

Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>



deeplearning.ai

Generative Models

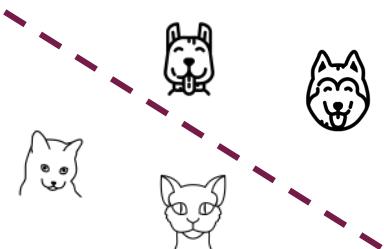
Outline

- What are generative models
- Types of generative models



Generative Models vs. Discriminative Models

Discriminative models



Features Class

$$X \rightarrow Y$$

$$P(Y|X)$$

Generative models



Noise Class Features

$$\xi, Y \rightarrow X$$

$$P(X|Y)$$

Generative Models vs. Discriminative Models



Generative models



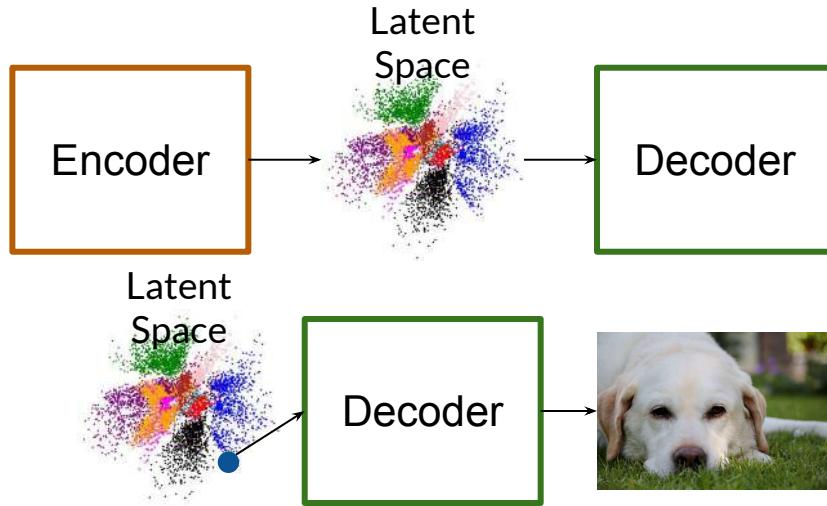
Noise Class Features

$$\xi, Y \rightarrow X$$

$$P(X|Y)$$

Generative Models

Variational Autoencoders

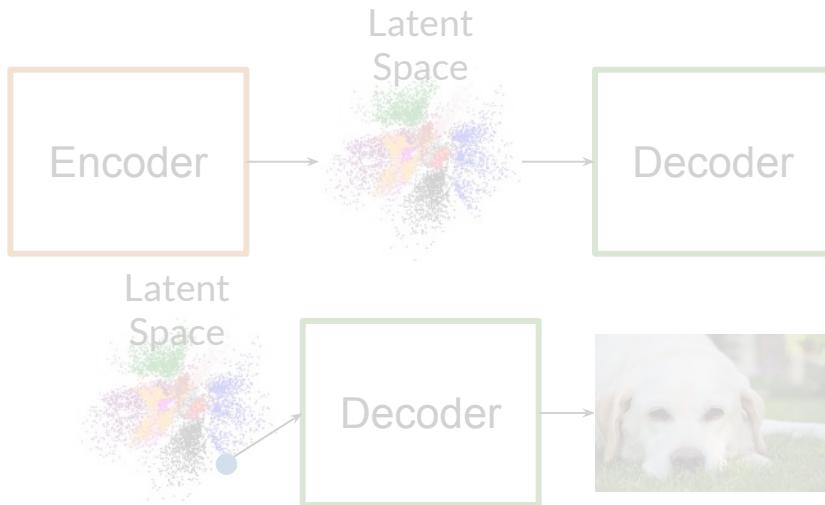


Generative Adversarial Networks

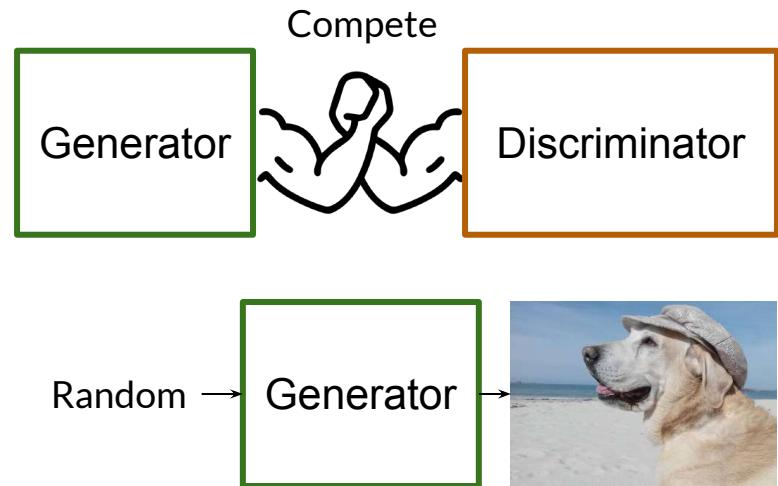
Available from: <https://arxiv.org/abs/1804.00891>

Generative Models

Variational Autoencoders



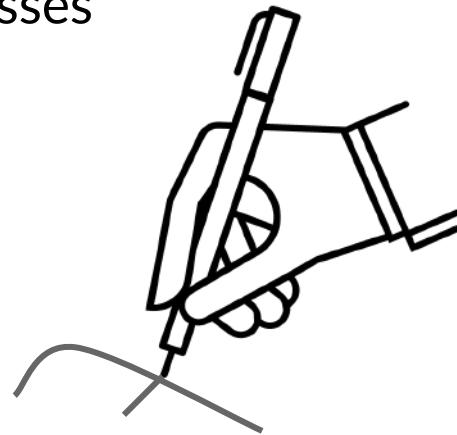
Generative Adversarial Networks



Available from: <https://arxiv.org/abs/1804.00891>

Summary

- Generative models learn to produce examples
- Discriminative models distinguish between classes
- Up next, GANs!



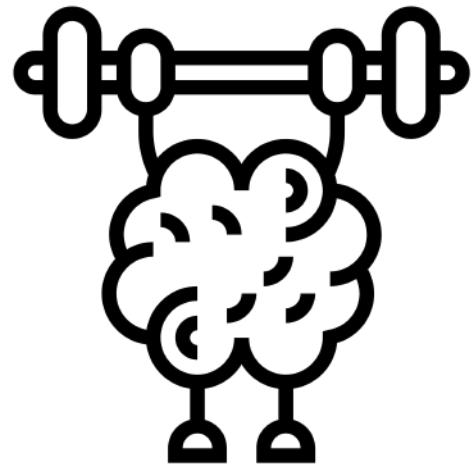


deeplearning.ai

Real Life GANs

Outline

- Cool applications of GANs
- Major companies using them



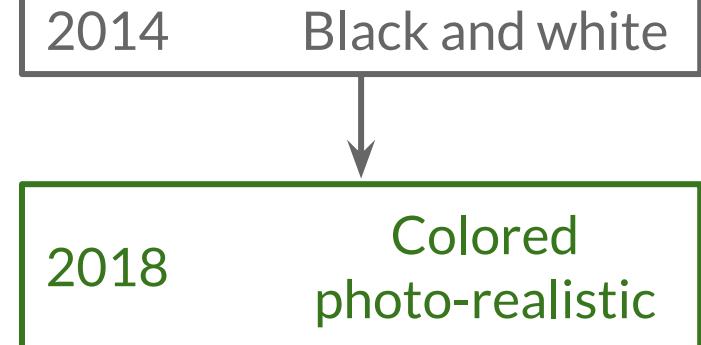
GANs Over Time



Ian Goodfellow
@goodfellow_ian

4.5 years of GAN progress on face generation.

arxiv.org/abs/1406.2661 arxiv.org/abs/1511.06434
arxiv.org/abs/1606.07536 arxiv.org/abs/1710.10196
arxiv.org/abs/1812.04948



GANs Over Time

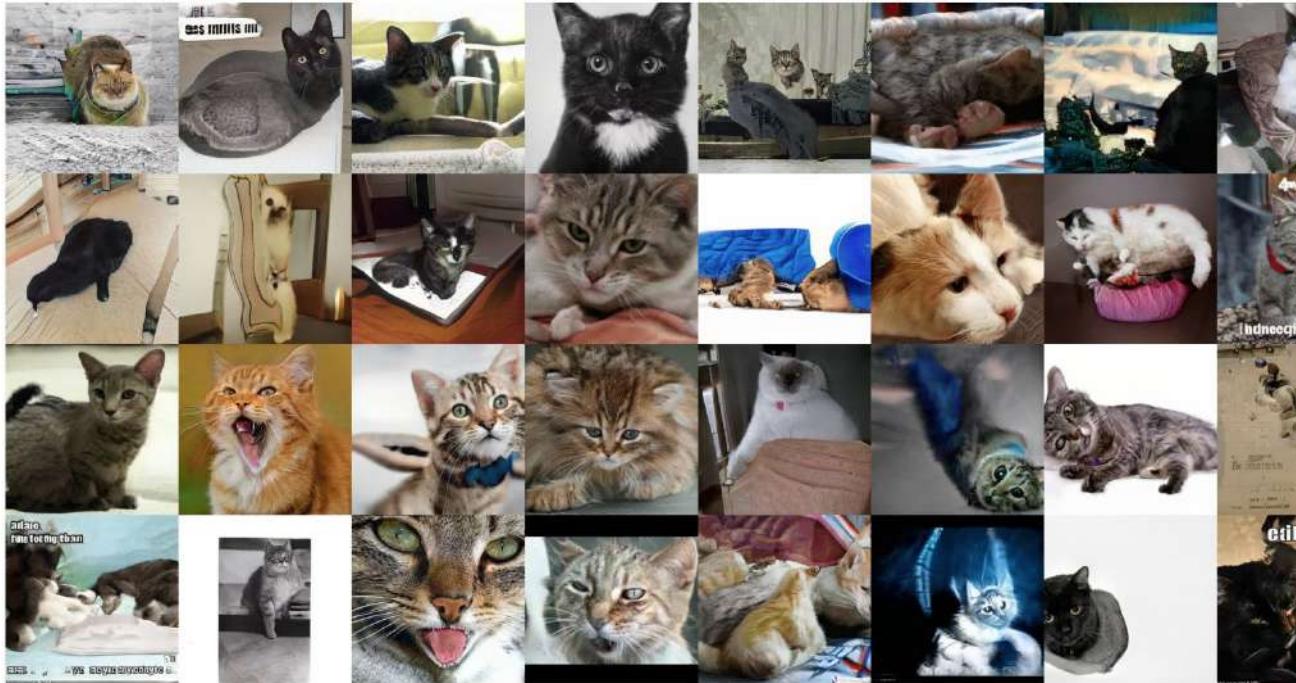


Face Generation
StyleGAN2

These people do
not exist!

Karras, Tero, et al. "Analyzing and improving the image quality of stylegan." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

GANs Over Time



StyleGAN2



Mimics the distribution of the training data

Karras, Tero, et al. "Analyzing and improving the image quality of stylegan." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

<https://9gag.com/gag/aWYZKWx>

GANs for Image Translation

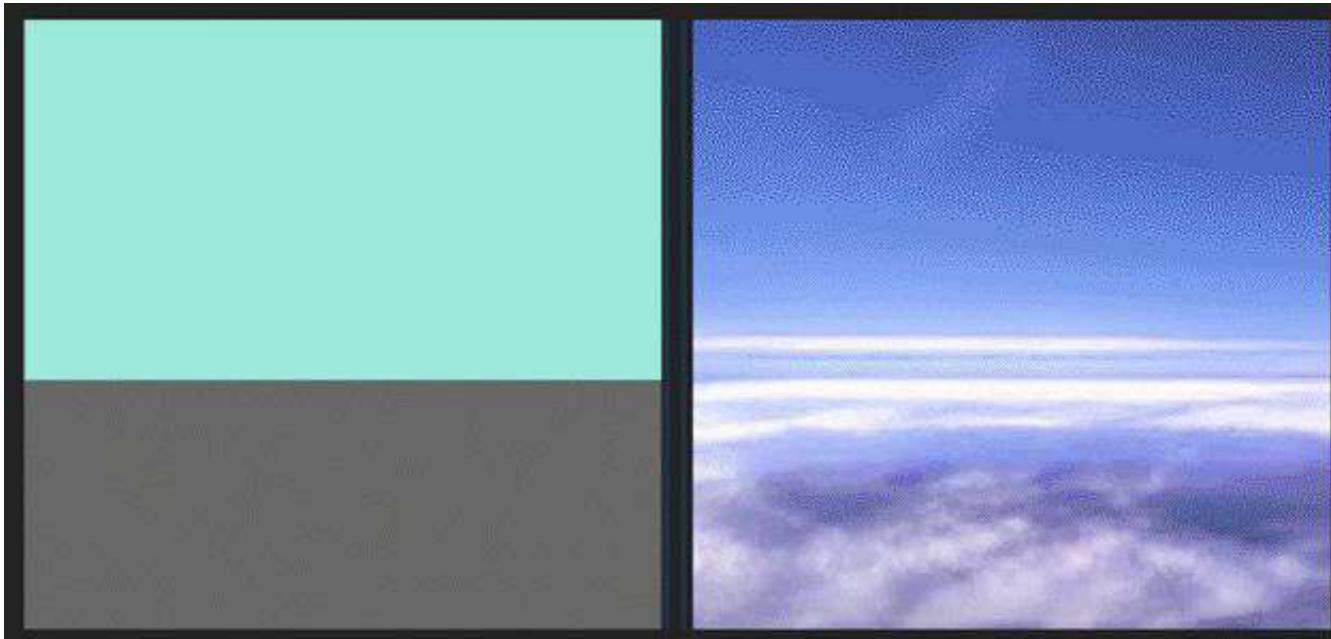
From one domain to another

CycleGAN



Park, Taesung, et al. "Semantic image synthesis with spatially-adaptive normalization." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

GANs for Image Translation

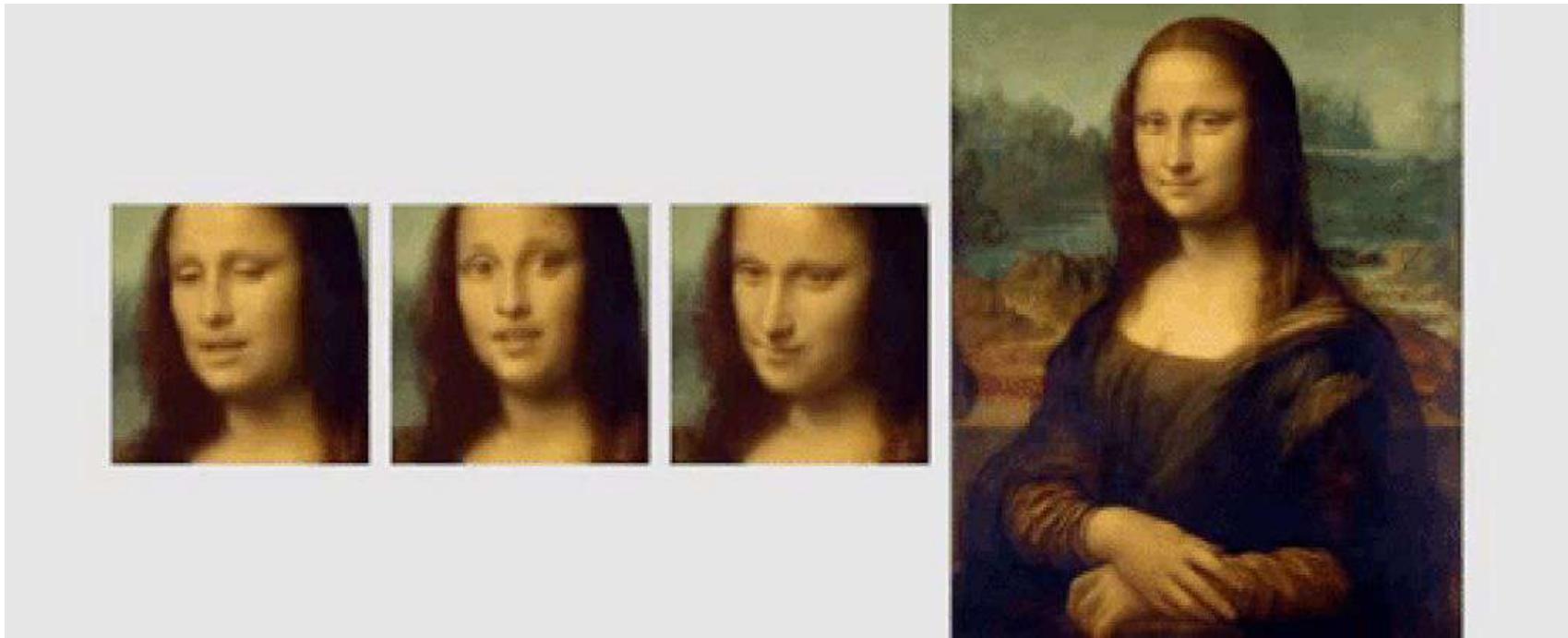


GauGAN

Doodles
↓
Pictures

Park, Taesung, et al. "Semantic image synthesis with spatially-adaptive normalization." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019.

GANs are Magic!



Zakharov, Egor, et al. "Few-shot adversarial learning of realistic neural talking head models." *Proceedings of the IEEE International Conference on Computer Vision*. 2019.

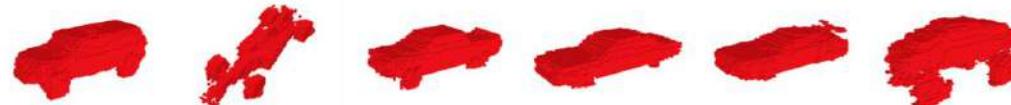
GANs for 3D Objects

Chair

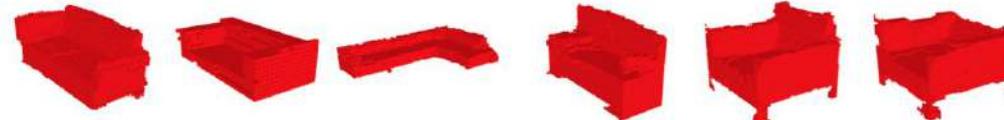


3D-GAN

Car



Sofa



Table



Generative
Design

Wu, Jiajun, et al. "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling." *Advances in neural information processing systems*. 2016.

Companies Using GANs



Next-gen
Photoshop



Text
Generation



Data
Augmentation



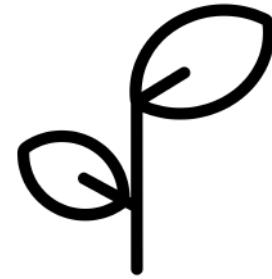
Image Filters



Super-resolution

Summary

- GANs' performance is rapidly improving
- Huge opportunity to work in this space!
- Major companies are using them





deeplearning.ai

Intuition Behind GANs

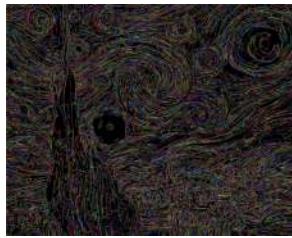
Outline

- The goal of the generator and the discriminator
- The competition between them

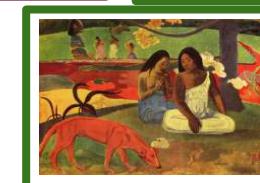
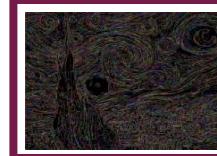


Generative Adversarial Network

Generator learns to make *fakes* that look **real**



Discriminator learns to distinguish **real** from *fake*



Generative Adversarial Network

Discriminator learns to distinguish
real from *fake*

Fake

Real



Generative Adversarial Network

Generator learns to make *fakes*
that look **real**

Discriminator learns to distinguish
real from *fake*



Generative Adversarial Network

Generator learns to make *fakes*
that look **real**



I don't know what I'm doing

Doesn't know how
it should look



Generative Adversarial Network

Discriminator learns to distinguish
real from *fake*



I don't know what I'm doing, either

The Game Is On!



5% Real



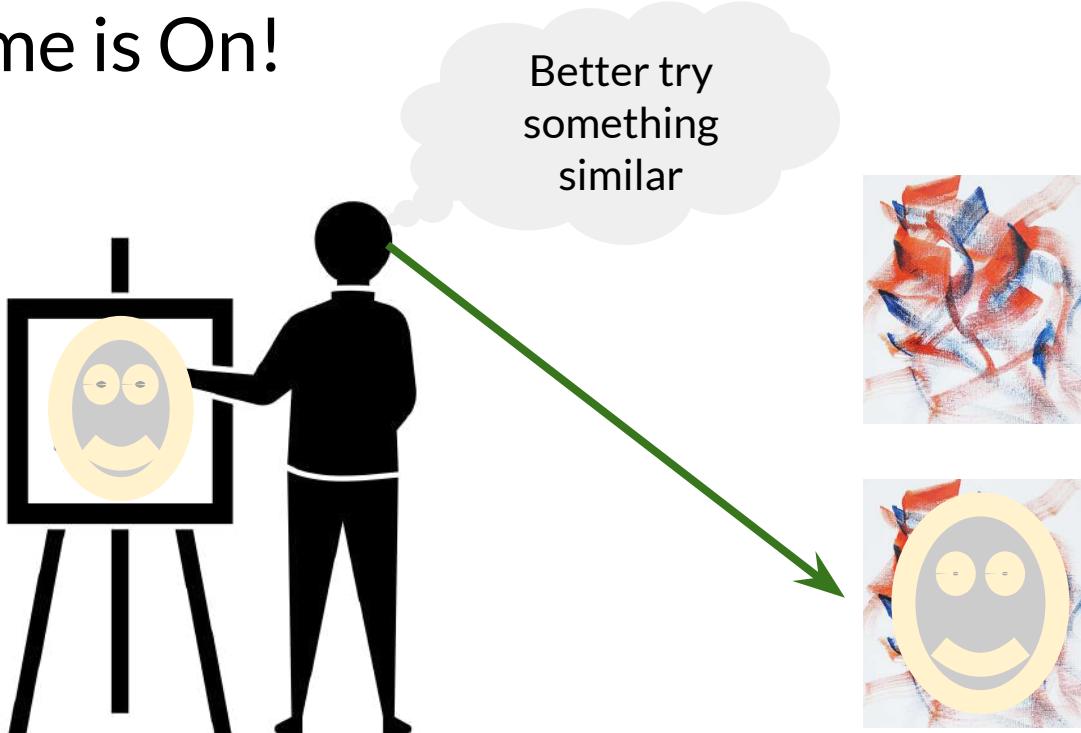
40% Real



80% Real



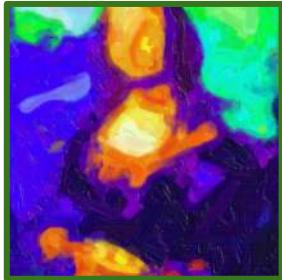
The Game is On!



The Game Is On!



30% Real



60% Real

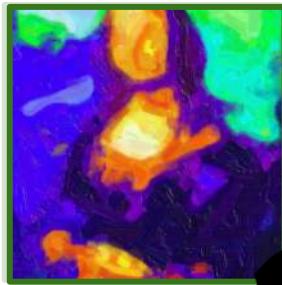


95% Real

The Game Is On!

