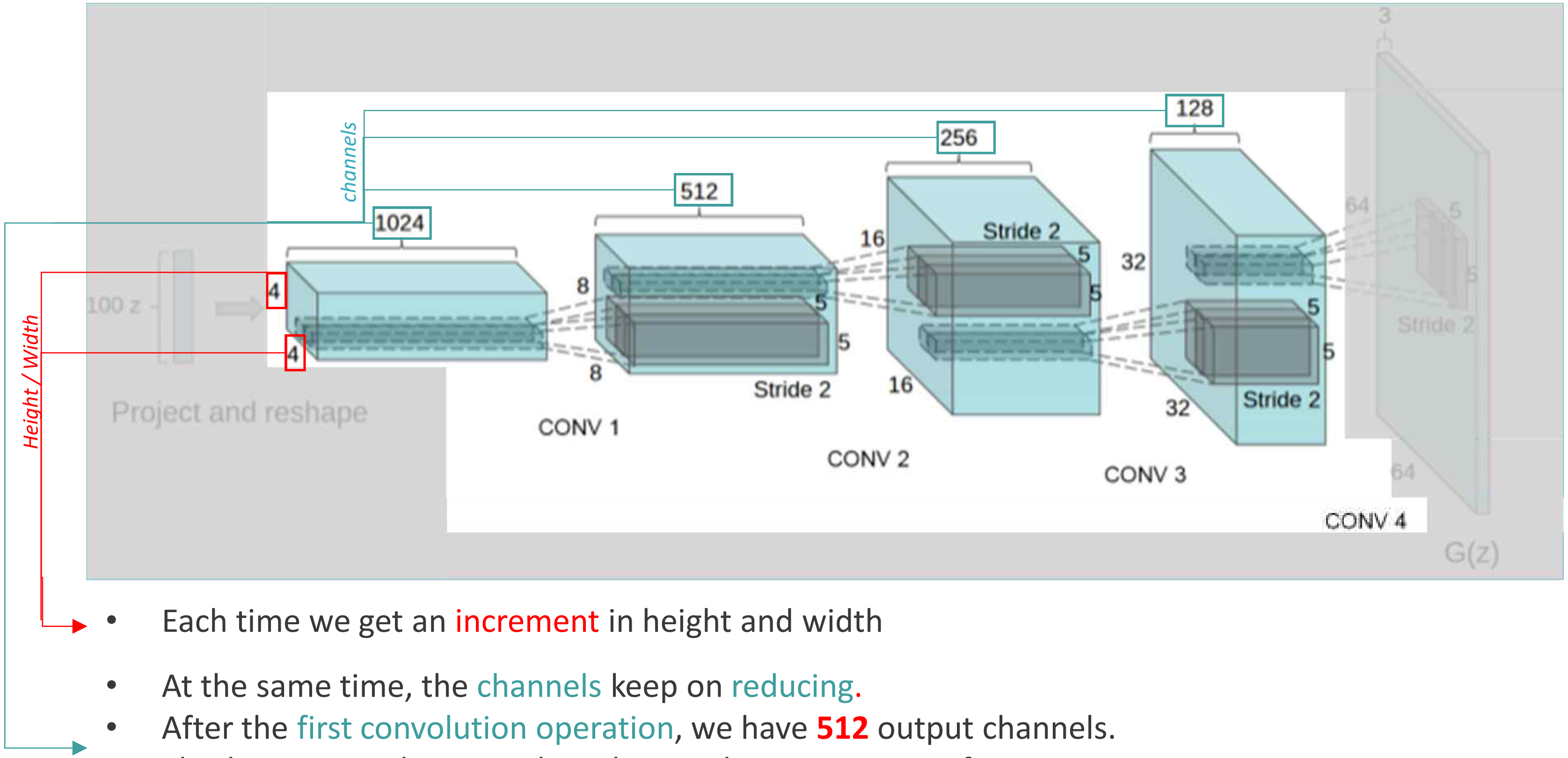
DCGAN generator workflow – Step 2



- After feeding input vector, projection, and reshape:
  - We have four convolutional operations to be performed

DCGAN Model Architecture - Generator

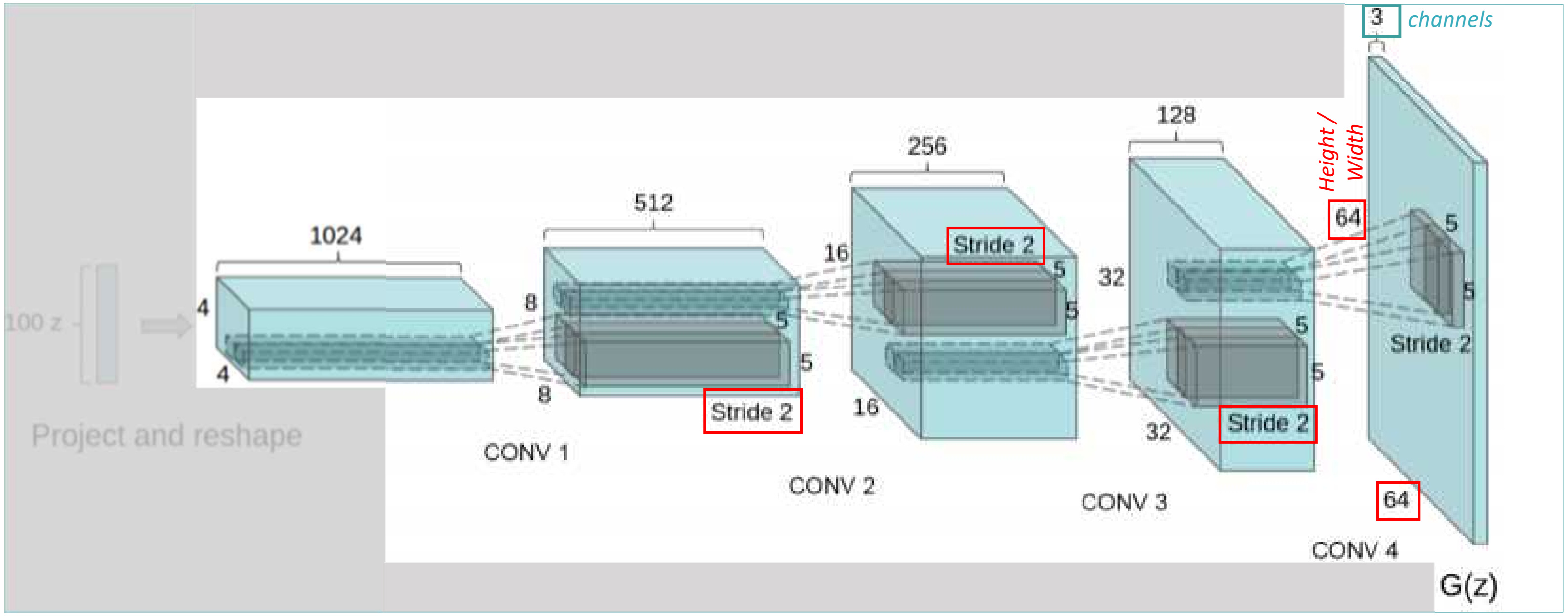
DCGAN generator workflow – Step 3



- Each time we get an increment in height and width
- At the same time, the channels keep on reducing.
- After the first convolution operation, we have **512** output channels.
- This keeps on reducing with each convolution operation from CONV1 to CONV4.
- After the third one, the output channels are **128**.