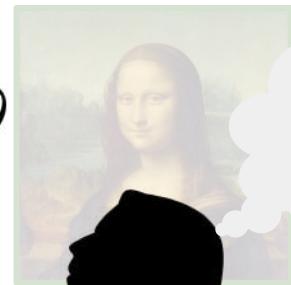


30% Real



60% Real

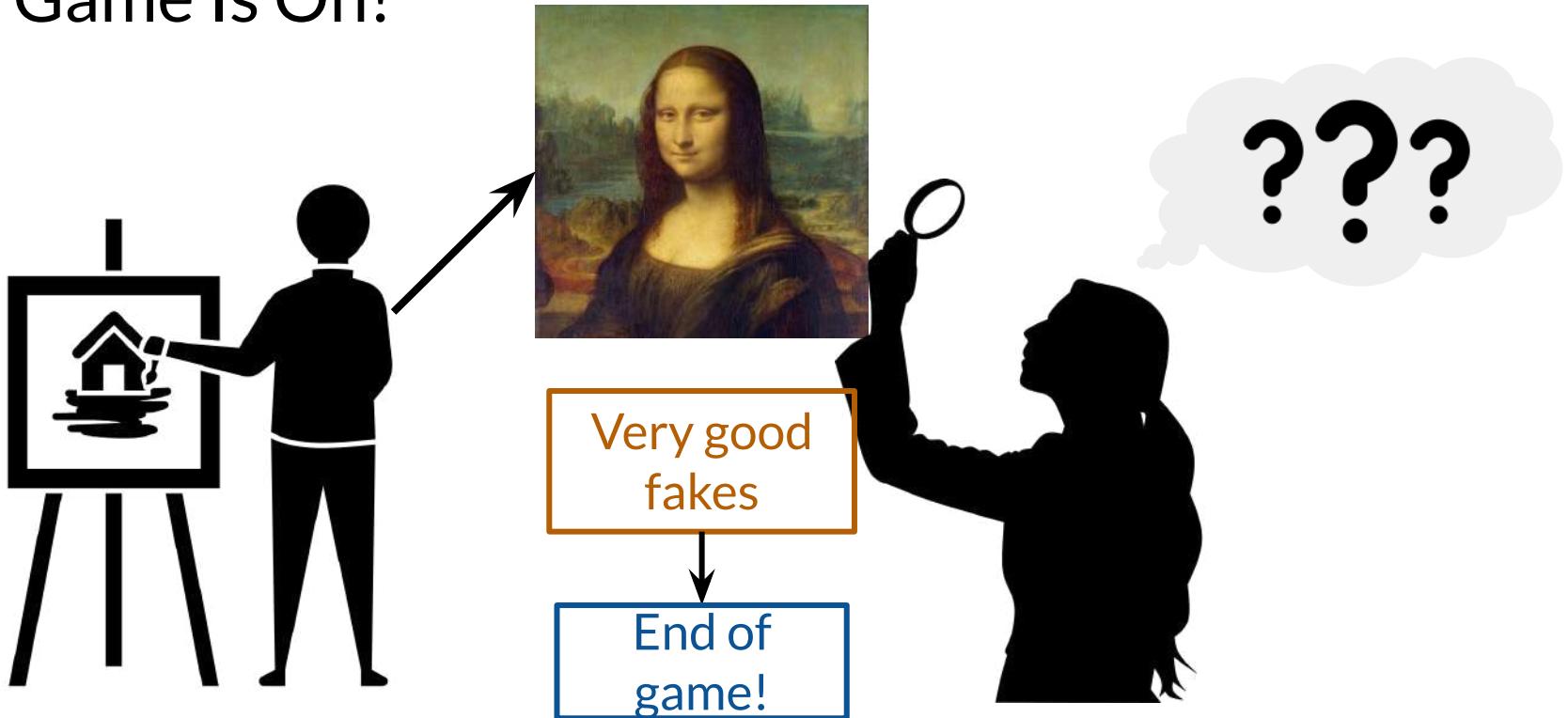
Wrong!



Won't fool me
again



The Game Is On!



Summary

- The generator's goal is to fool the discriminator
- The discriminator's goal is to distinguish between real and fake
- They learn from the competition with each other
- At the end, *fakes* look real



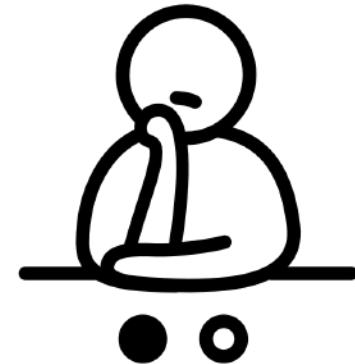


deeplearning.ai

Discriminator

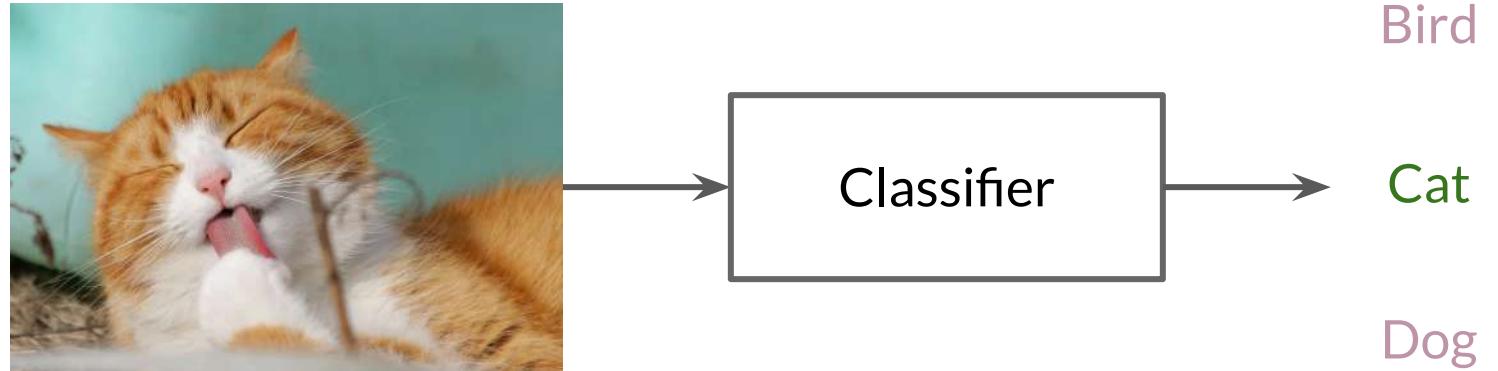
Outline

- Review of classifiers
- The role of classifiers in terms of probability
- Discriminator



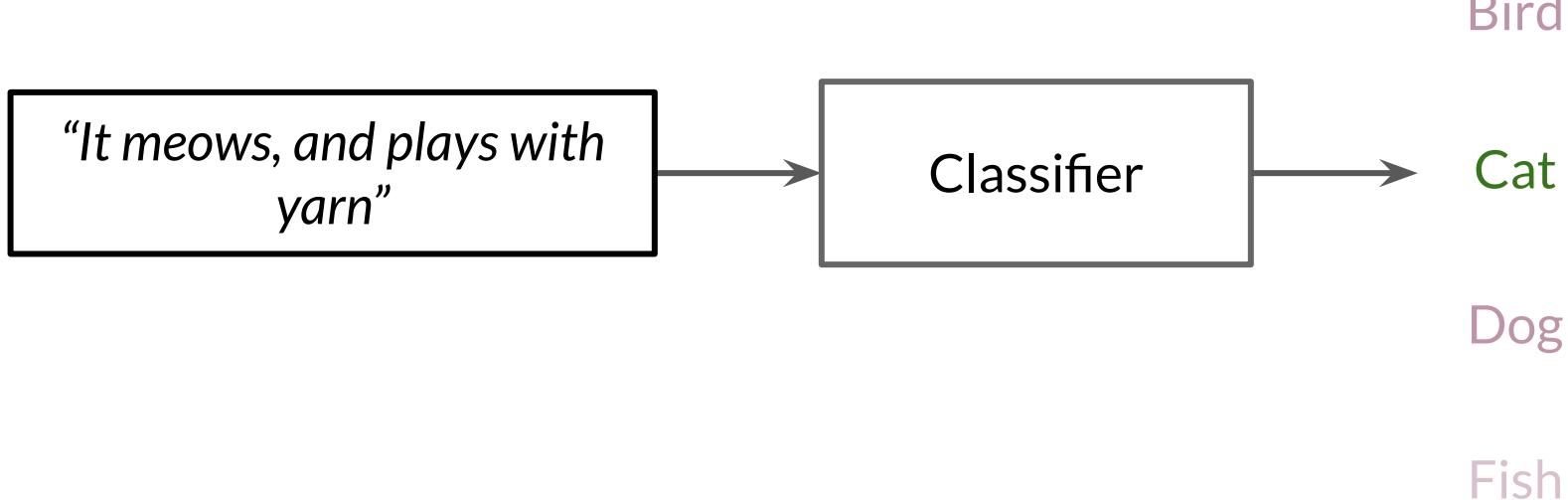
Classifiers

Distinguish between different classes

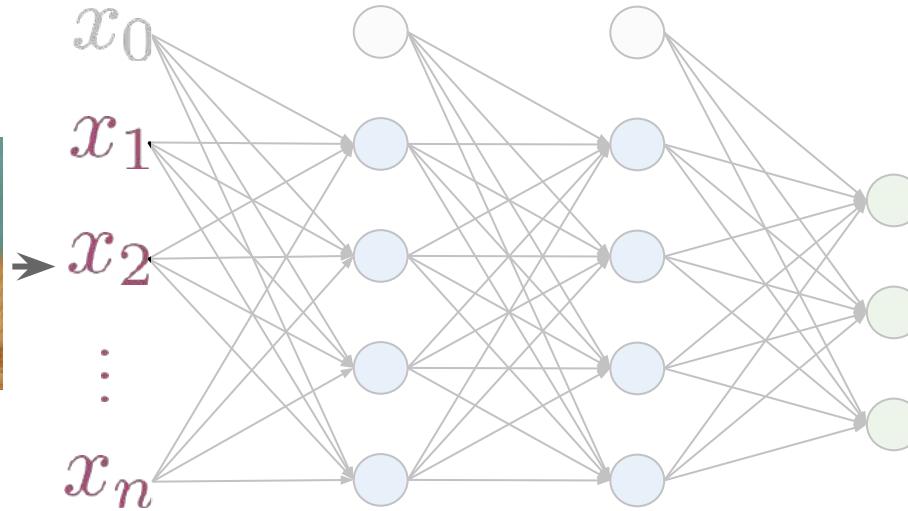


Classifiers

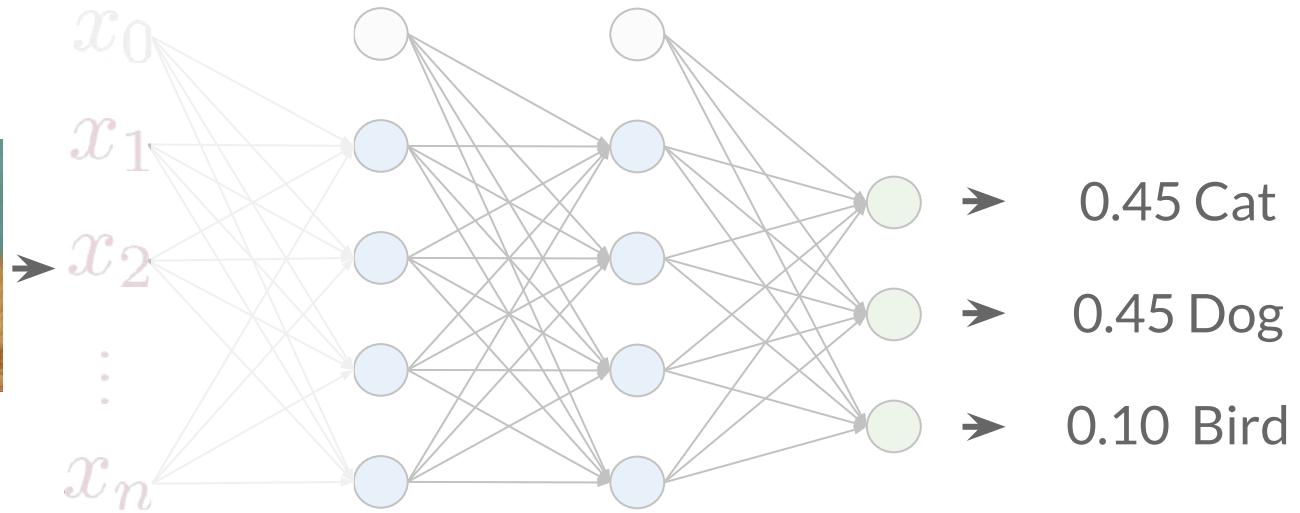
Distinguish between different classes



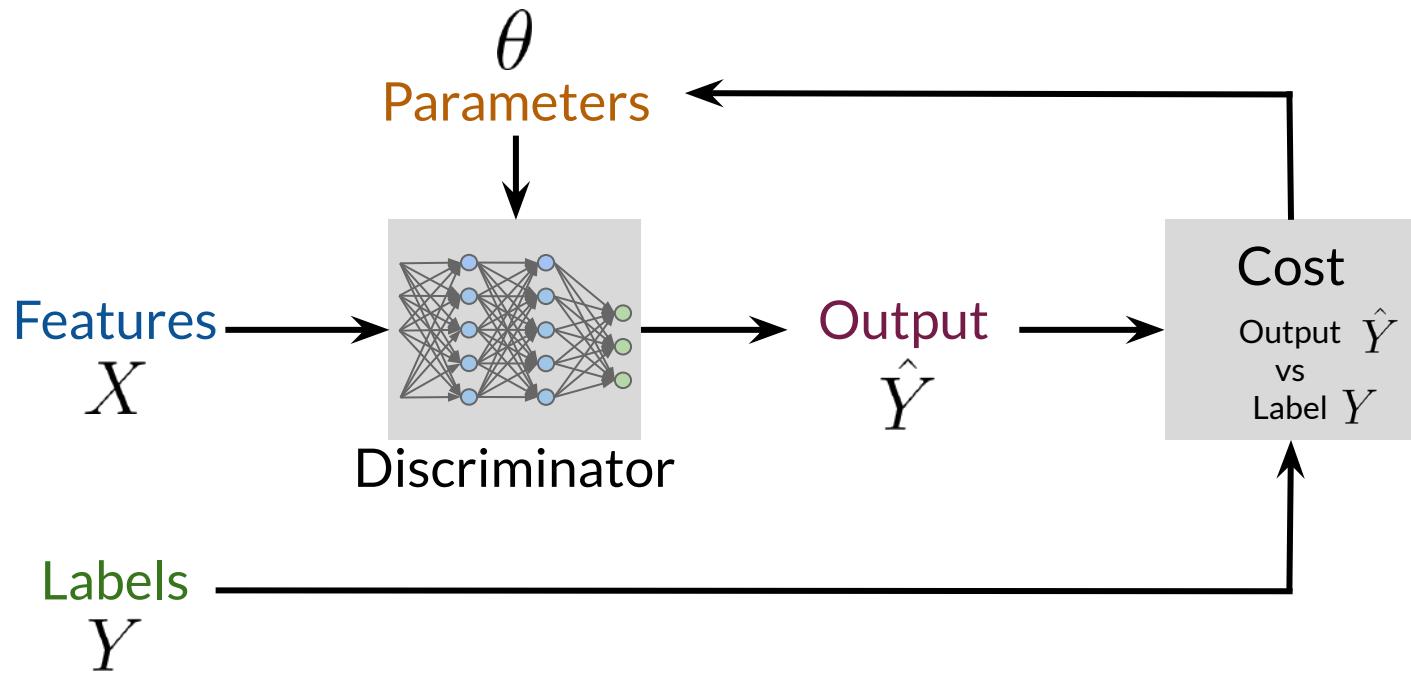
Neural Networks



Neural Networks



Classifiers (training)



Classifiers

Turtle

Bird

$$P(\text{Cat} \mid \text{ })$$


Dog

Fish

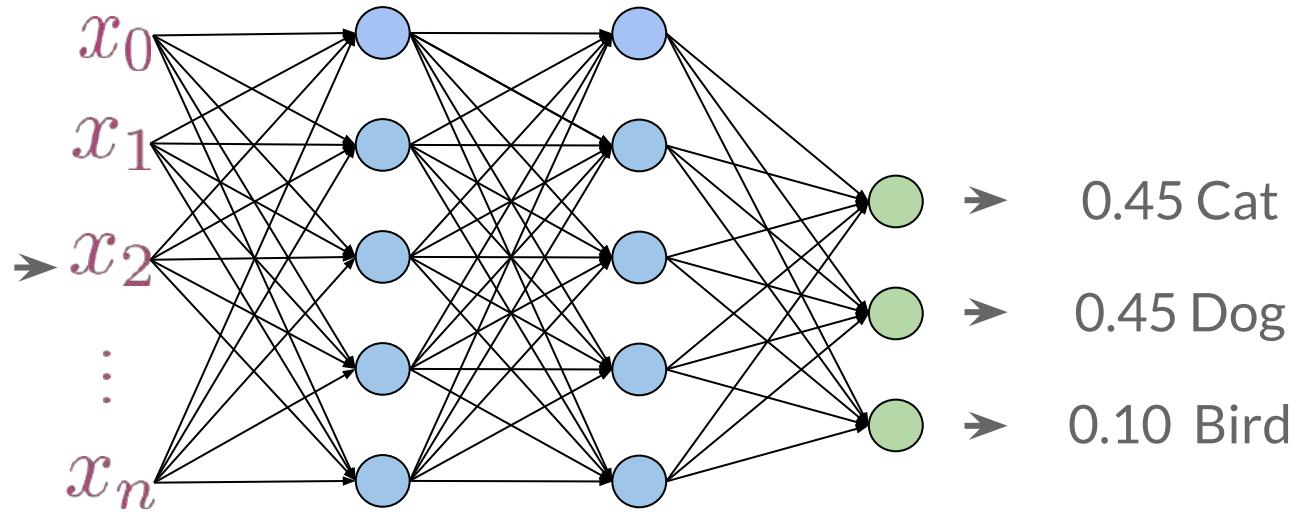
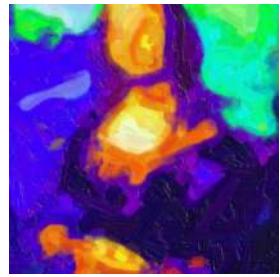
Classifiers

$$P(Y | X)$$

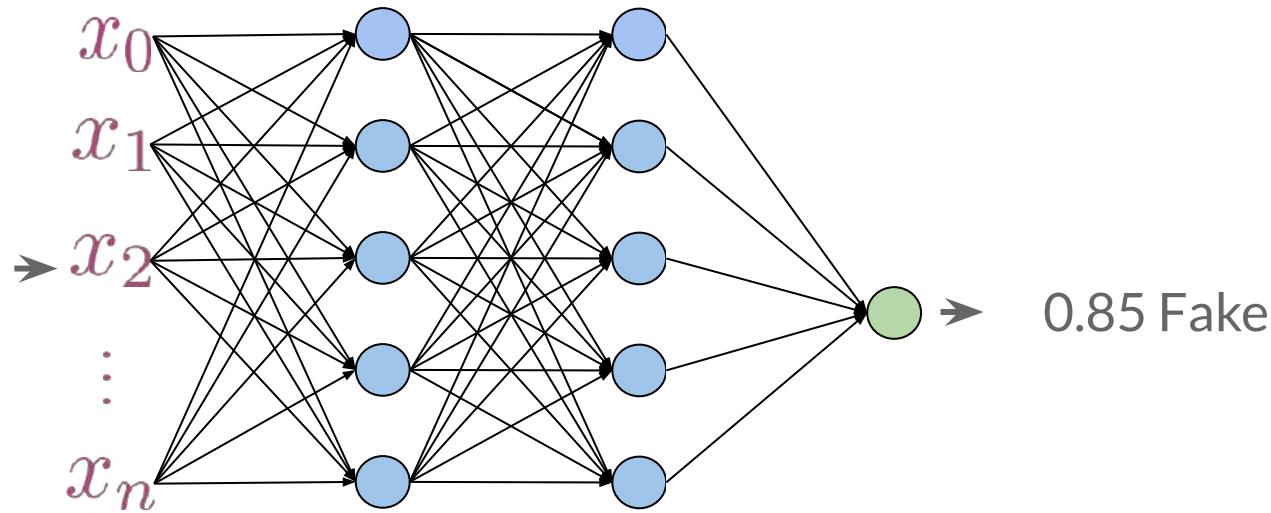
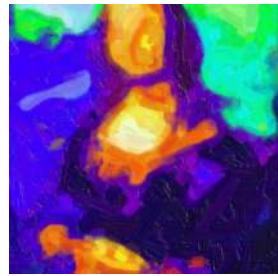
Class Features

A diagram illustrating the components of a classifier. The formula $P(Y | X)$ is shown, where Y is colored purple and X is colored blue. Below the formula, the word "Class" is aligned under Y and the word "Features" is aligned under X . An arrow points from the vertical bar separating Y and X to a rectangular box with a brown border. Inside the box, the text "Conditional Probability" is written in brown.