DCGAN Model Architecture - Generator

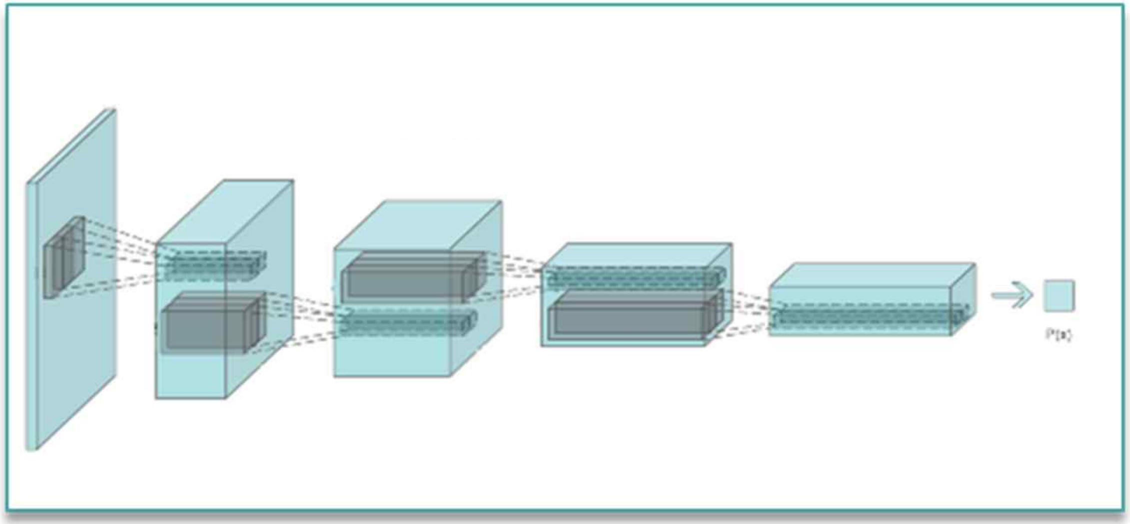
DCGAN generator workflow – Step 4



- By the end, we have a generated image of 64×64 dimensions and three output channels.
- Except for the first convolution layer, all the other layers have a stride of 2.