Discriminator



Discriminator



Discriminator

$$P(\text{Fake} \mid X)$$

Class Features

Discriminator

$$P(\text{Fake} \mid \begin{matrix} \text{Class} \\ | \\ \text{Features} \end{matrix}) = 0.85 \rightarrow \boxed{\text{Fake}}$$

Summary

- The **discriminator** is a classifier
- It learns the probability of class Y (**real** or **fake**) given features X
- The probabilities are the feedback for the **generator**





deeplearning.ai

Generator

Outline

- What the generator does
- How it improves its performance
- Generator in terms of probability



Generator

Turtle

Generates examples of the class

Bird

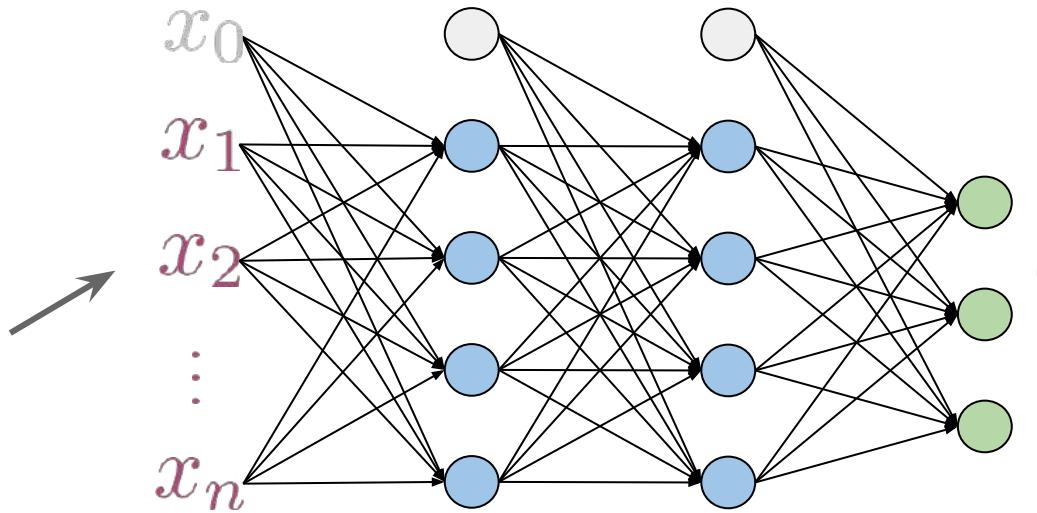
Cat

Dog

Fish

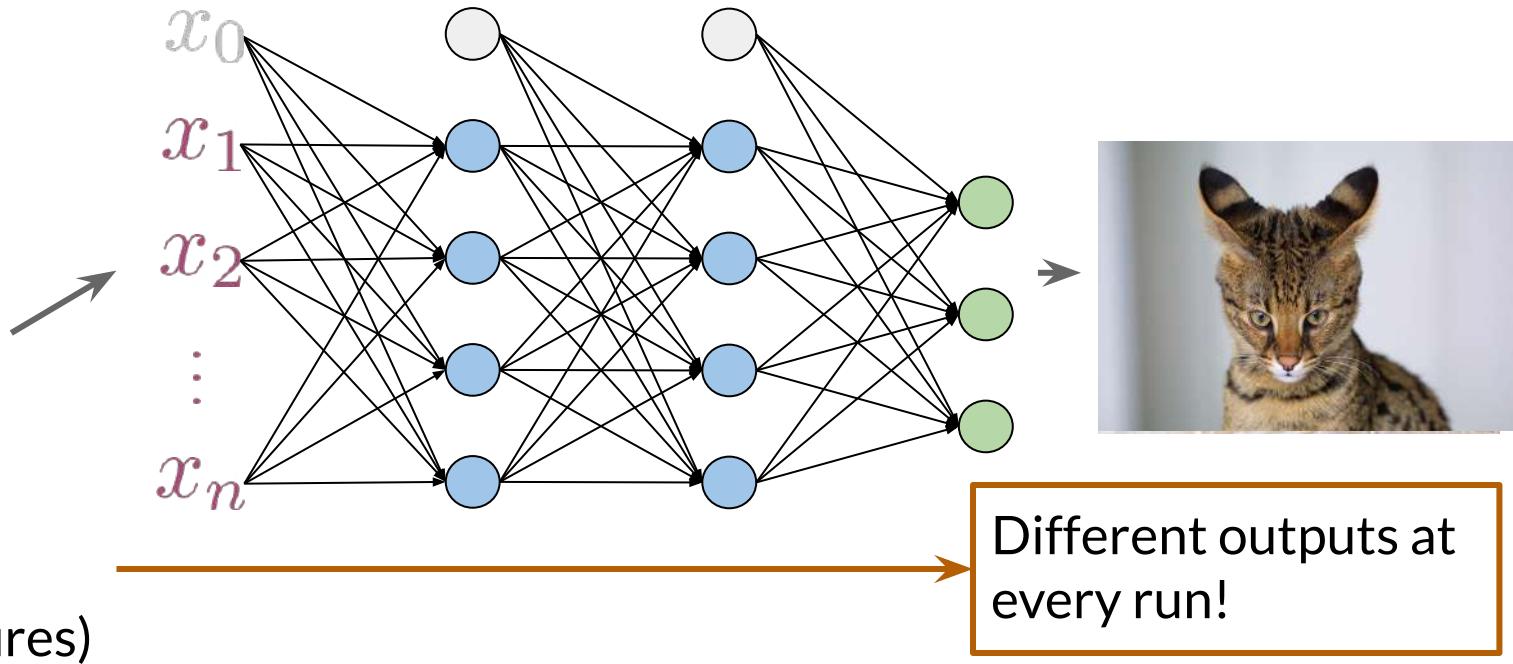


Neural Networks

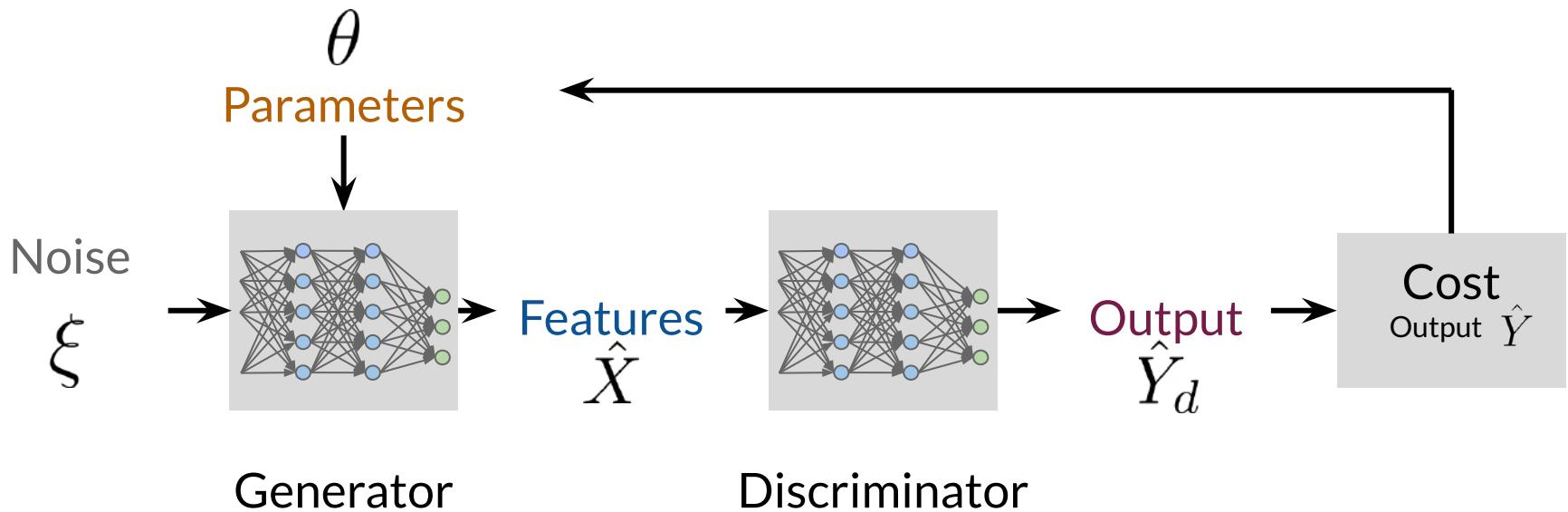


Noise
(random features)

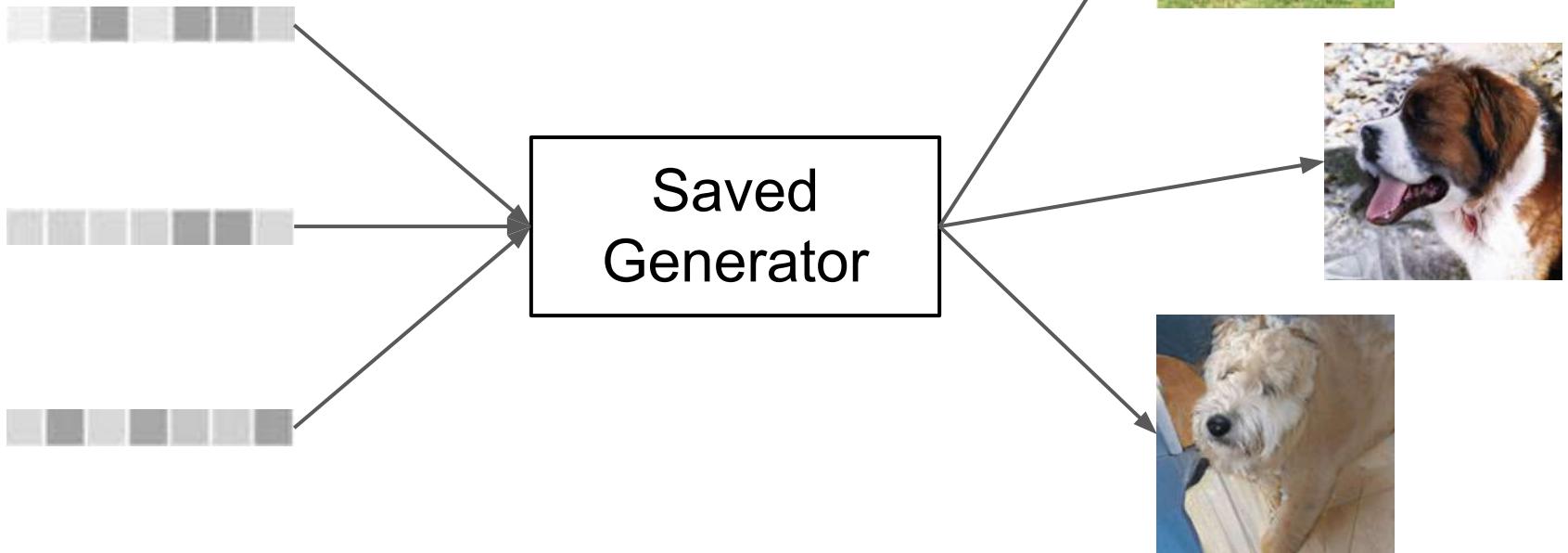
Neural Networks



Generator: Learning



Sampling



Generator

$$P\left(\begin{array}{c|c} \text{Image} & \text{Label} \\ \hline \text{Cat} & \text{Cat} \\ \text{Dog} & \text{Dog} \\ \text{Bird} & \text{Bird} \\ \text{Turtle} & \text{Turtle} \end{array}\right)$$


Generator

$$P(X \mid Y)$$

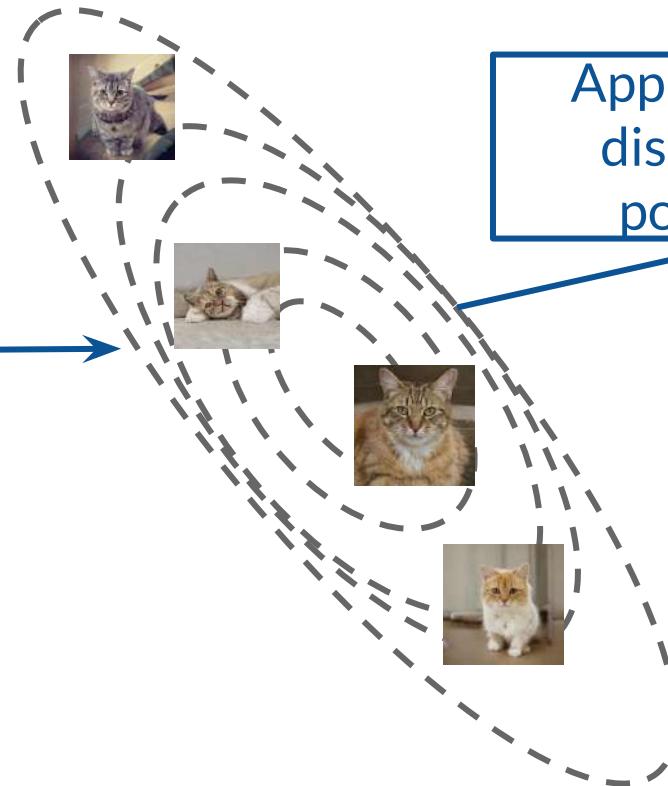
Features Class

A diagram illustrating a conditional probability formula. The formula is $P(X \mid Y)$. Below the formula, the word "Features" is aligned under the variable X , and the word "Class" is aligned under the variable Y . A grey arrow points from the vertical bar (\mid) in the formula to a rectangular box with a brown border. Inside the box, the text "Conditional Probability" is written in brown.

Generator

$$P(X)$$

Features



Images available from: <http://thesecatsdonotexist.com/>

Summary

- The **generator** produces fake data
- It learns the probability of features X
- The **generator** takes as input noise (random features)



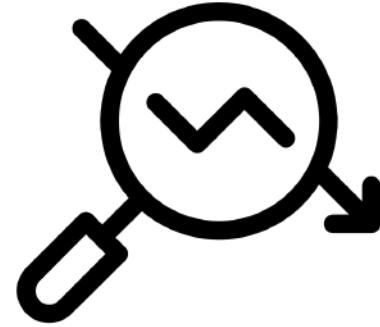


deeplearning.ai

BCE Cost Function

Outline

- Binary Cross Entropy (BCE) Loss equation by parts
- How it looks graphically



BCE Cost Function

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h(x^{(i)}, \theta) + (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta))]$$

The diagram illustrates the components of the BCE Cost Function. A large purple circle highlights the summation term $\sum_{i=1}^m$. Arrows point from each component to its corresponding part in the formula:

- An arrow labeled "Prediction" points to $h(x^{(i)}, \theta)$.
- An arrow labeled "Label" points to $y^{(i)}$.
- An arrow labeled "Features" points to $x^{(i)}$.
- An arrow labeled "Parameters" points to θ .

A red box at the bottom contains the text "Average loss of the whole batch".

BCE Cost Function

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h(x^{(i)}, \theta) + (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta))]$$

BCE Cost Function

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h(x^{(i)}, \theta) + (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta))]$$

$$\begin{array}{c|c} y^{(i)} & h(x^{(i)}, \theta) \\ \hline \end{array}$$

0	any	0
1	0.99	~0
1	~0	-inf

Relevant when
the label is 1

BCE Cost Function

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h(x^{(i)}, \theta) + (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta))]$$

$$\begin{array}{c|c} y^{(i)} & h(x^{(i)}, \theta) \\ \hline (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta)) & \end{array}$$

1	any	0
0	0.01	~0
0	~1	-inf

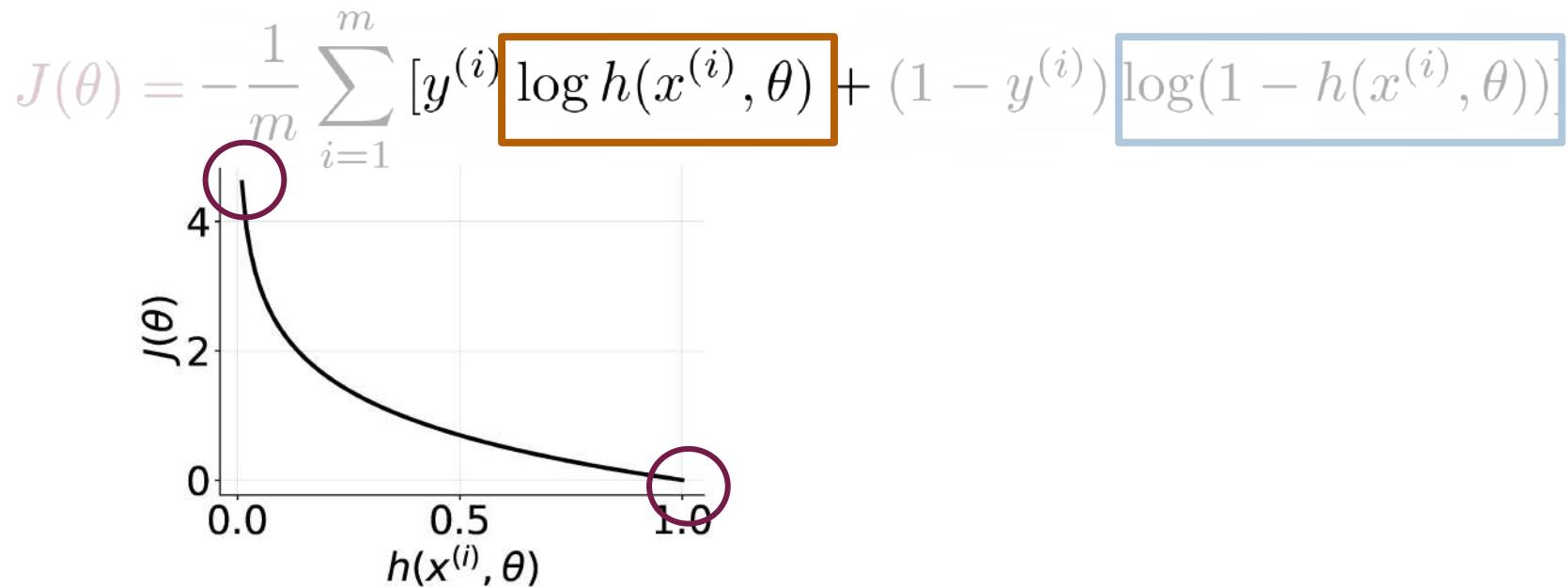
Relevant when
the label is 0

BCE Cost Function

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h(x^{(i)}, \theta) + (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta))]$$

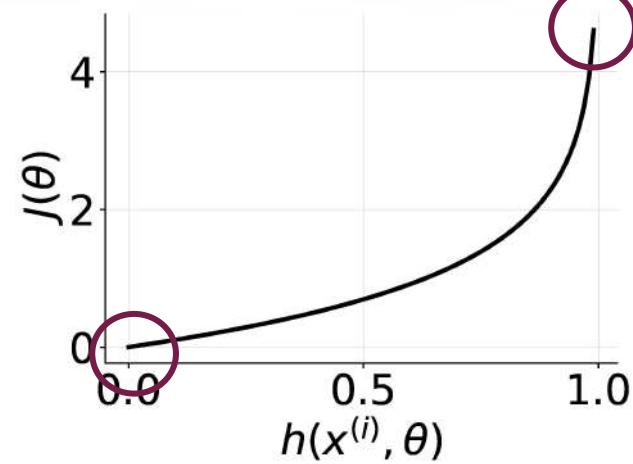
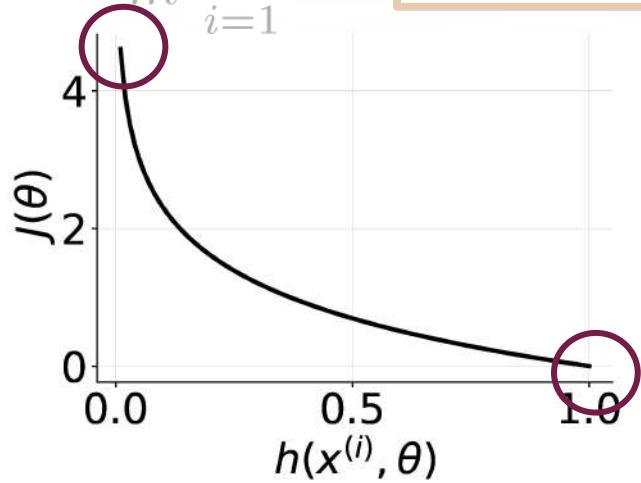
Ensures that the cost is always greater or equal to 0

BCE Cost Function



BCE Cost Function

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h(x^{(i)}, \theta) + (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta))]$$



Summary

- The BCE cost function has two parts (one relevant for each class)
- Close to zero when the label and the prediction are similar
- Approaches infinity when the label and the prediction are different



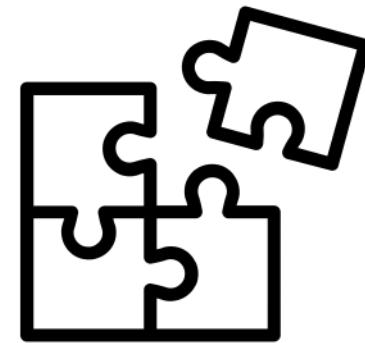


deeplearning.ai

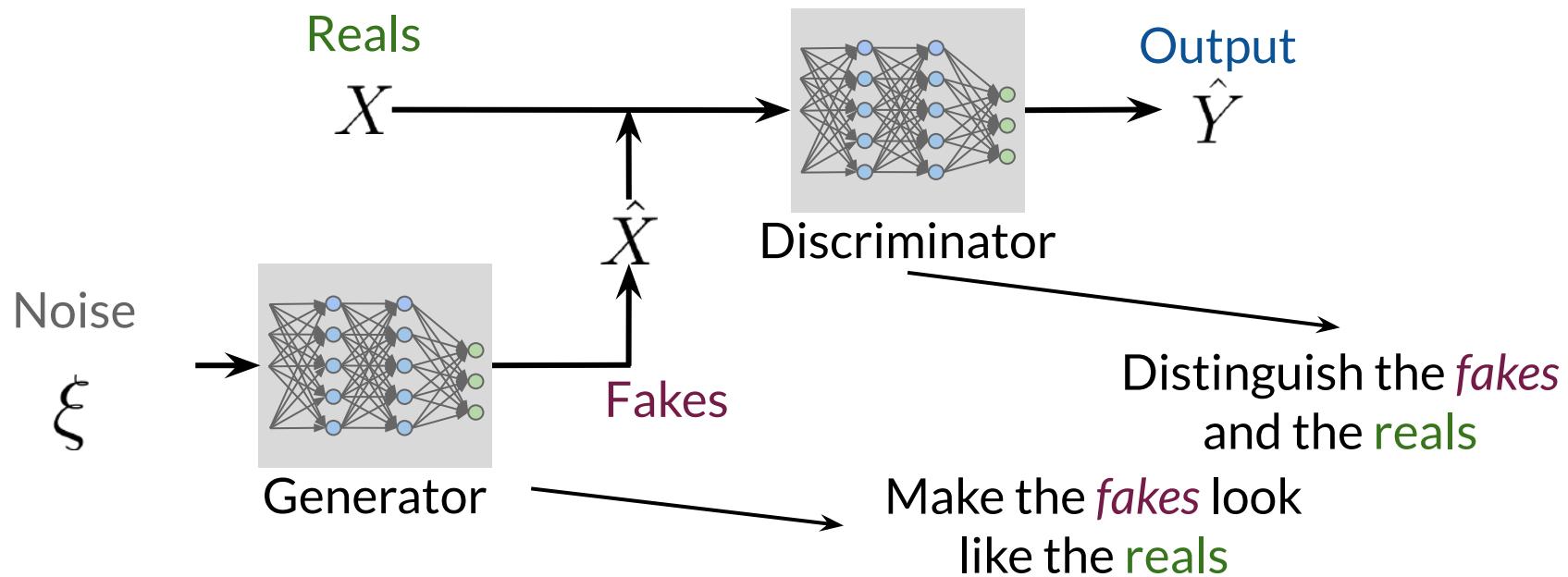
Putting It All Together

Outline

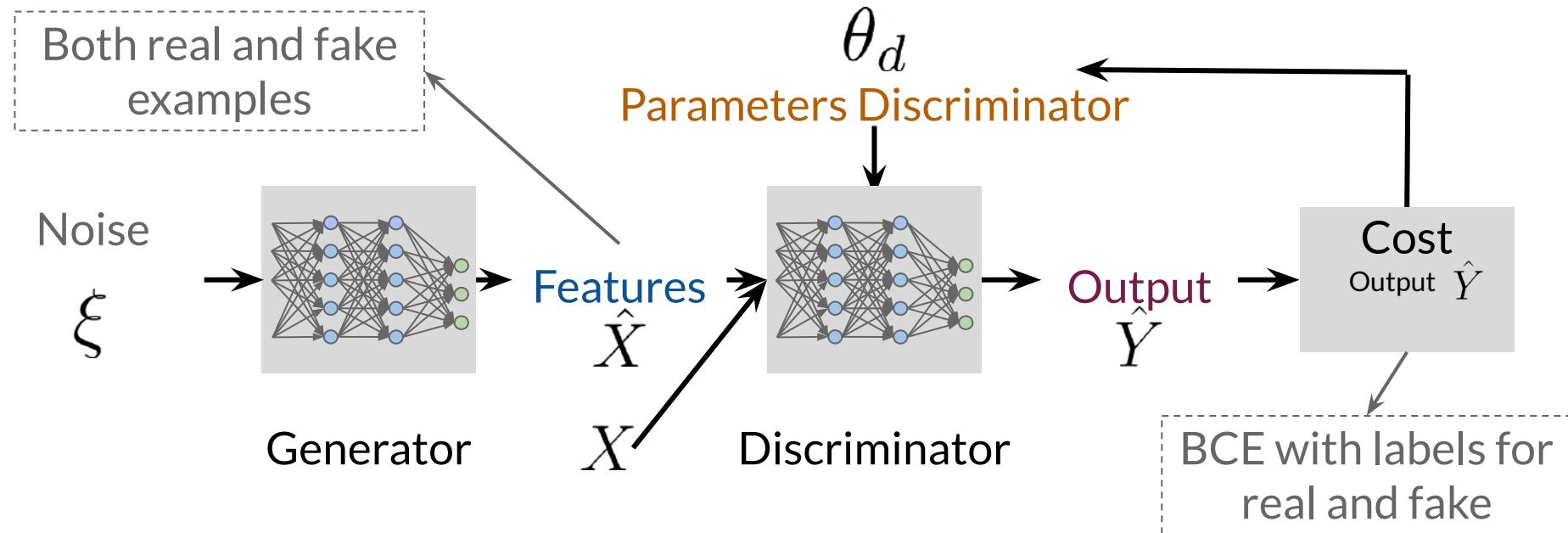
- How the whole architecture looks
- How to train GANs