DCGAN Model Architecture - Discriminator

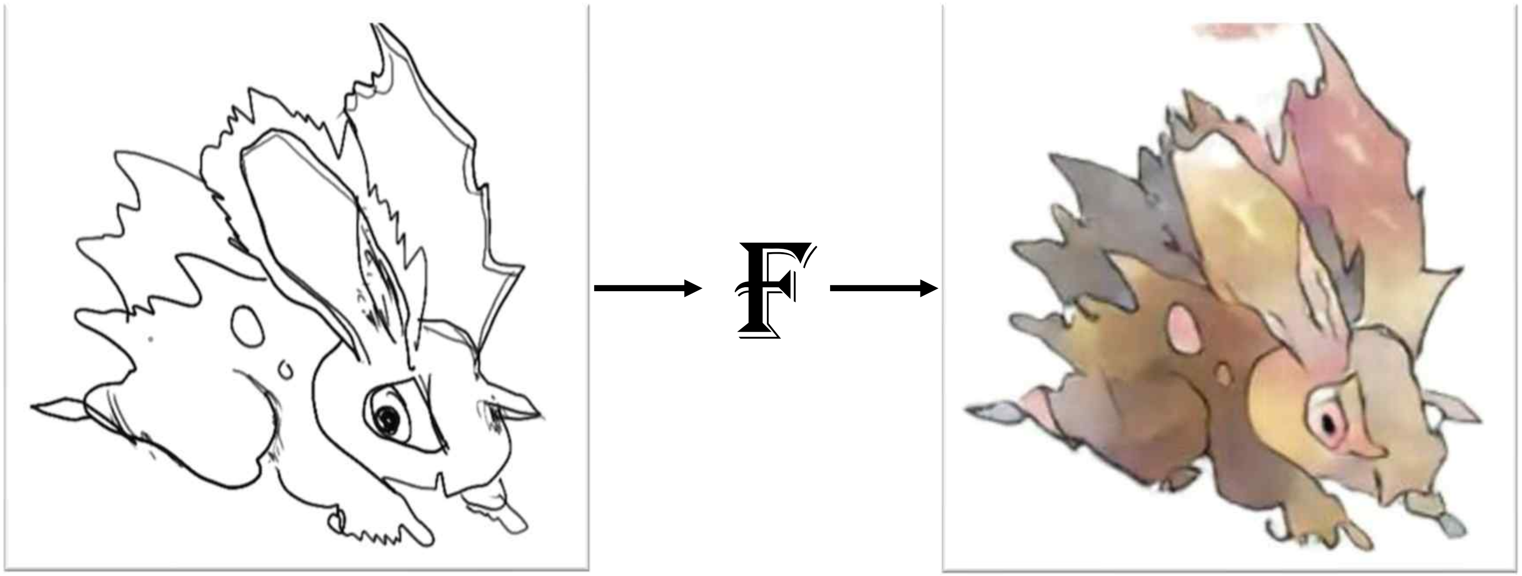
DCGAN discriminator workflow



- The discriminator takes an image as input from generator,
- Passes through convolution stacks, and
- Output a probability (sigmoid value) telling whether or not the image is real.

Image-to-Image (I2I) Translation

I2I is a class of vision and graphics problems where the goal is to learn the mapping between an input image and an output image.