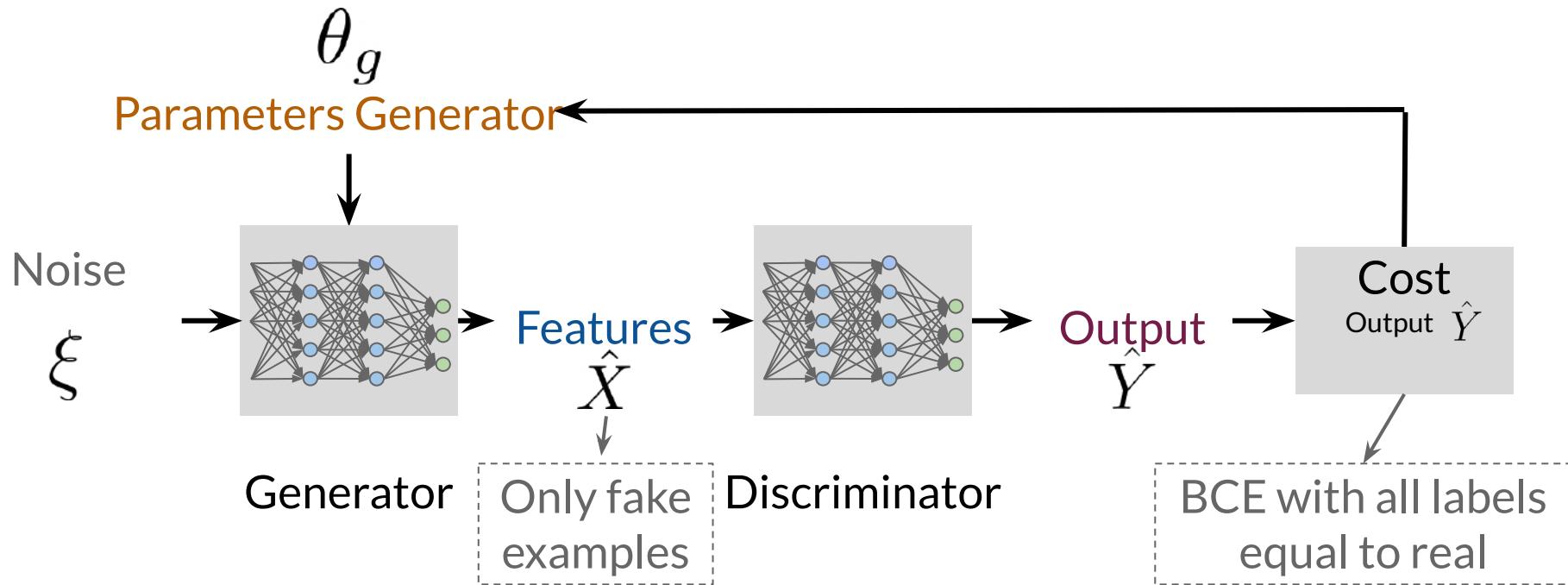
GANs Model



Training GANs: Discriminator



Training GANs: Generator

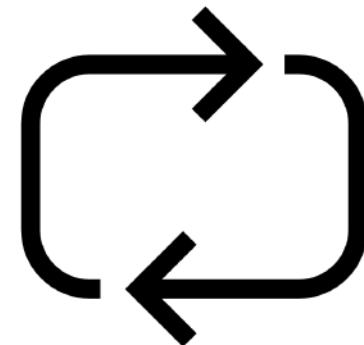


Training GANs



Summary

- GANs train in an alternating fashion
- The two models should always be at a similar “skill” level





deeplearning.ai

Intro to PyTorch (Optional)

Outline

- Comparison with TensorFlow
- Defining Models
- Training



PyTorch vs TensorFlow

PyTorch	TensorFlow
Imperative, computations on the go	Symbolic, first define and then compile
<pre>A, B = 1, 2 C = A + B print(C)</pre> 3 Dynamic Computational Graphs	<pre>C = A + B f = compile(C) print(f(A = 1, B = 2))</pre> 3 Static Computational Graphs
Tensorflow > 2.0 moves toward PyTorch by including Eager Execution	

PyTorch vs TensorFlow

PyTorch

TensorFlow

Currently very similar frameworks!

Tensorflow > 2.0 moves toward PyTorch by
including Eager Execution

Defining Models in PyTorch

```
import torch  
from torch import nn
```

→ Custom layers for DL

```
class LogisticRegression(nn.Module):  
    def __init__(self, in_):  
        super().__init__()  
        self.log_reg = nn.Sequential(  
            nn.Linear(in_, 1),  
            nn.Sigmoid()  
        )  
    def forward(self, x):  
        return self.log_reg(x)
```

Define the model as a class

Initialization method with parameters

Definition of the architecture

Forward computation of the model
with inputs x

Training Models In PyTorch

```
model = LogisticRegression(16)
```

Initialization of the model

```
criterion = nn.BCELoss()
```

Cost function

```
optimizer = torch.optim.SGD(model.parameters(), lr=0.01)
```

Optimizer

```
for t in range(n_epochs):
```

Training loop for number of epochs

```
    y_pred = model(x)
    loss = criterion(y_pred, y)
```

Forward propagation

```
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()
```

Optimization step

Summary

- PyTorch makes computations on the run
- Dynamic computational graphs in Pytorch
- Just another framework, and similar to Tensorflow!



Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>

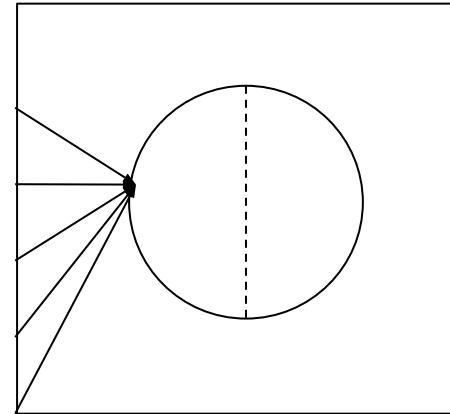


deeplearning.ai

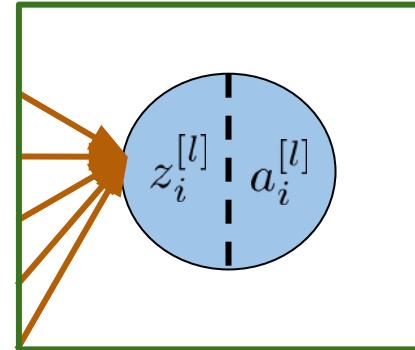
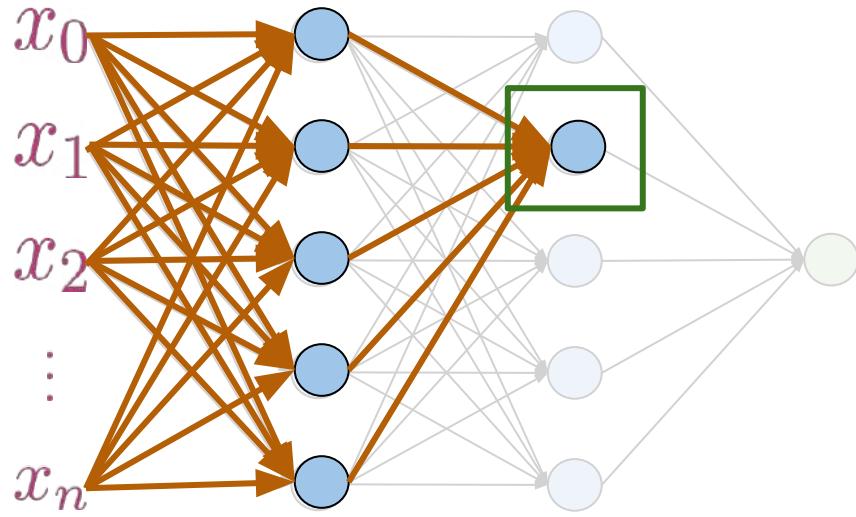
Activations (Basic Properties)

Outline

- What are activations
- Reasoning behind non-linear differential activations



Activations



$$z_i^{[l]} = \sum_{i=0} W_i^{[l]} a_i^{[l-1]}$$

$$a_i^{[l]} = g^{[l]}(z_i^{[l]})$$

Differentiable
non-linear
function

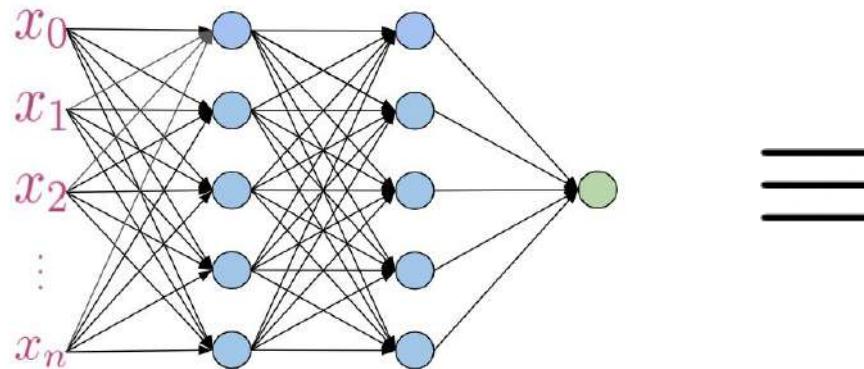
Activations

$$a_i^{[l]} = g^{[l]}(z_i^{[l]})$$

Differentiable
non-linear
function

1. Differentiable for backpropagation

2. Non-linear to compute complex features, **if not:**



$$WX + b$$

Linear
regression

Summary

- Activation functions are non-linear and differentiable
- Differentiable for backpropagation
- Non-linear to approximate complex functions

$f(x)$



deeplearning.ai

Common Activation Functions

Outline

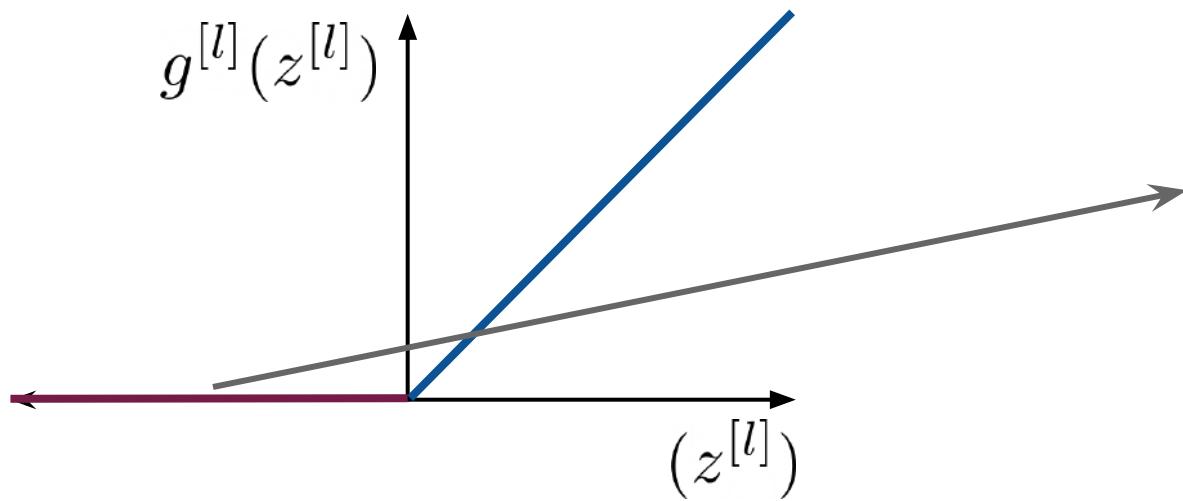
- Common activations and their structure
 - ReLU
 - Leaky ReLU
 - Sigmoid
 - Tanh



Activations: ReLU

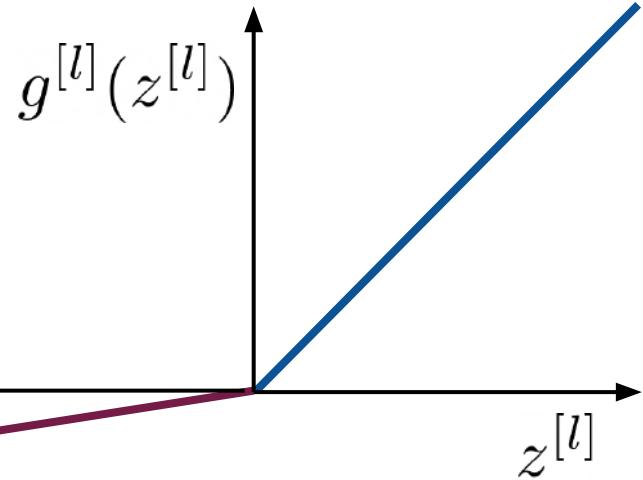
ReLU = Rectified Linear Unit

$$g^{[l]}(z^{[l]}) = \max(0, z^{[l]})$$



Dying ReLU
problem

Activations: Leaky ReLU

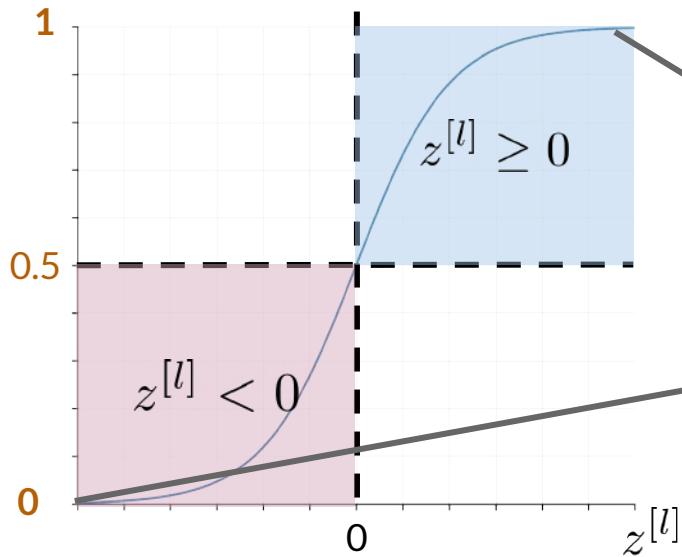


$$g^{[l]}(z^{[l]}) = \max(az^{[l]}, z^{[l]})$$

Solves the dying
ReLU problem

Activations: Sigmoid

$$g^{[l]}(z^{[l]}) = \frac{1}{1 + e^{-z^{[l]}}}$$

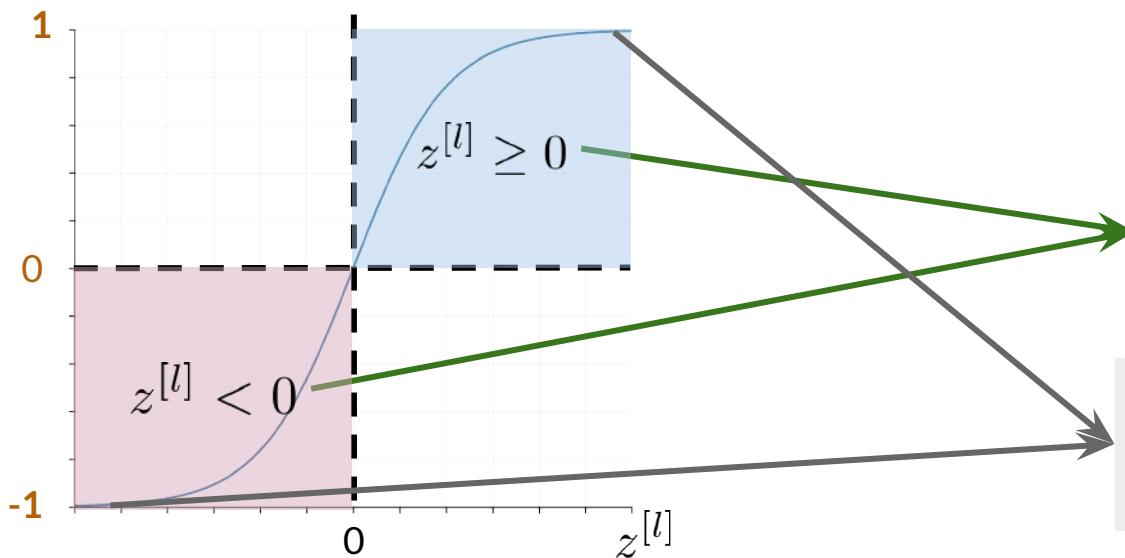


Values between 0
and 1

Vanishing gradient
and saturation
problems

Activations: Tanh

$$g^{[l]}(z^{[l]}) = \tanh(z^{[l]})$$



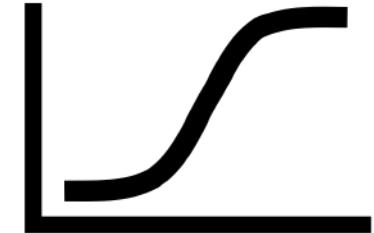
Values between -1 and 1

Keeps the sign of the input

Same issues as Sigmoid

Summary

- ReLU activations suffer from dying ReLU
- Leaky ReLU solve the dying ReLU problem
- Sigmoid and Tanh have vanishing gradient and saturation problems





deeplearning.ai

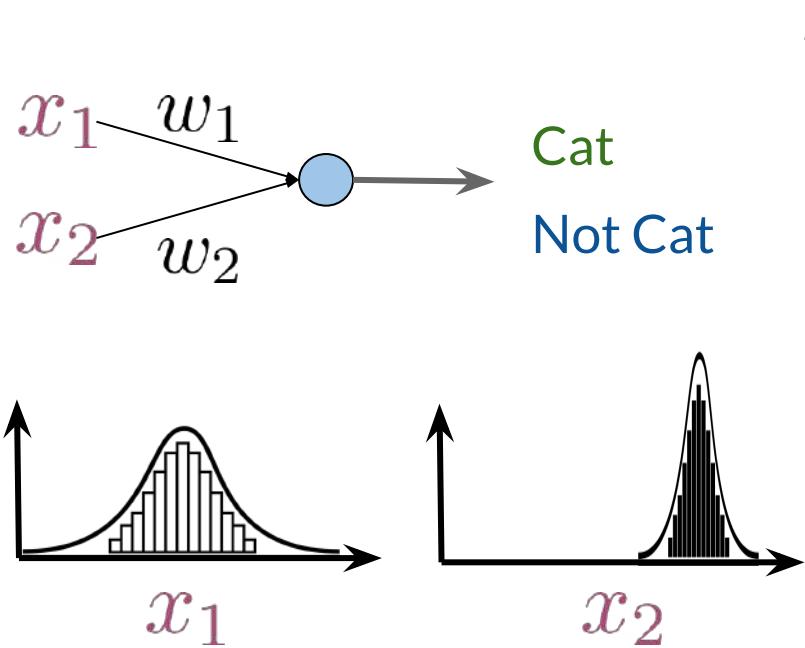
Batch Normalization (Explained)

Outline

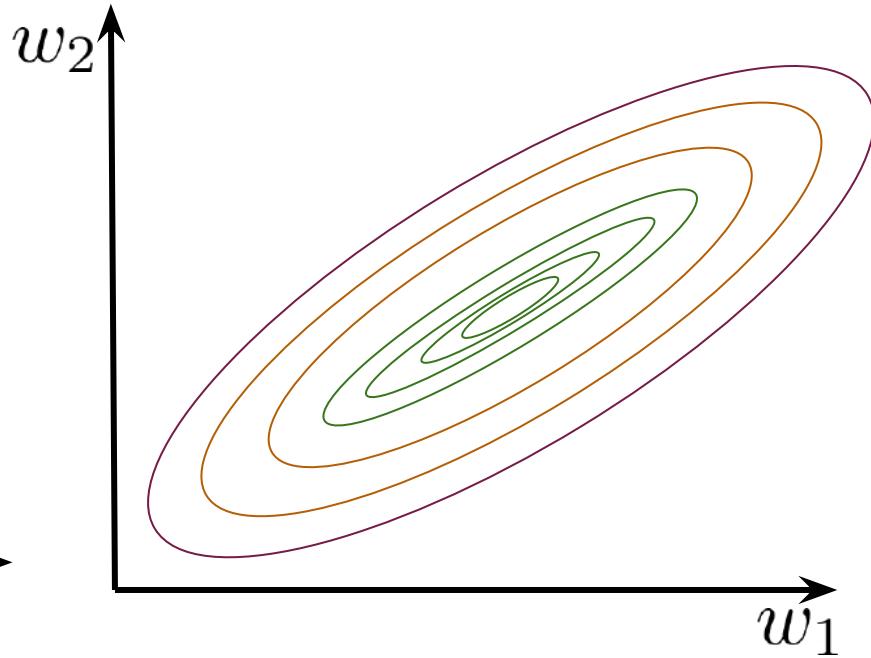
- How normalization helps models
- Internal covariate shift
- Batch normalization



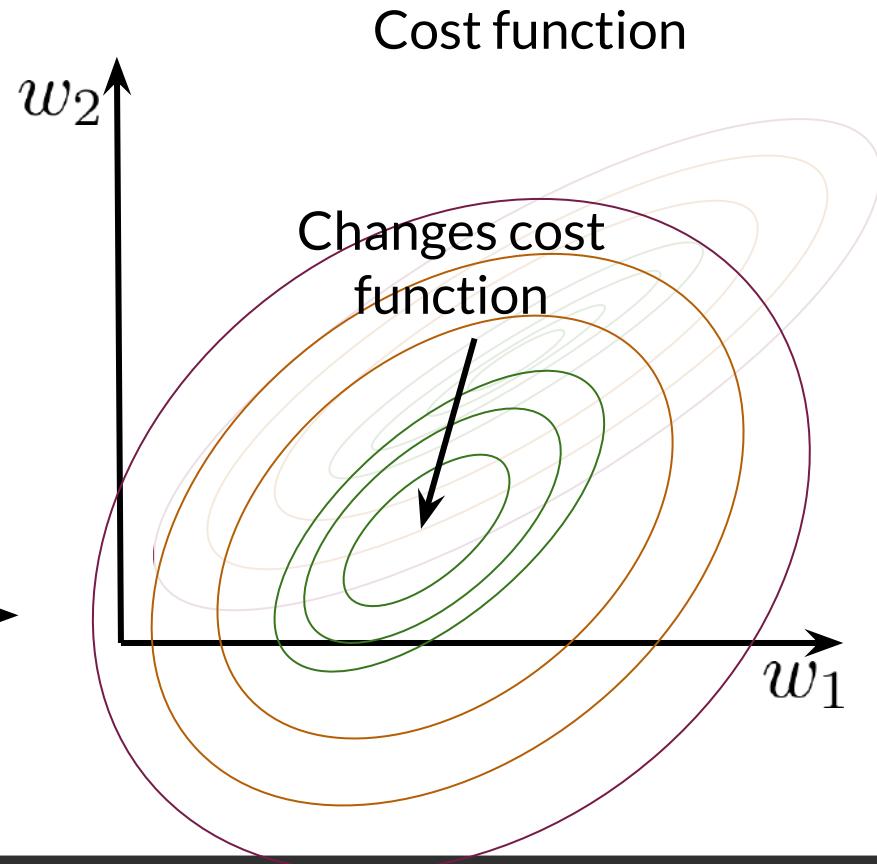
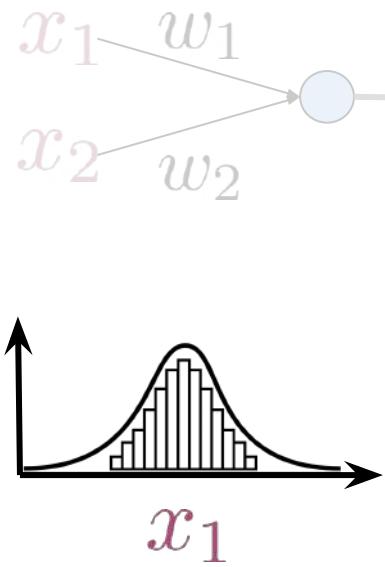
Different Distributions



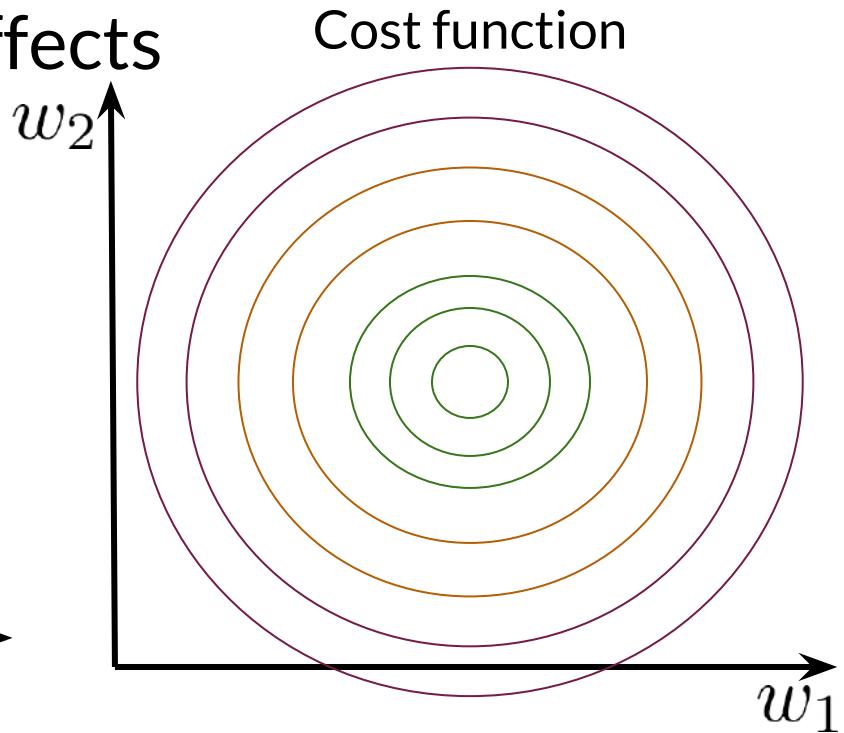
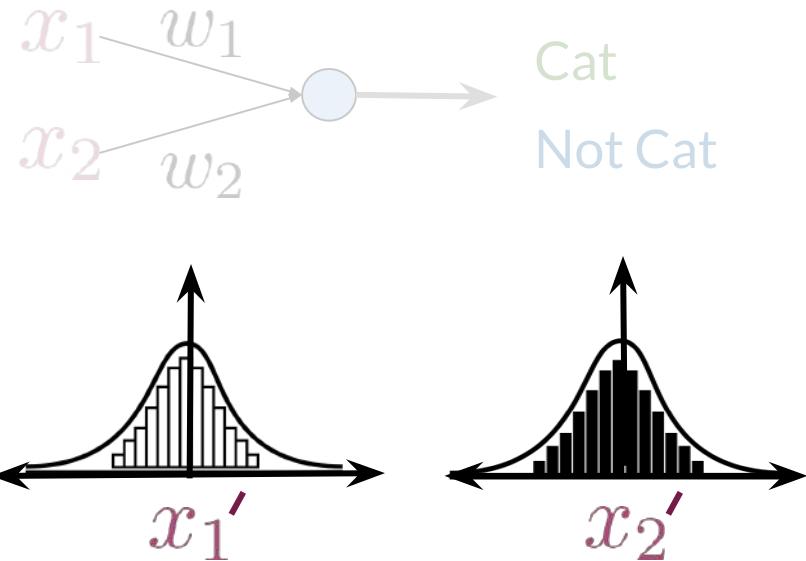
Cost function



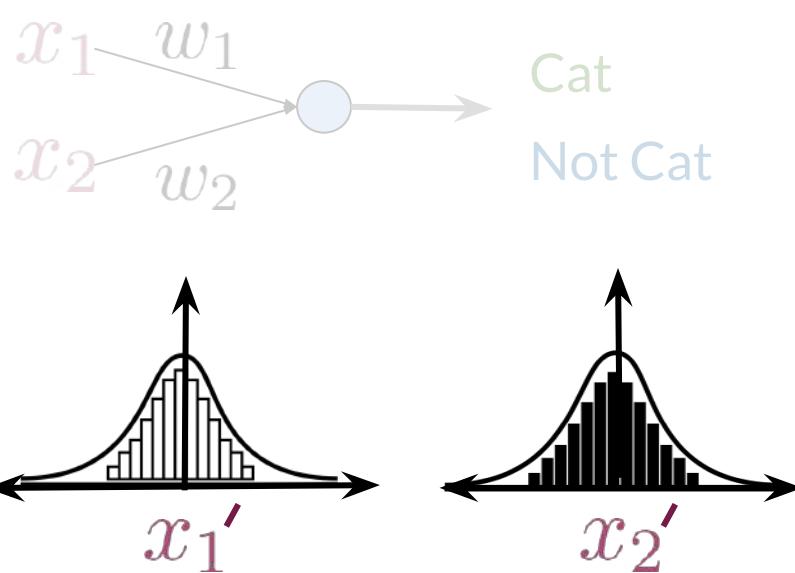
Covariate Shift



Normalization and Its Effects



Normalization and Its Effects



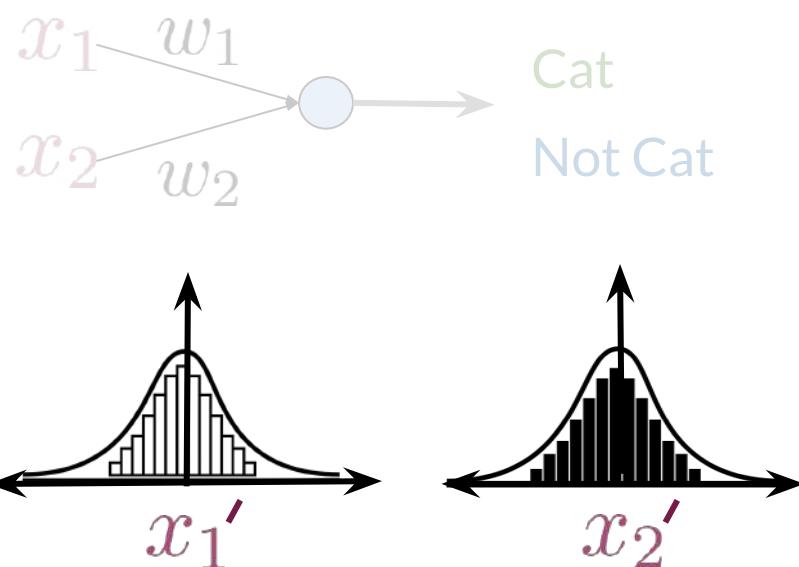
$$x_1' \quad x_2'$$

Around mean at 0
and std. at 1

Training data uses
batch stats

Test data uses
training stats

Normalization and Its Effects



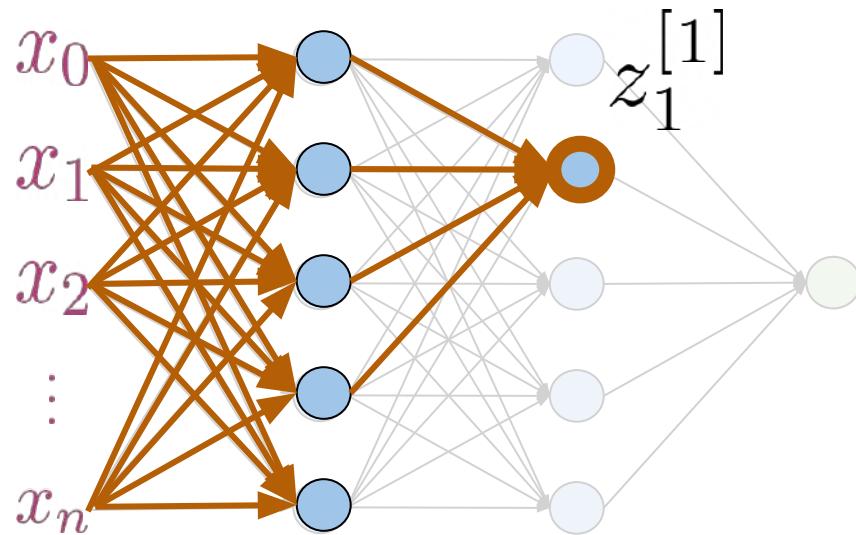
x_1' x_2'

Around mean at 0
and std. at 1



Reduction of
covariate shift

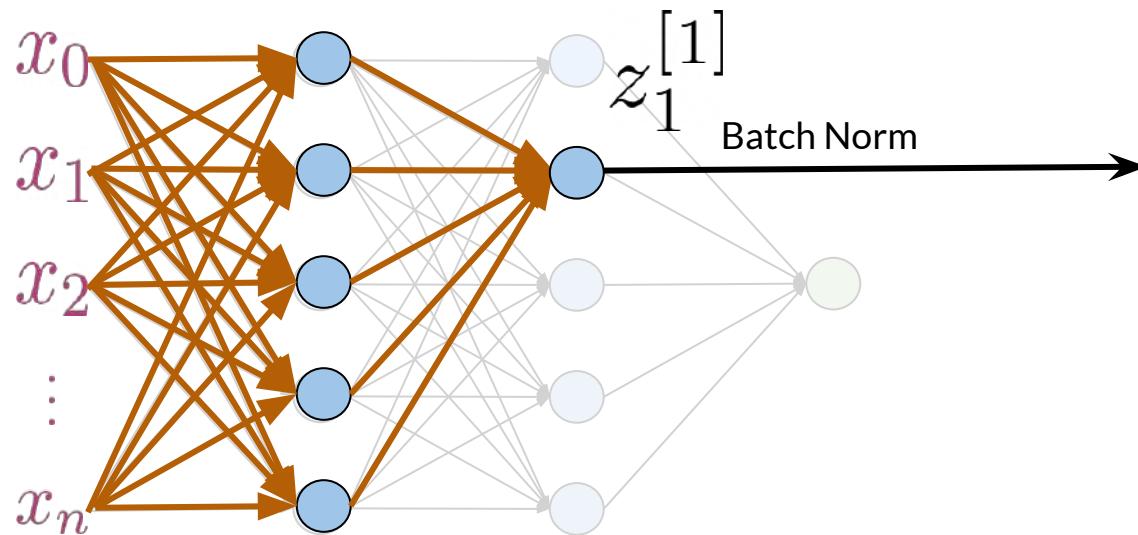
Internal Covariate Shift



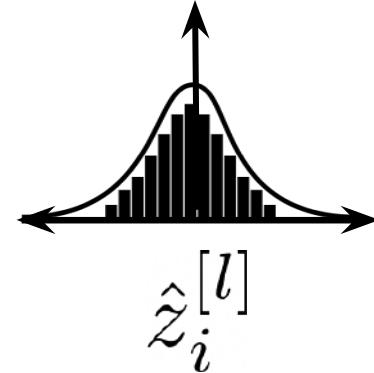
Changes in
weights

↓
Changes in
activation
distribution

Batch Normalization

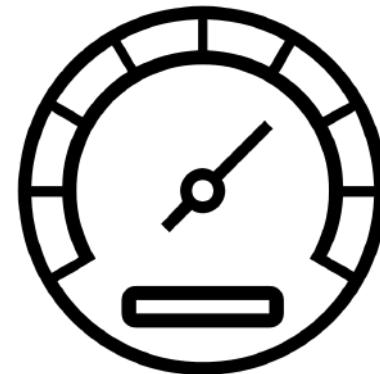


Normalizes the
input for each
neuron



Summary

- Batch normalization smooths the cost function
- Batch normalization reduces the internal covariate shift
- Batch normalization speeds up learning!



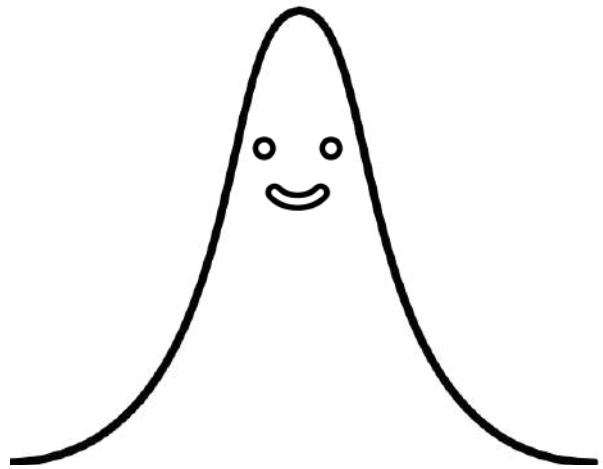


deeplearning.ai

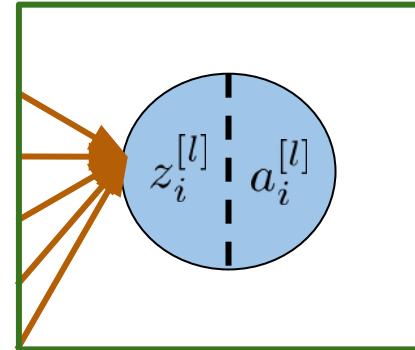
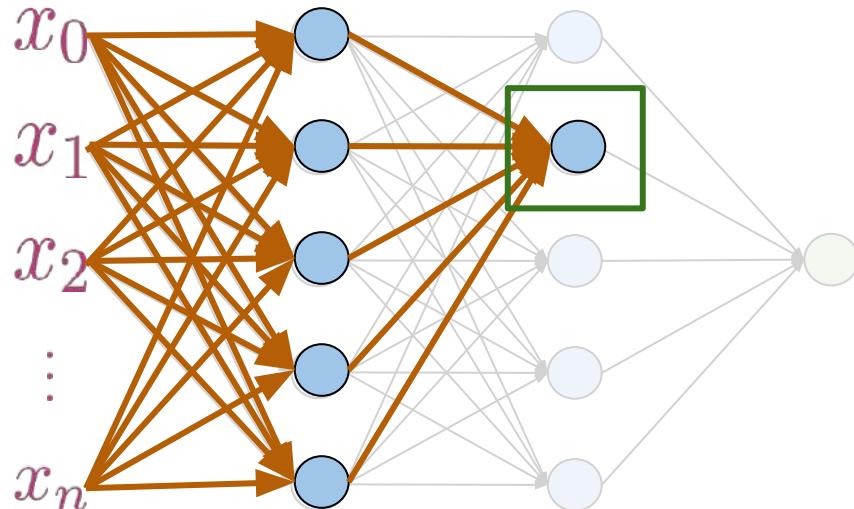
Batch Normalization (Procedure)

Outline

- Batch norm for training
- Batch norm for testing



Batch Normalization: Training



$$z_i^{[l]} = \sum_{i=0} W_i^{[l]} a_i^{[l-1]} \longrightarrow$$

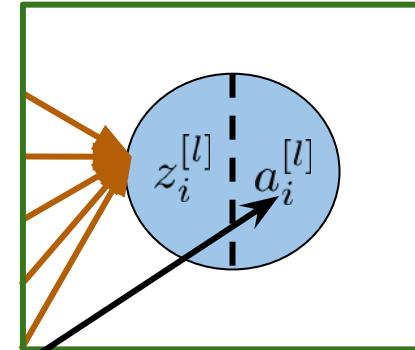
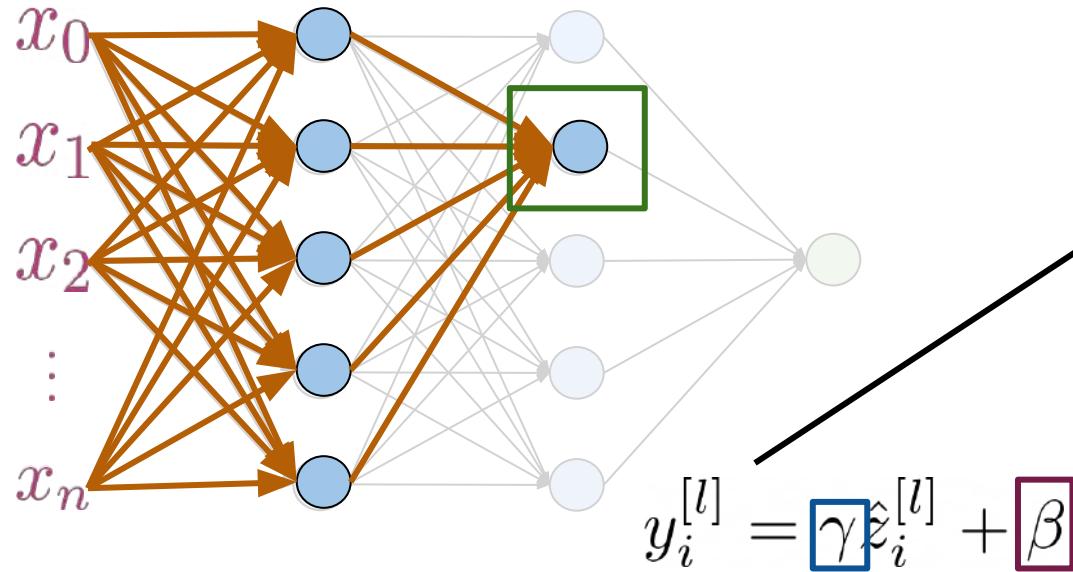
For every
example in
the batch

$$\hat{z}_i^{[l]} = \frac{z_i^{[l]} - \mu_{z_i^{[l]}}}{\sqrt{\sigma_{z_i^{[l]}}^2 + \epsilon}}$$

Batch mean of $z_i^{[l]}$

Batch std of $z_i^{[l]}$

Batch Normalization: Training



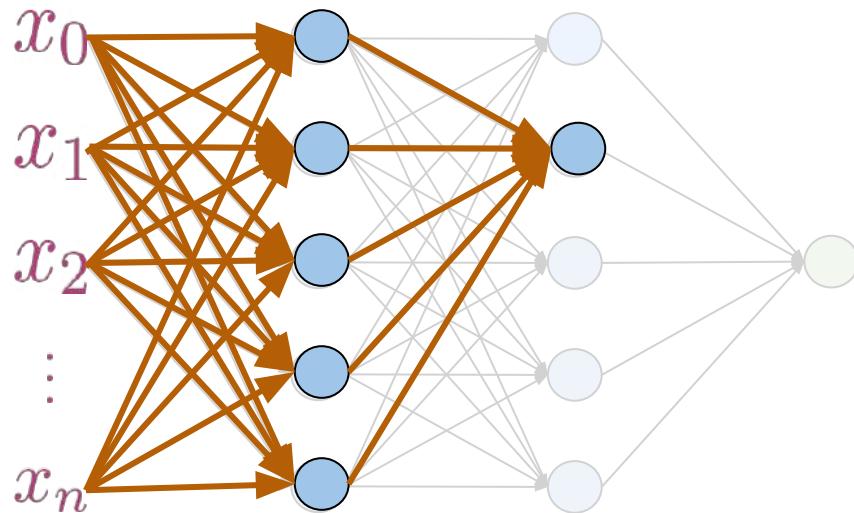
$$\hat{z}_i^{[l]} = \frac{z_i^{[l]} - \mu_{z_i^{[l]}}}{\sqrt{\sigma_{z_i^{[l]}}^2 + \epsilon}}$$

Shift factor

Scale Factor

Learnable
parameters to get
the optimal dist.

Batch Normalization: Test



$$\hat{z}_i^{[l]} = \frac{z_i^{[l]} - \mathbf{E}(z_i^{[l]})}{\sqrt{\text{Var}(z_i^{[l]}) + \epsilon}}$$

Running **mean** and running
std from training

Frameworks like
Tensorflow and Pytorch
keep track of them

Summary

- Batch norm introduces learnable shift and scale factors
- During test, the running statistics from training are used
- Frameworks take care of the whole process



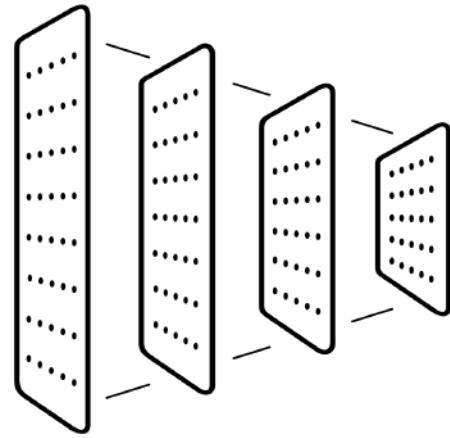


deeplearning.ai

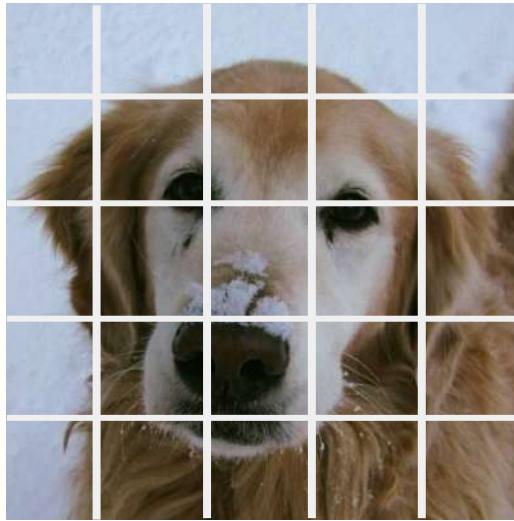
Review of Convolutions

Outline

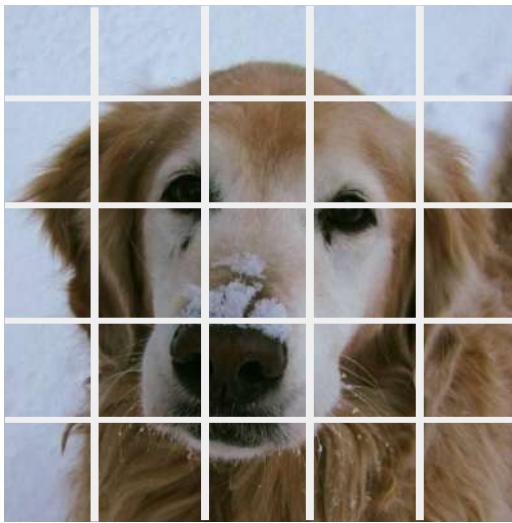
- What convolutions are
- How they work



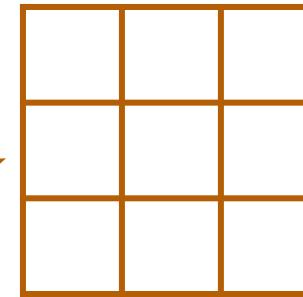
What is a convolution?