Input Image

Translated Image

I2I Example use cases

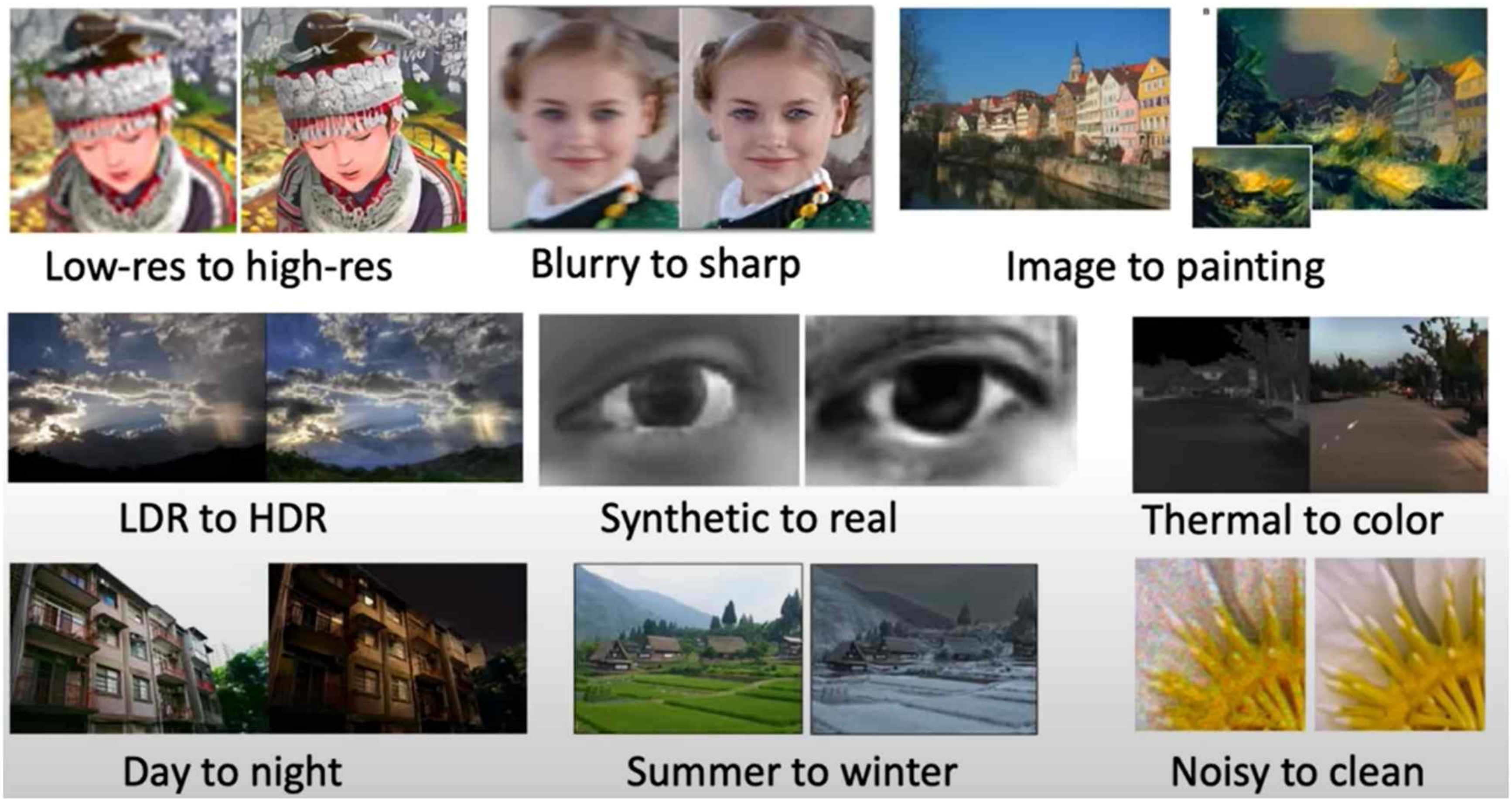


Image-to-Image Translation – Pix2pix

- Class of computer vision and graphics problems
- Goal: Learn the mapping between input image and output image
  - OR learning the translation from input domain DA to output domain DB
- This can be achieved by a Conditional GAN (cGAN) pix2pix