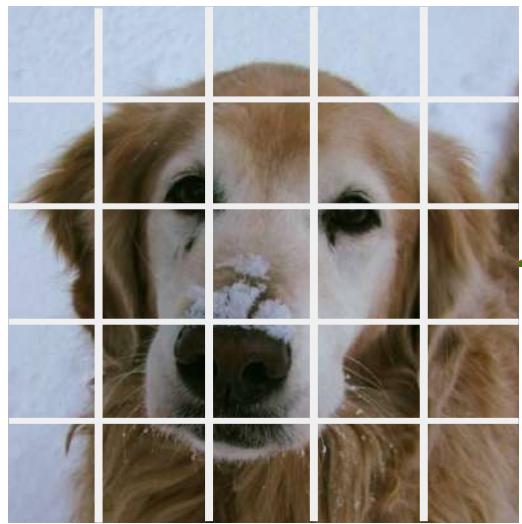
What is a convolution?



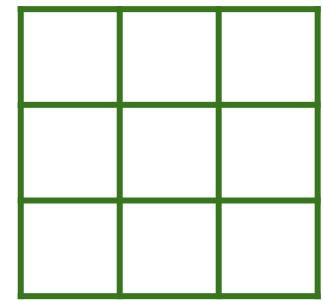
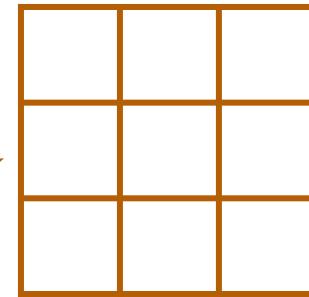
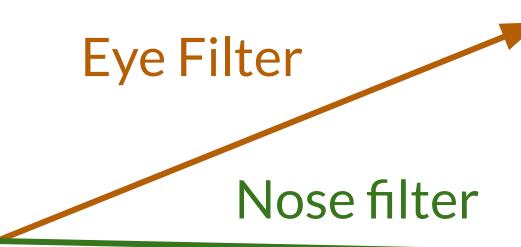
Eye Filter



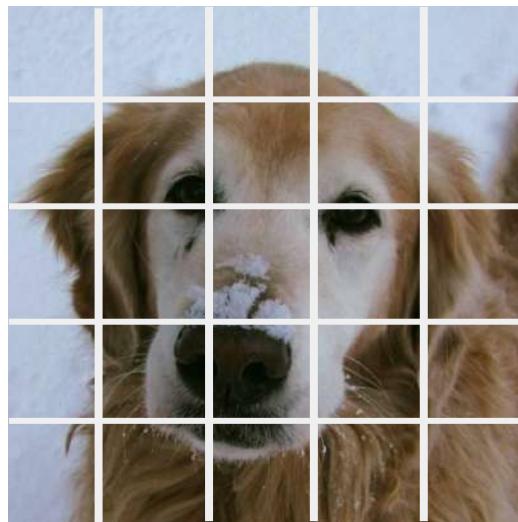
What is a convolution?



Eye Filter



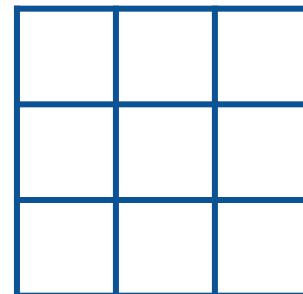
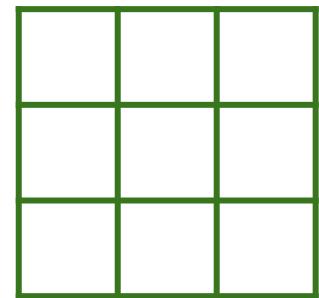
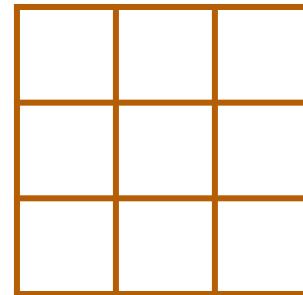
What is a convolution?



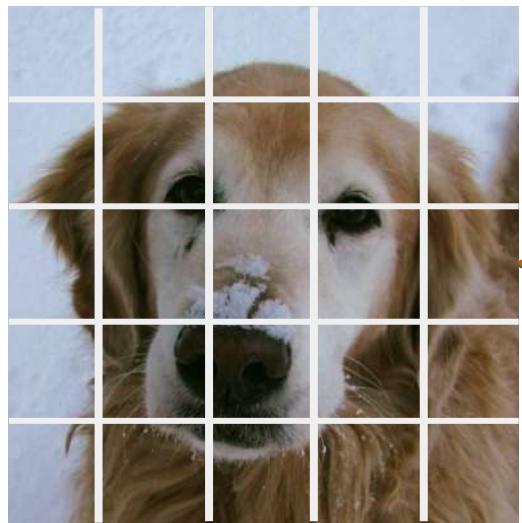
Eye Filter

Nose filter

Ear filter



What is a convolution?



Eye Filter

Nose filter

Ear filter

5	3	1
10	23	1
1	42	4

2	21	5
23	45	12
3	32	4

32	2	24
25	12	66
3	45	2

What is a convolution?

50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0

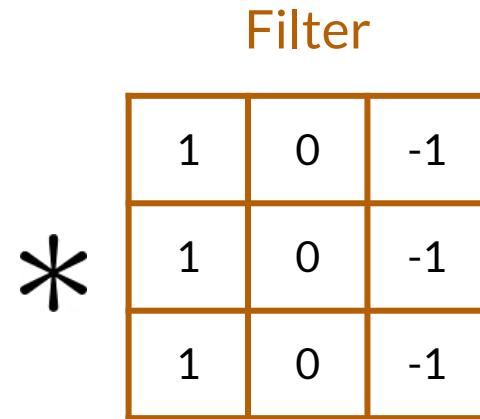
Grayscale image

0 (black) to 255 (white)

What is a convolution?

50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0

Grayscale image

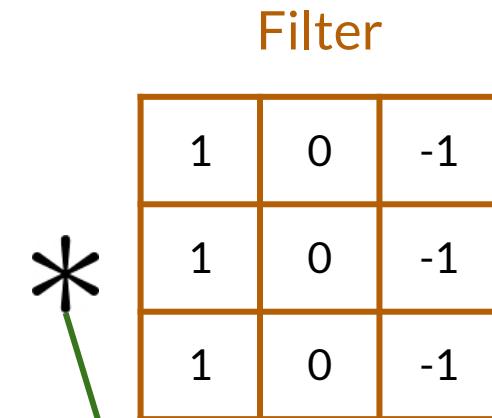


0 (black) to 255 (white)

What is a convolution?

50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0

Grayscale image



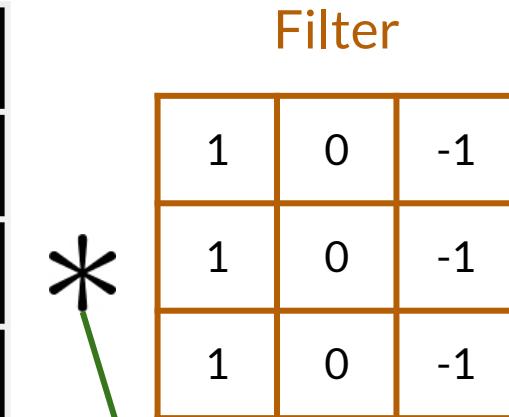
Convolution

0 (black) to 255 (white)

What is a convolution?

50*1	50*0	0*-1	0	0
50*1	50*0	0*-1	0	0
50*1	50*0	0*-1	0	0
50	50	0	0	0
50	50	0	0	0

Grayscale image



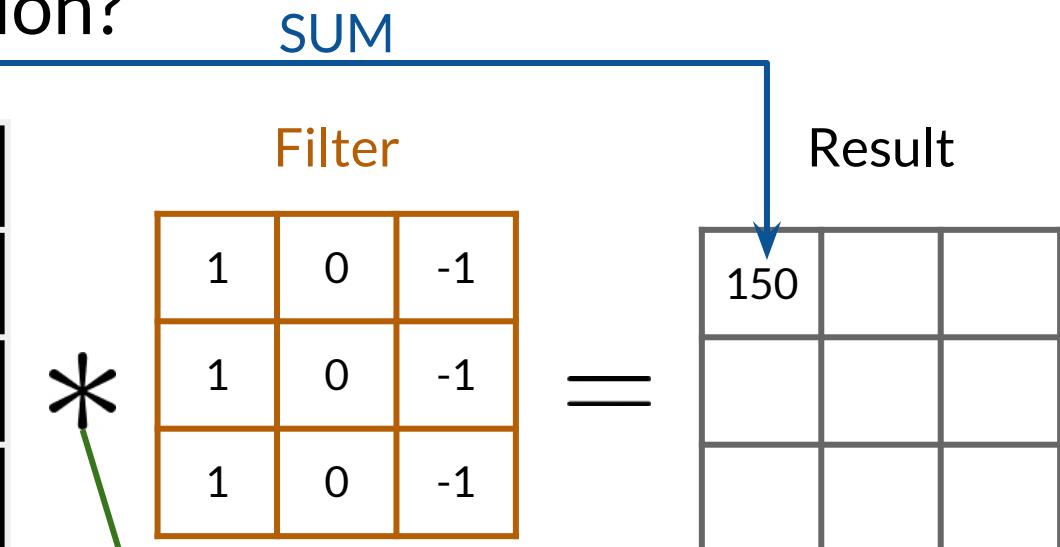
Convolution

0 (black) to 255 (white)

What is a convolution?

50*1	50*0	0*-1	0	0
50*1	50*0	0*-1	0	0
50*1	50*0	0*-1	0	0
50	50	0	0	0
50	50	0	0	0

Grayscale image



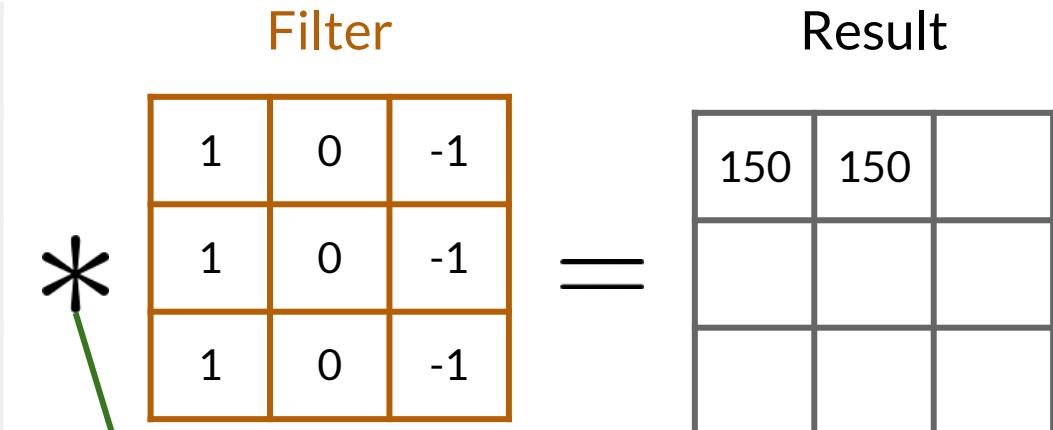
Convolution

0 (black) to 255 (white)

What is a convolution?

50	50*1	0*0	0*-1	0
50	50*1	0*0	0*-1	0
50	50*1	0*0	0*-1	0
50	50	0	0	0
50	50	0	0	0

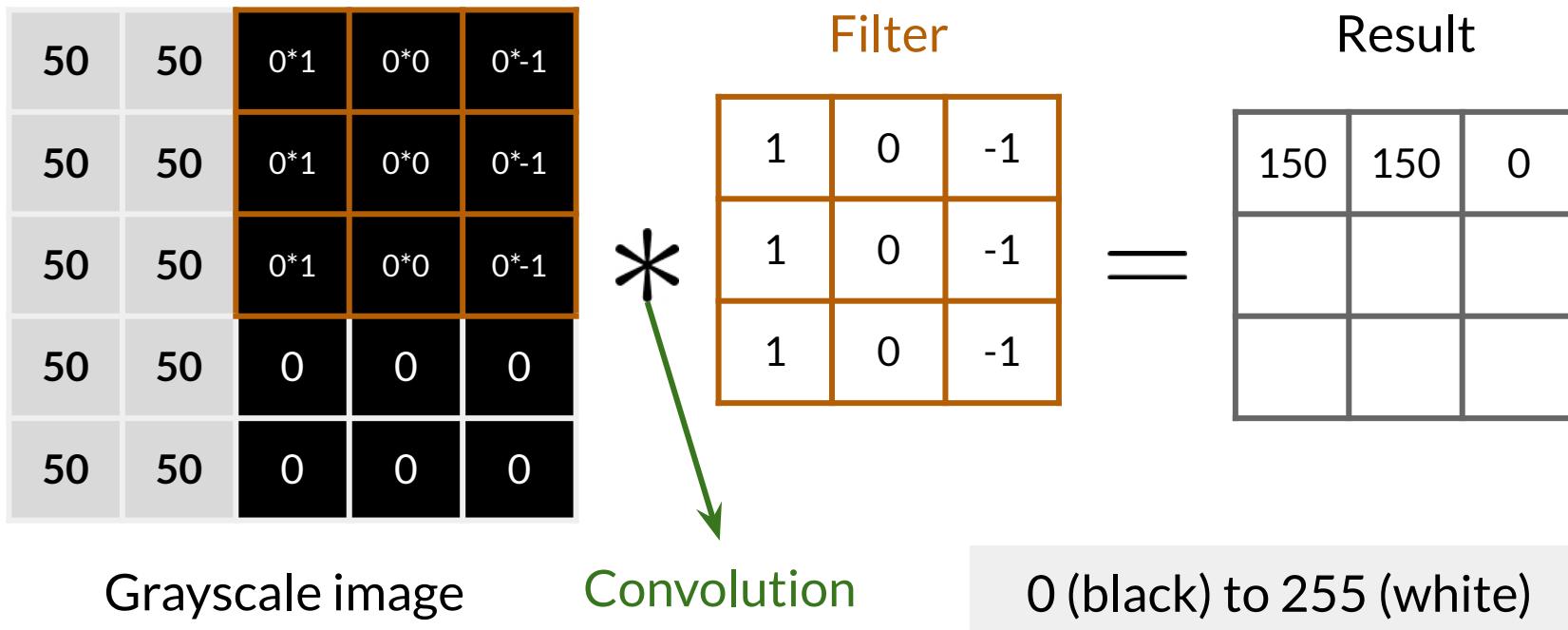
Grayscale image



Convolution

0 (black) to 255 (white)

What is a convolution?



What is a convolution?

50	50	0	0	0
50*1	50*0	0*-1	0	0
50*1	50*0	0*-1	0	0
50*1	50*0	0*-1	0	0
50	50	0	0	0

Grayscale image

Filter

1	0	-1
1	0	-1
1	0	-1



Convolution

Result

150	150	0
150		

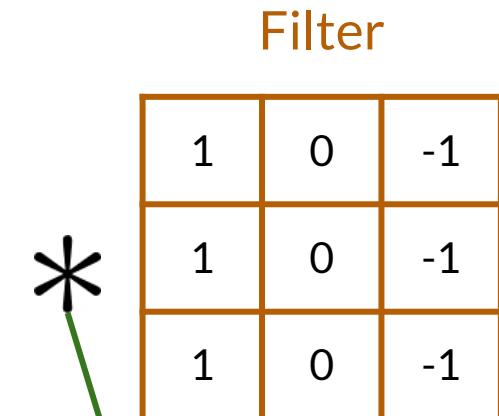
=

0 (black) to 255 (white)

What is a convolution?

50	50	0	0	0
50	50*1	0*0	0*-1	0
50	50*1	0*0	0*-1	0
50	50*1	0*0	0*-1	0
50	50	0	0	0

Grayscale image



Convolution

Result

150	150	0
150	150	

0 (black) to 255 (white)

What is a convolution?

50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0

Grayscale image

Filter

1	0	-1
1	0	-1
1	0	-1



Convolution

=

150	150	0
150	150	0
150	150	0

0 (black) to 255 (white)

Summary

- Convolutions are useful layers for processing images
- They scan the image to detect useful features
- Just element-wise products and sums!



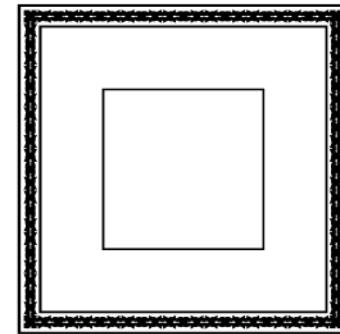


deeplearning.ai

Padding and Stride

Outline

- Padding and stride
- The intuition behind padding



Stride

50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0
50	50	0	0	0

*

Filter

1	0	-1
1	0	-1
1	0	-1

Grayscale image

Stride

50*1	50*0	0*-1	0	0
50*1	50*0	0*-1	0	0
50*1	50*0	0*-1	0	0
50	50	0	0	0
50	50	0	0	0

Grayscale image

Filter

*

Result

=

The diagram illustrates a convolution operation. On the left, a 5x5 grayscale image is shown with its dimensions (50x1, 50x0, etc.) and values (50, 50). A 3x3 filter is applied to the image, indicated by an asterisk (*). The result of the convolution is 150, shown in the first cell of a 3x3 result matrix.