Image-to-Image Translation with Conditional Adversarial Networks

Phillip Isola   Jun-Yan Zhu   Tinghui Zhou   Alexei A. Efros

Berkeley AI Research (BAIR) Laboratory, UC Berkeley

{isola,junyanz,tinghuiz,efros}@eecs.berkeley.edu

Pix2pix Architecture

- pix2pix depends on paired training data

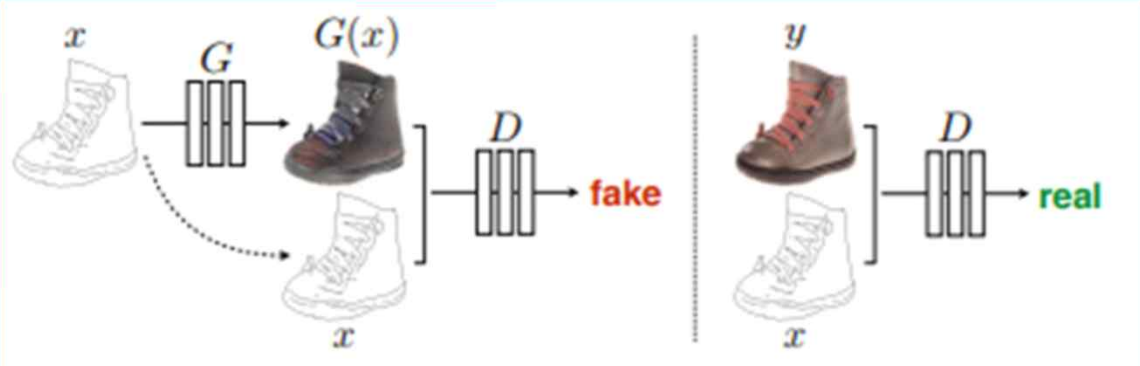
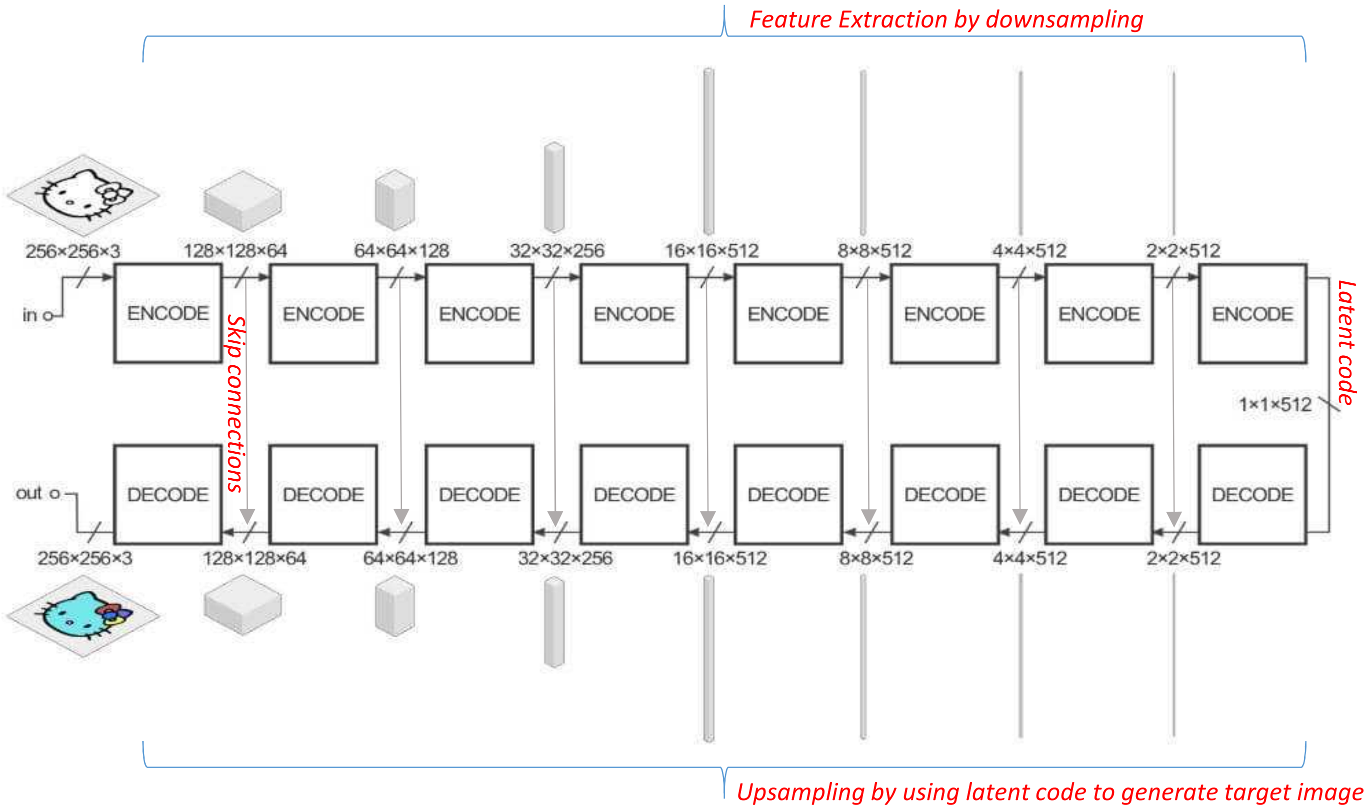


Figure 2: Training a conditional GAN to map edges→photo. The discriminator, D, learns to classify between fake (synthesized by the generator) and real {edge, photo} tuples. The generator, G, learns to fool the discriminator. Unlike an unconditional GAN, both the generator and discriminator observe the input edge map.