1	0	-1
1	0	-1
1	0	-1

150		

Stride

→ 1 Pixel to the right

50	50*1	0*0	0*-1	0
50	50*1	0*0	0*-1	0
50	50*1	0*0	0*-1	0
50	50	0	0	0
50	50	0	0	0

Grayscale image

Filter

1	0	-1
1	0	-1
1	0	-1

Result

150	150	



Stride

→ 1 Pixel to the right

50	50	0*1	0*0	0*-1
50	50	0*1	0*0	0*-1
50	50	0*1	0*0	0*-1
50	50	0	0	0
50	50	0	0	0

Grayscale image

Filter

1	0	-1
1	0	-1
1	0	-1

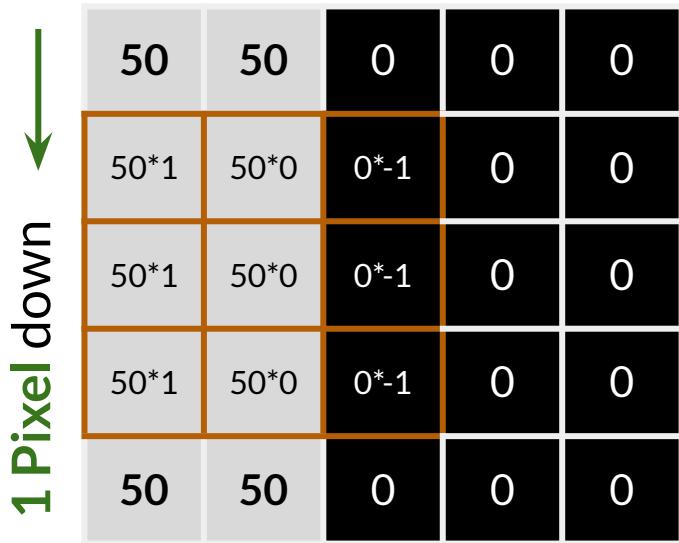
Result

150	150	0



Stride

→ 1 Pixel to the right



*

Filter

1	0	-1
1	0	-1
1	0	-1

=

Result

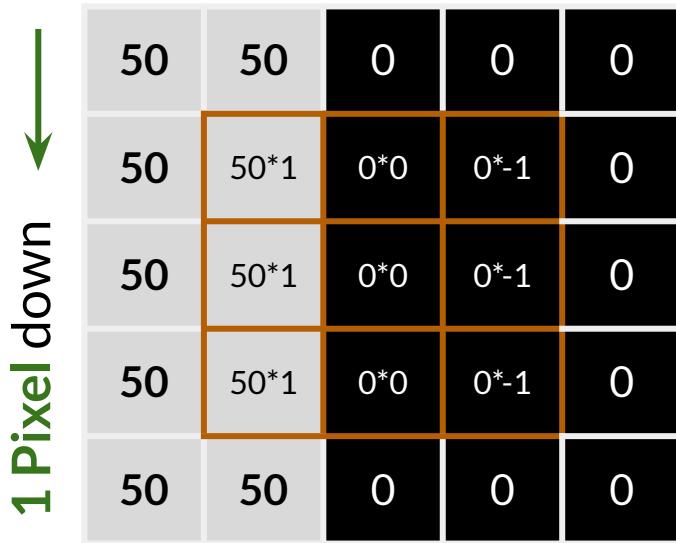
150	150	0
150		

Grayscale image



Stride

→ 1 Pixel to the right



*

Filter

1	0	-1
1	0	-1
1	0	-1

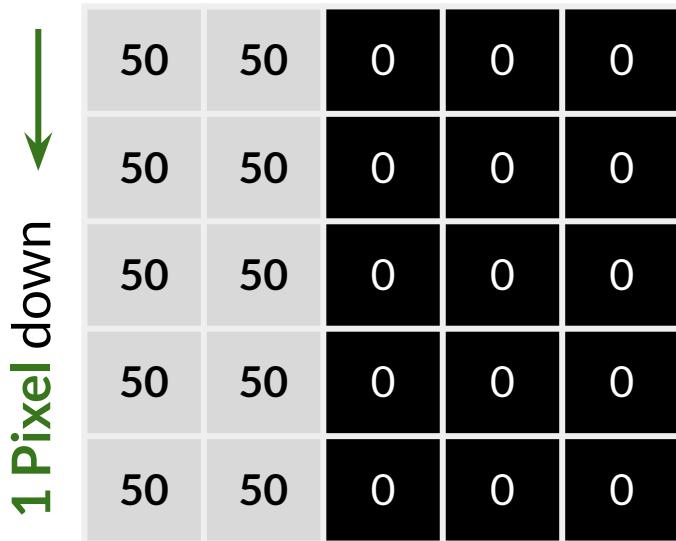
Result

150	150	0
150	150	

Grayscale image

Stride

→ 1 Pixel to the right



Grayscale image

* =

Filter

1	0	-1
1	0	-1
1	0	-1

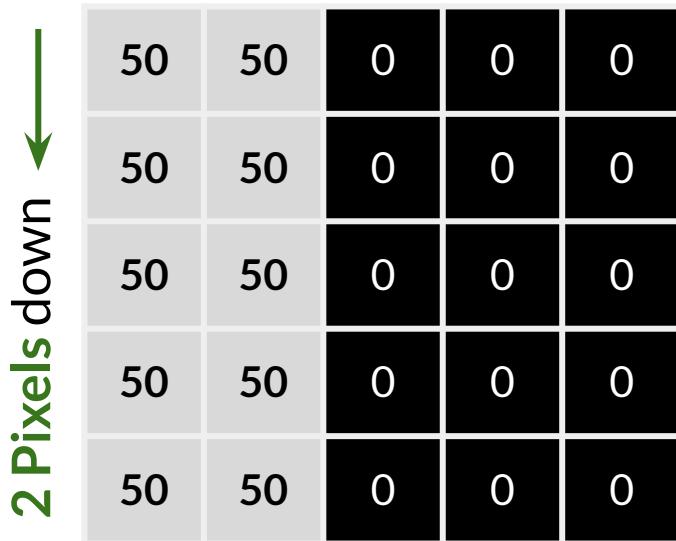
Result

150	150	0
150	150	0
150	150	0

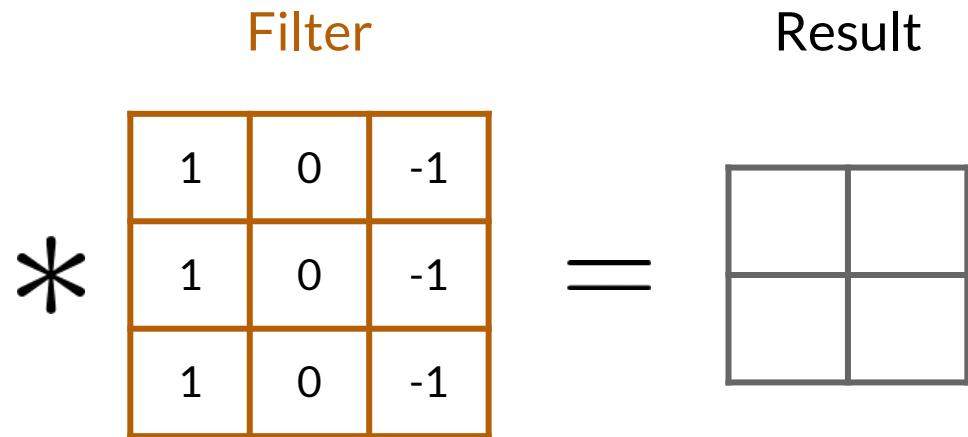
Stride = 1

Stride

→ 2 Pixels to the right



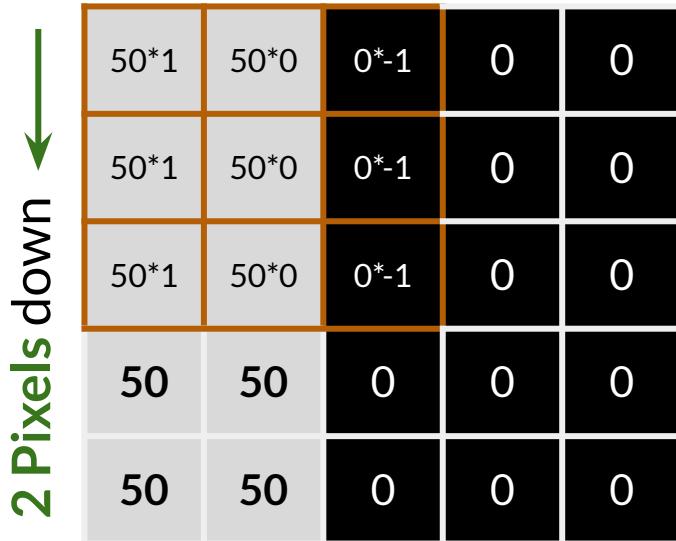
Grayscale image



Stride = 2

Stride

→ 2 Pixels to the right



Grayscale image

Filter

*

A diagram showing a convolution operation. On the left is a 5x5 input image with values: Row 1: 50*1, 50*0, 0*-1, 0, 0. Row 2: 50*1, 50*0, 0*-1, 0, 0. Row 3: 50*1, 50*0, 0*-1, 0, 0. Row 4: 50, 50, 0, 0, 0. Row 5: 50, 50, 0, 0, 0. A 3x3 filter with values [1, 0, -1; 1, 0, -1; 1, 0, -1] is applied to it. The result is a single value 150 in a 2x2 output unit.

1	0	-1
1	0	-1
1	0	-1

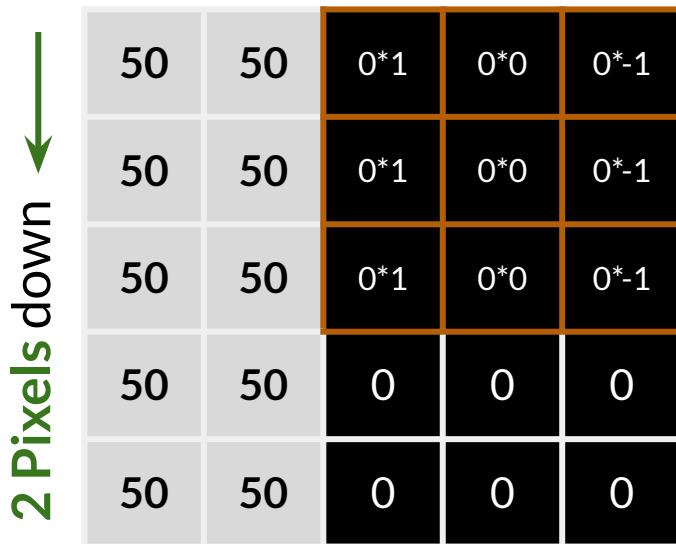
=

150	

Stride = 2

Stride

→ 2 Pixels to the right



Grayscale image

Filter

*

1	0	-1
1	0	-1
1	0	-1

=

150	0

Stride = 2

Stride

→ 2 Pixels to the right



Grayscale image

Filter

*

The convolution operation is shown as $\text{Grayscale image} * \text{Filter} = \text{Result}$. The result is a 2x2 matrix with values 150 and 0. The filter is a 3x3 matrix with values 1, 0, -1 in each row.

1	0	-1
1	0	-1
1	0	-1

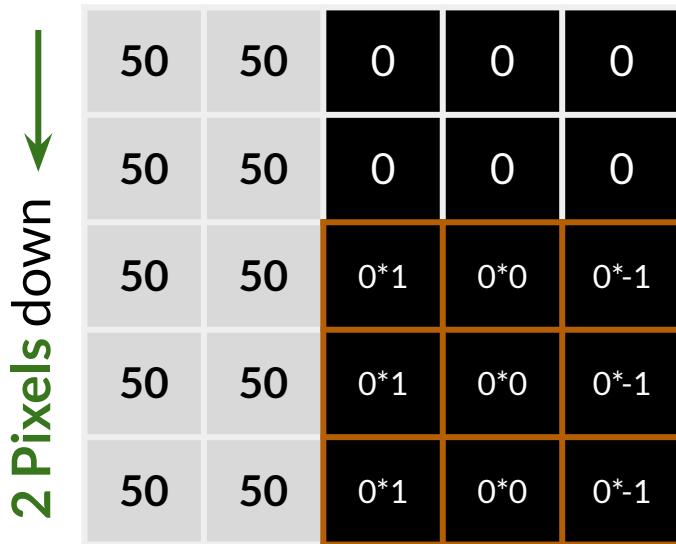
=

150	0
150	

Stride = 2

Stride

→ 2 Pixels to the right



Grayscale image

*

Filter

1	0	-1
1	0	-1
1	0	-1

=

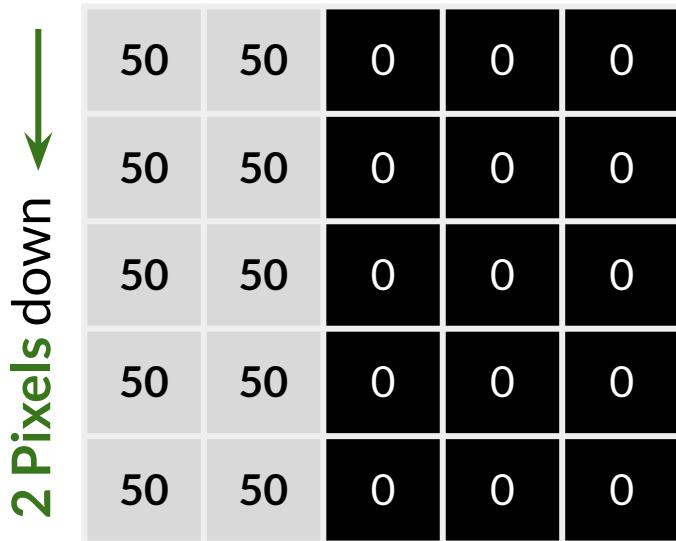
Result

150	0
150	0

Stride = 2

Stride

→ 2 Pixels to the right



Grayscale image

Filter

*

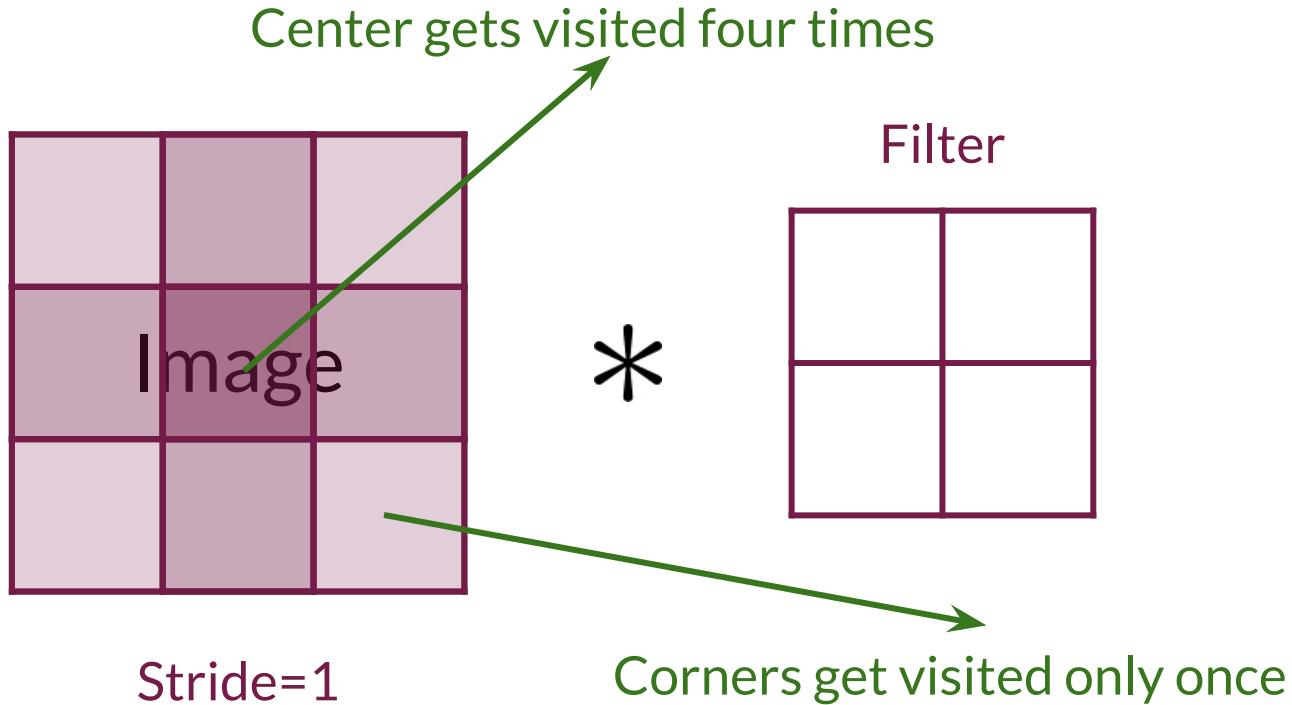
1	0	-1
1	0	-1
1	0	-1

=

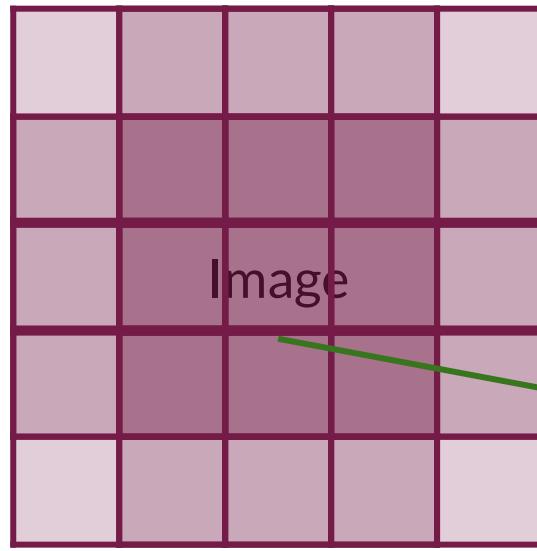
150	0
150	0

Stride = 2

Padding

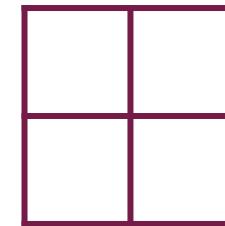


Padding



Filter

*



Every pixel within the image gets visited the same number of times

Summary

- Stride determines how the filter scans the image
- Padding is like a frame on the image
- Padding gives similar importance to the edges and the center



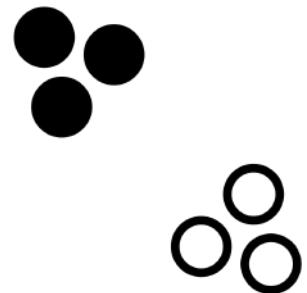


deeplearning.ai

Pooling and Upsampling

Outline

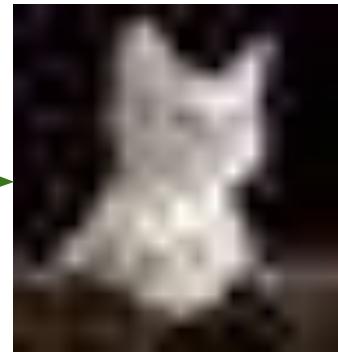
- Pooling
- Upsampling and its relation to pooling



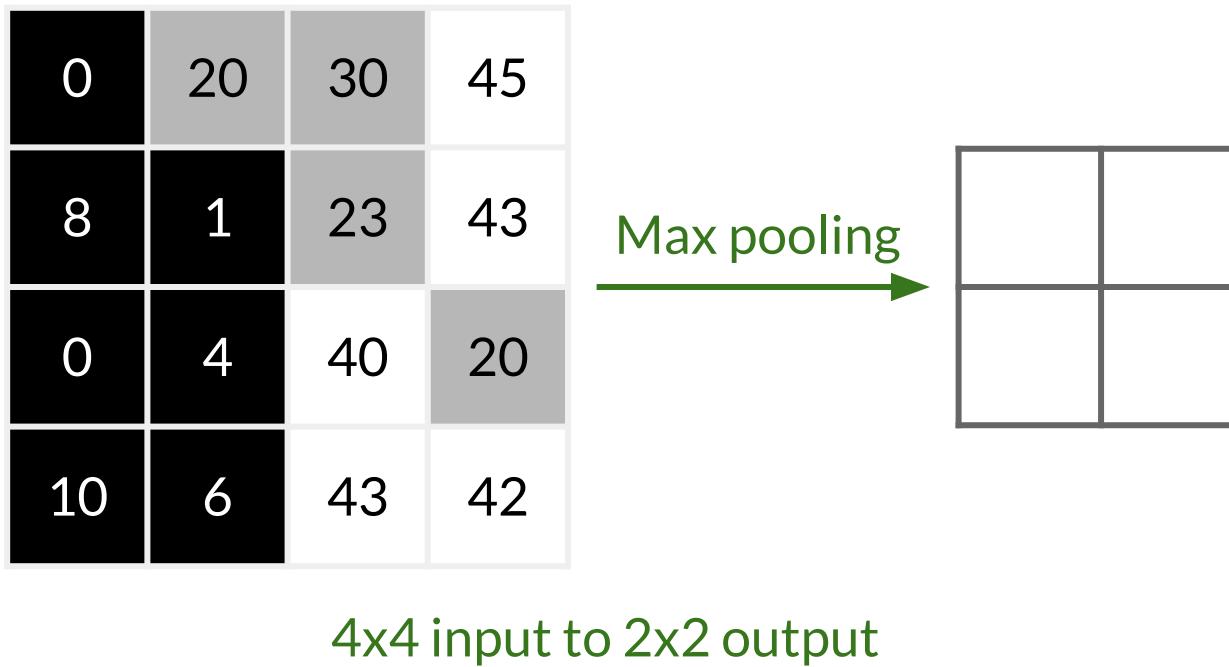
Pooling



Pooling



Max Pooling



Max Pooling

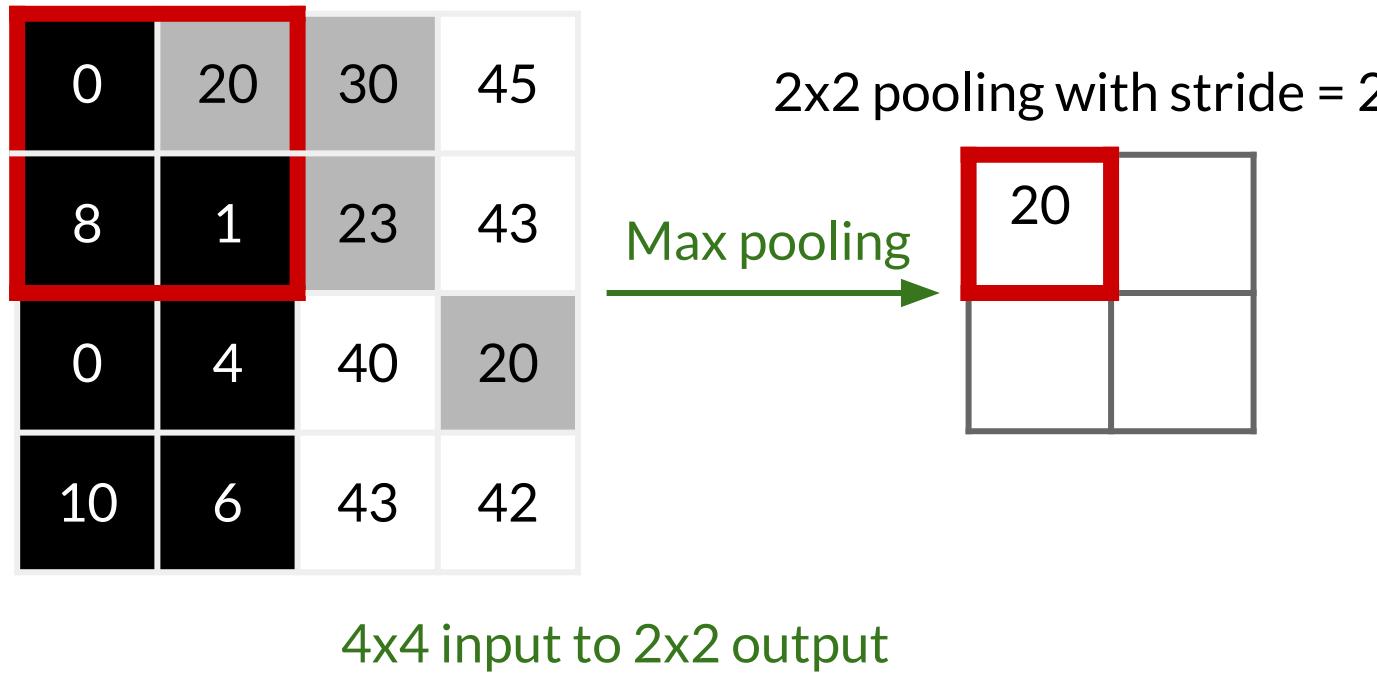
0	20	30	45
8	1	23	43
0	4	40	20
10	6	43	42