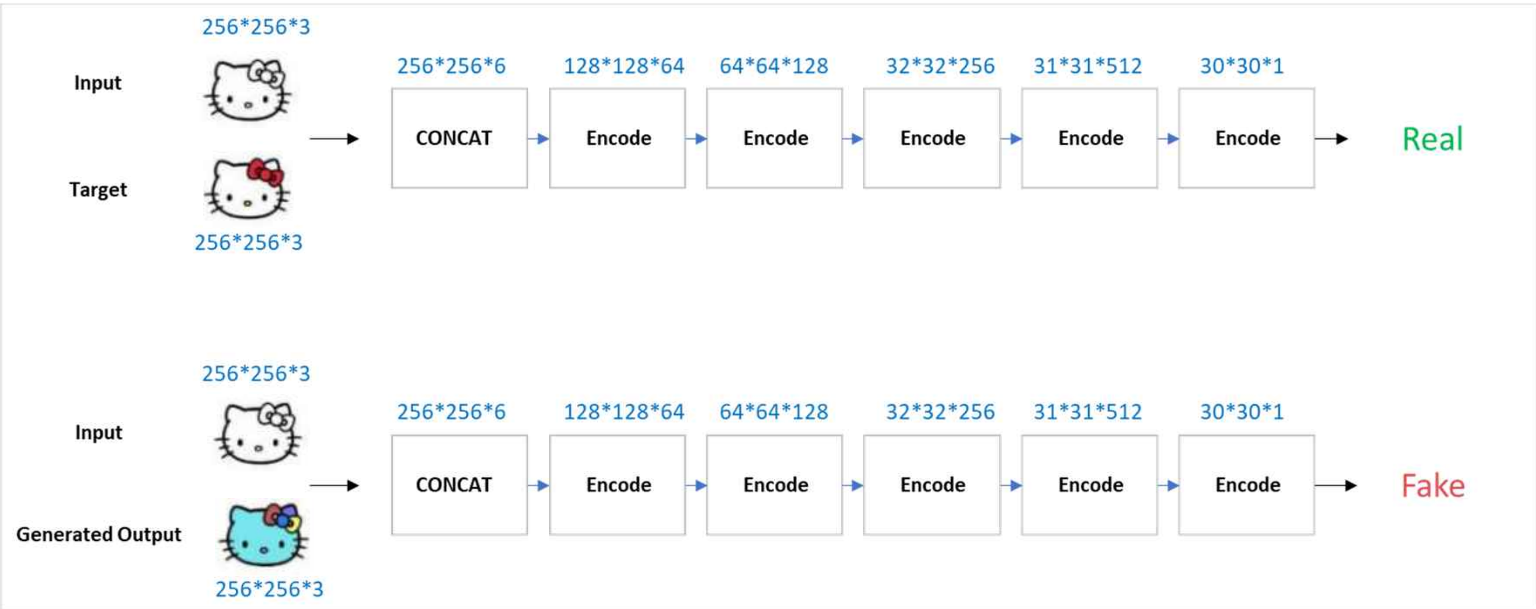
Pix2pix Architecture – Generator

- The structure of the generator is called U-Net based encoder-decoder with skip-connections
- Job of generator is:
  - taking an input image (could be black and white image)
  - performing the transform to produce the target image (be a colorized version)



Pix2pix Architecture – Discriminator

- The structure of the discriminator is based on PatchGAN
- The Discriminator has the job of taking two images:
    - an input image and
    - an unknown image (which will be either a target or output image from the generator) and
    - decide if the other image was produced by the generator or not



Pix2pix Objective Function

- Composite Adversarial and L1 Loss

Conditional Generative Adversarial Loss – cGANs

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z))],$$

L1 loss

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1].$$

Combined Loss = cGANs + Lambda * L1 loss

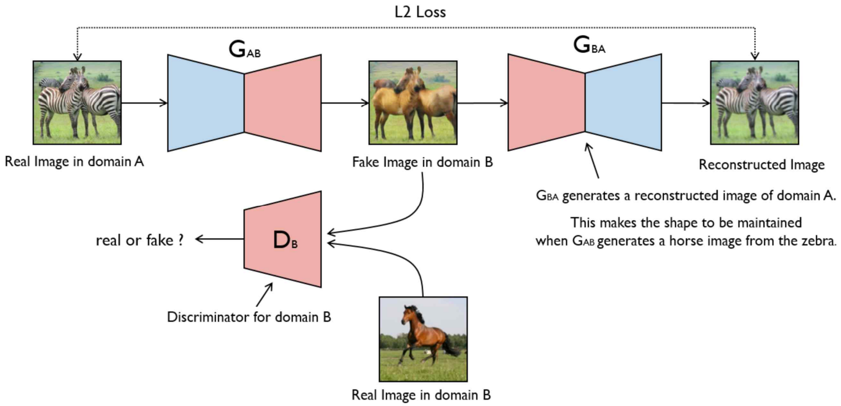
$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G)$$

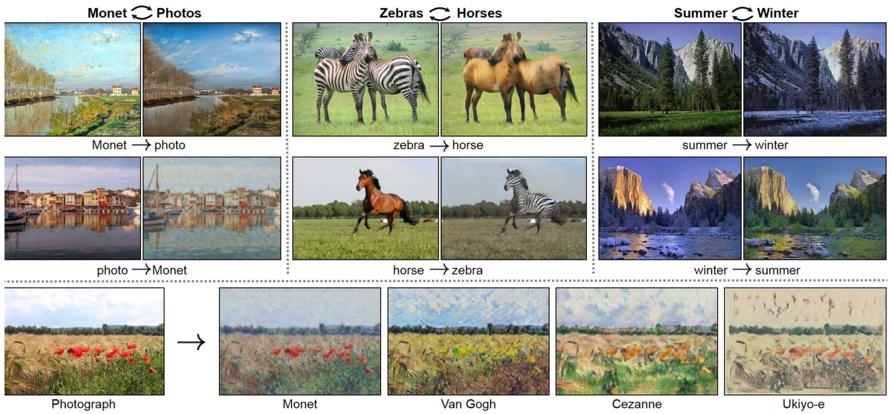
GANs: Recent Advances

Two imaginary celebrities that were dreamed up by a random number generator.

"Progressive Growing of GANs for Improved Quality, Stability, and Variation" by
Tero Karras (NVIDIA), Timo Aila (NVIDIA), Samuli Laine (NVIDIA), Jaakko Lehtinen (NVIDIA and Aalto University)

Domain transformation (CycleGAN)

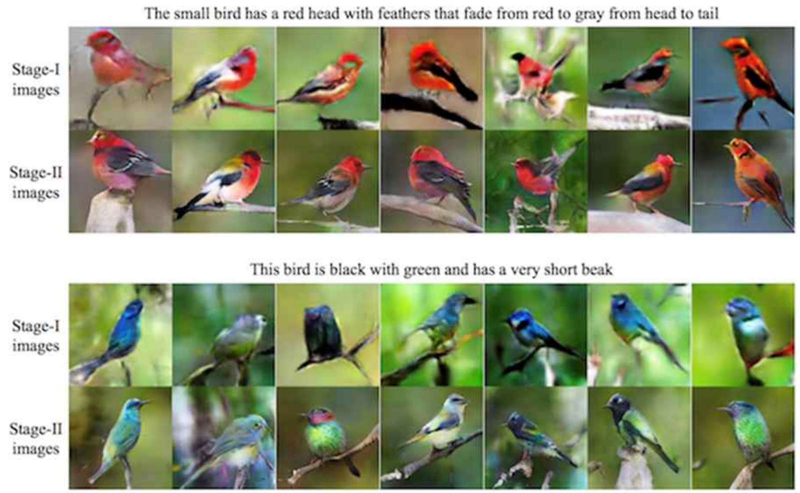




"Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks" by
Jun-Yan Zhu*, Taesung Park, Phillip Isola, Alexei A. Efros

"Image-to-Image Translation with Conditional Adversarial Nets" by
Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros

Text-to-Image Translation (Text2image)



"Generative Adversarial Text to Image Synthesis" by

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran

Hands-on Lab

Speaker: Ammar Ul Hassan

✓Font Generation using GANs

- Lab0- Setting up environment for running labs
- Lab1 - Building vanilla GAN in TensorFlow for MNIST
- Lab2 - Building DCGAN in TensorFlow for font dataset
- Lab3 - Building CDCGAN in TensorFlow for font dataset
- Lab4 - Building pix2pix in TensorFlow for font dataset

✓Our current research on font generation

Thank you

감사합니다