2x2 pooling with stride = 2

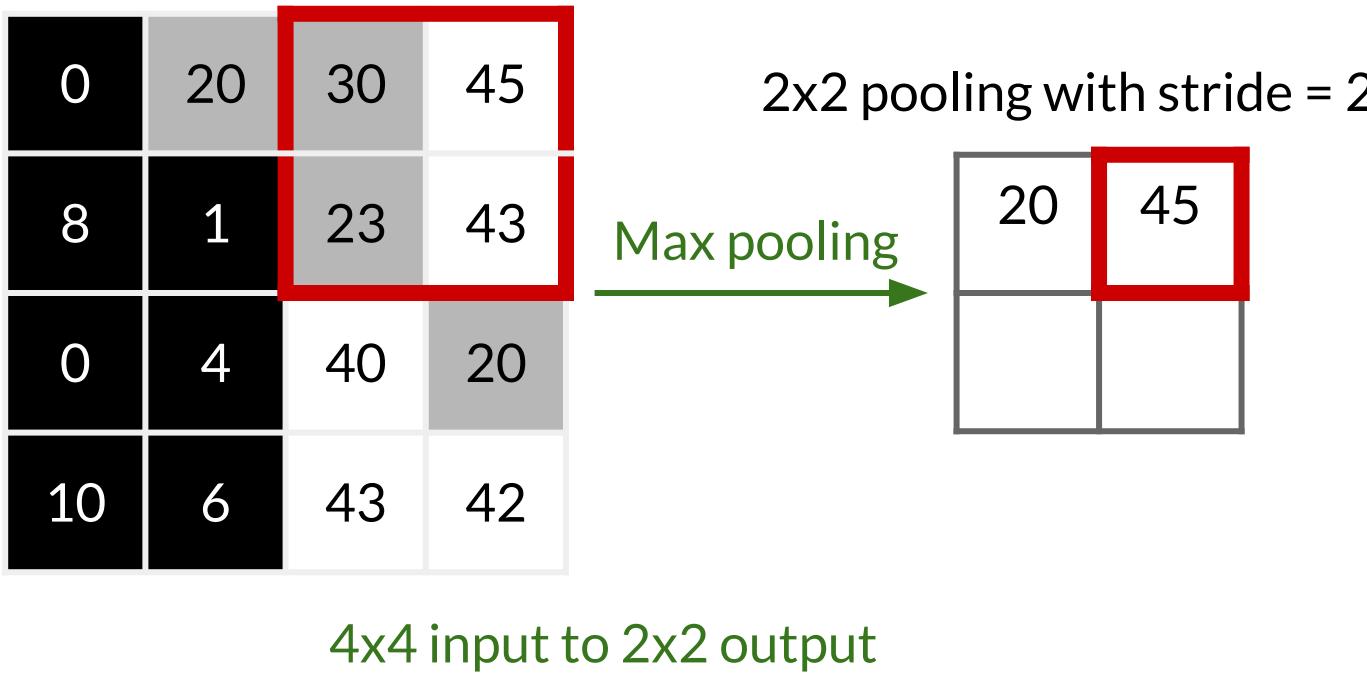
Max pooling

4x4 input to 2x2 output

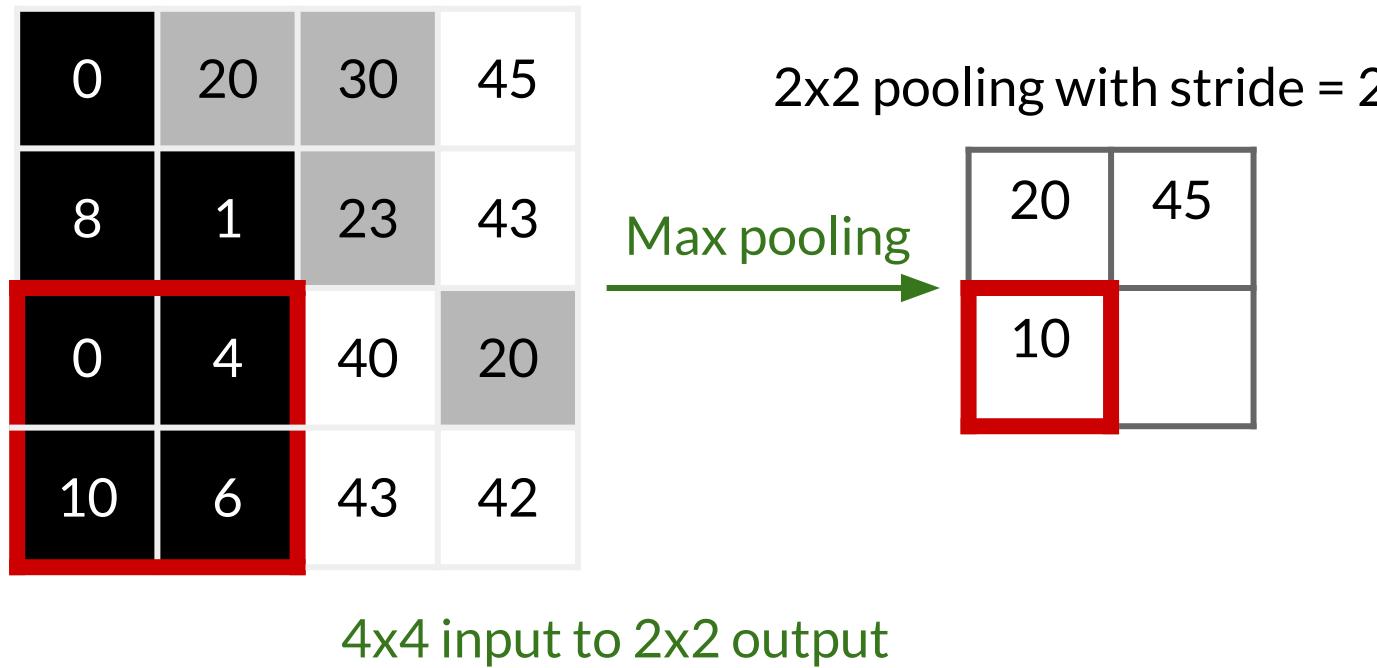
Max Pooling



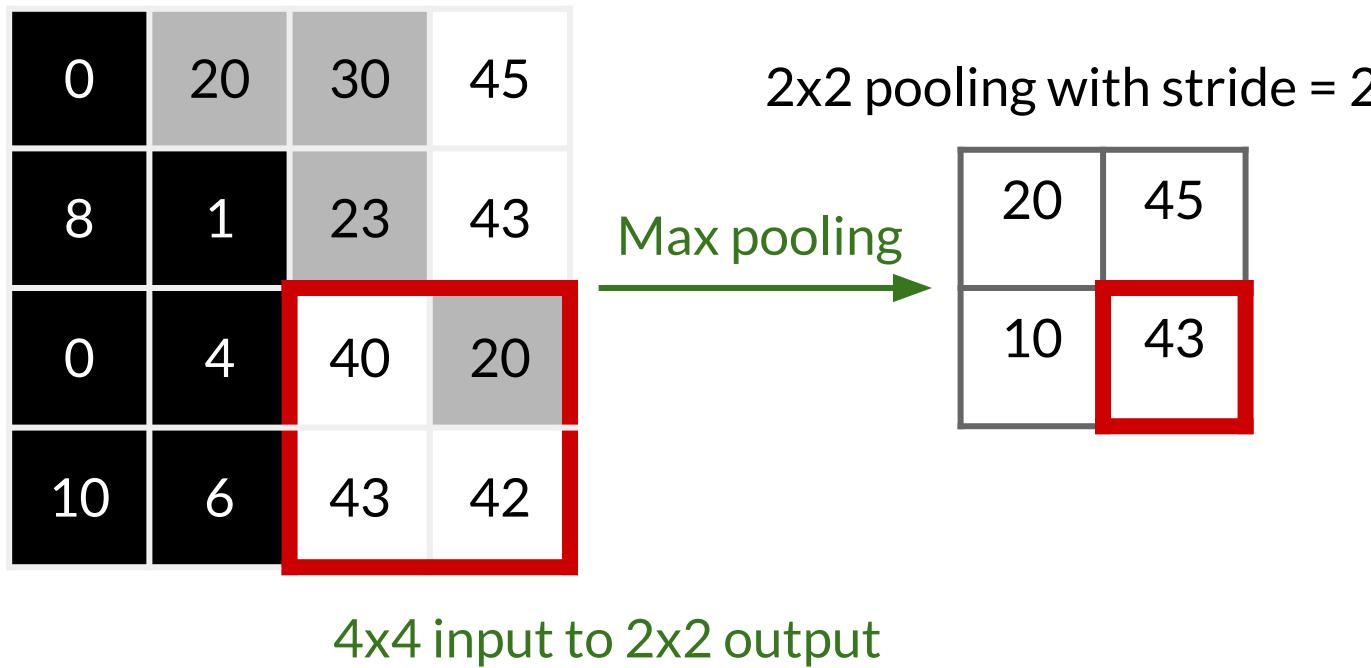
Max Pooling



Max Pooling



Max Pooling



Max Pooling

0	20	30	45
8	1	23	43
0	4	40	20
10	6	43	42

4x4 input to 2x2 output

2x2 pooling with stride = 2

Max pooling

20	45
10	43

Other types include:

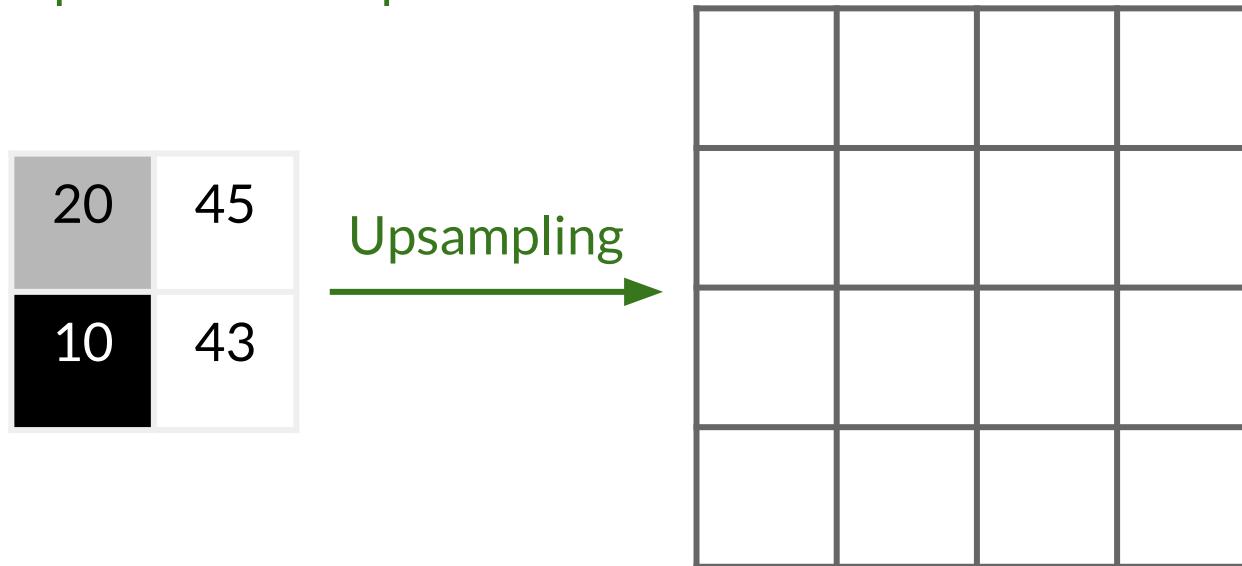
1. Average pooling
2. Min pooling

Upsampling



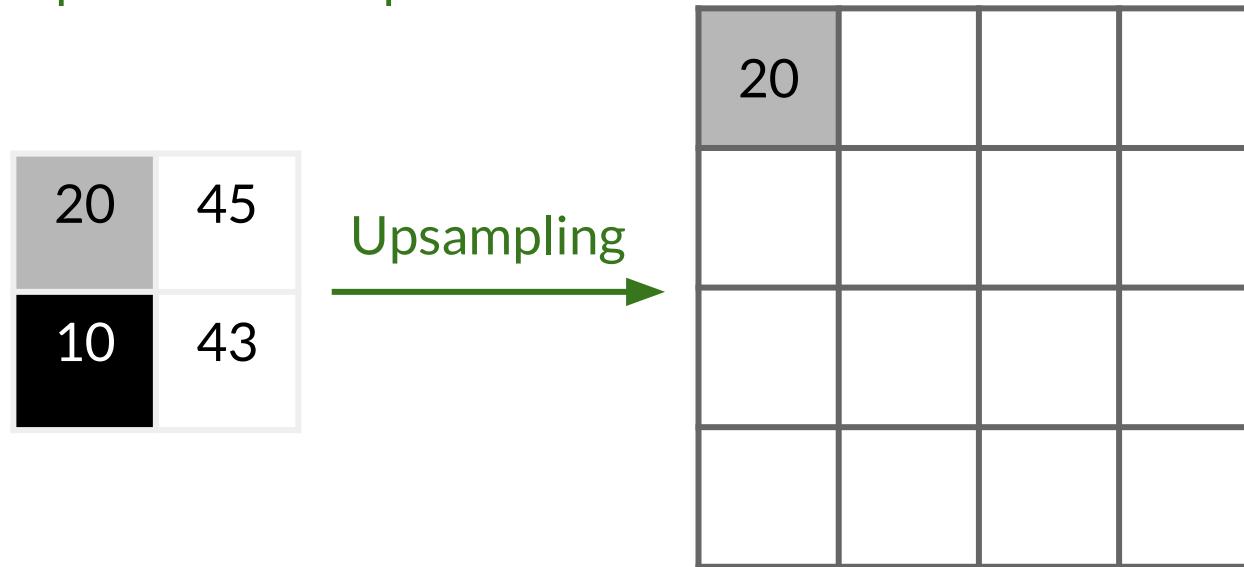
Upsampling: Nearest Neighbors

2x2 input to 4x4 output



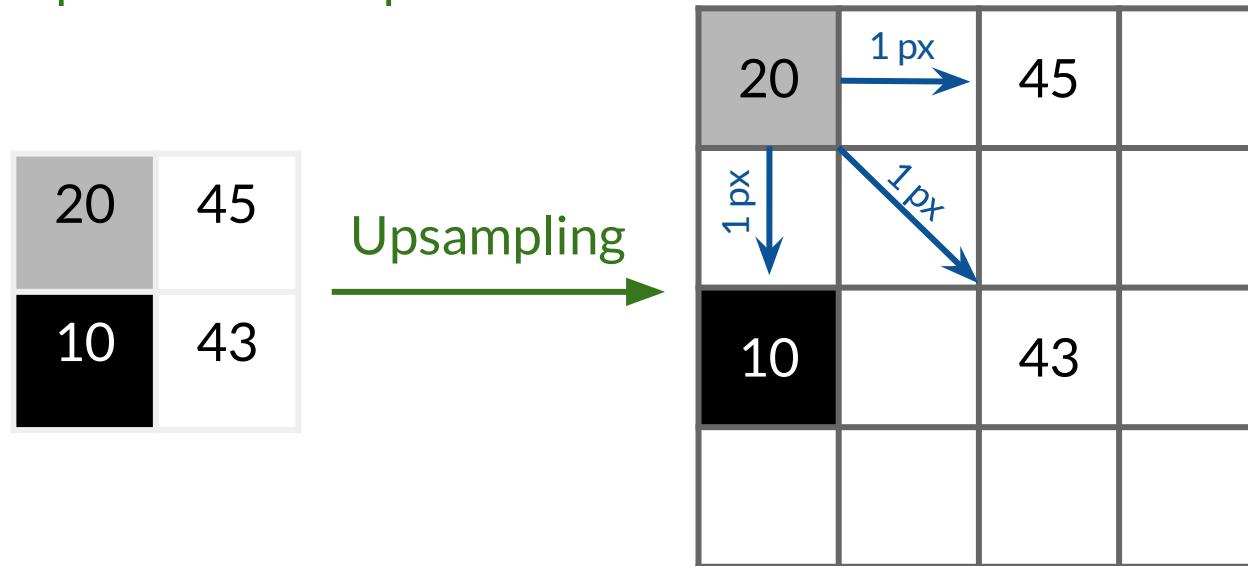
Upsampling: Nearest Neighbors

2x2 input to 4x4 output



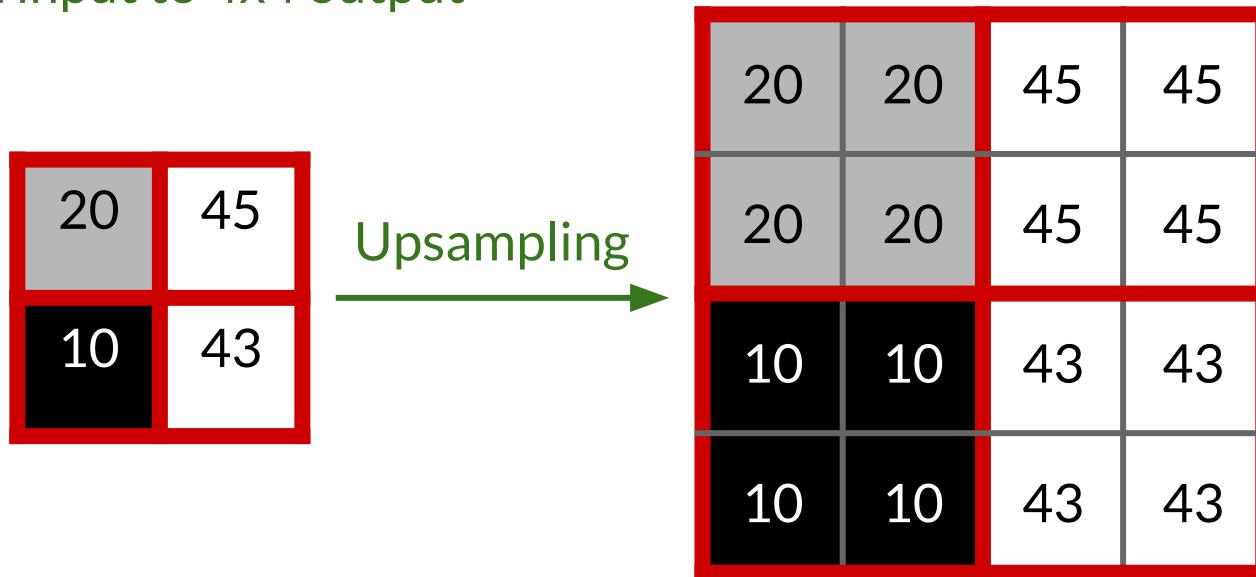
Upsampling: Nearest Neighbors

2x2 input to 4x4 output



Upsampling: Nearest Neighbors

2x2 input to 4x4 output



Upsampling: Nearest Neighbors

2x2 input to 4x4 output

20	45
10	43

Upsampling

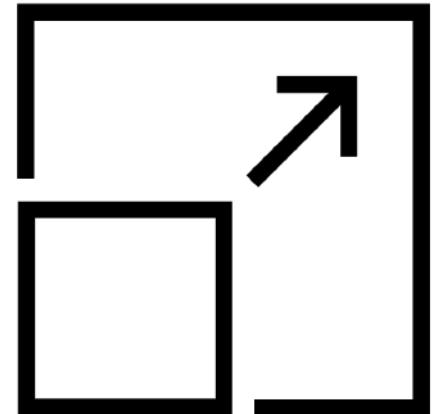
20	20	45	45
20	20	45	45
10	10	43	43
10	10	43	43

Other types include:

1. Linear interpolation
2. Bi-linear interpolation

Summary

- Pooling reduces the size of the input
- Upsampling increases the size of the input
- No learnable parameters!



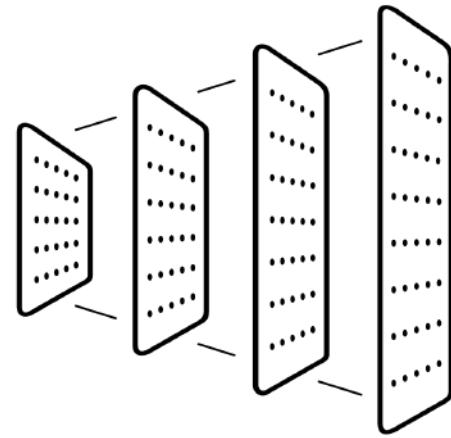


deeplearning.ai

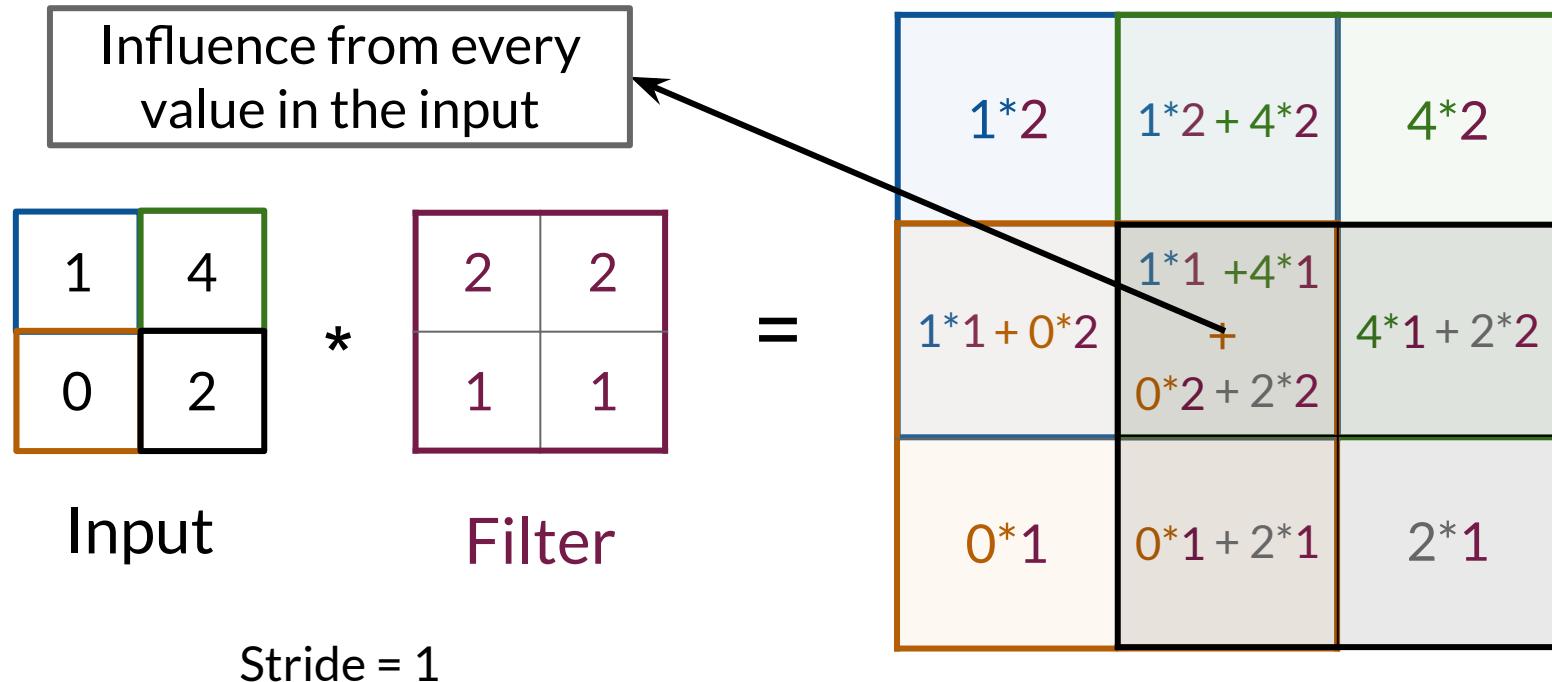
Transposed Convolutions

Outline

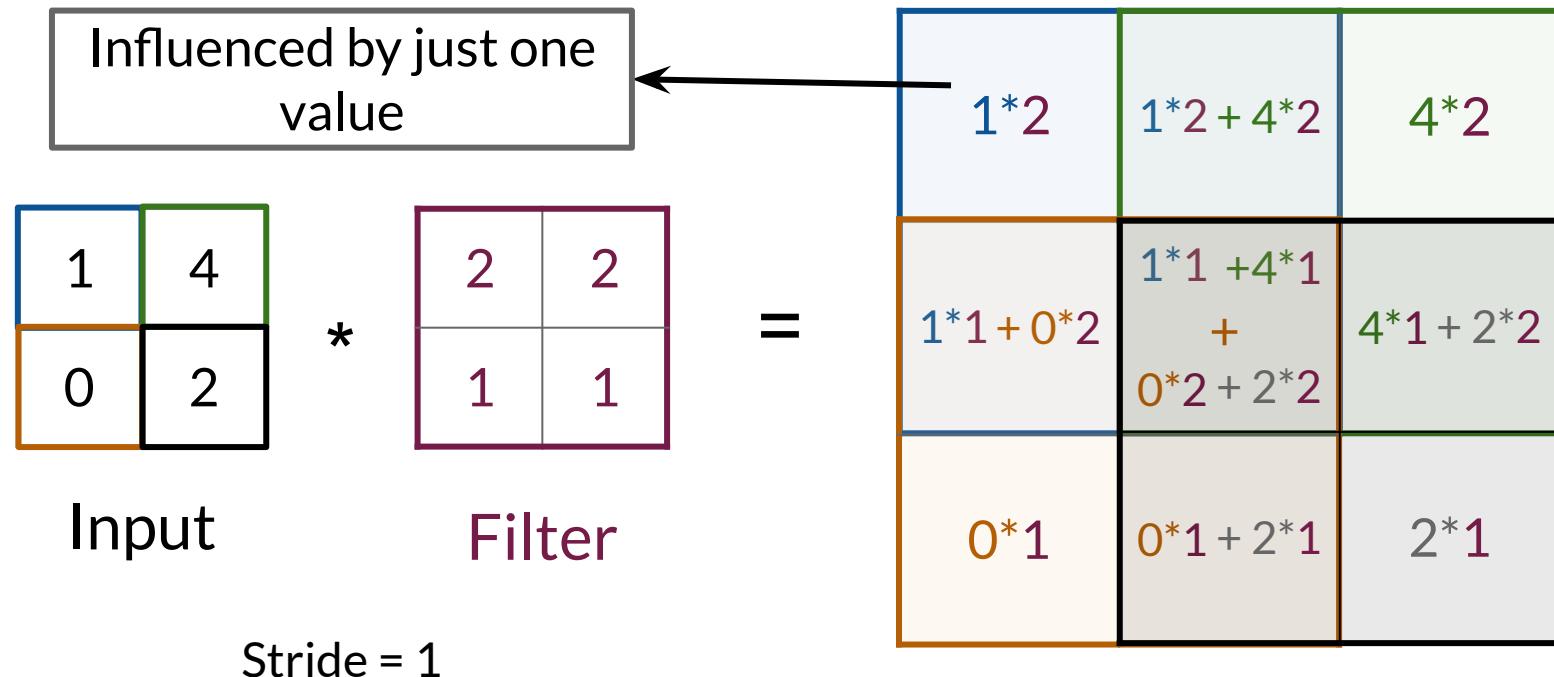
- Transposed convolutions as an upsampling technique
- Issues with transposed convolutions



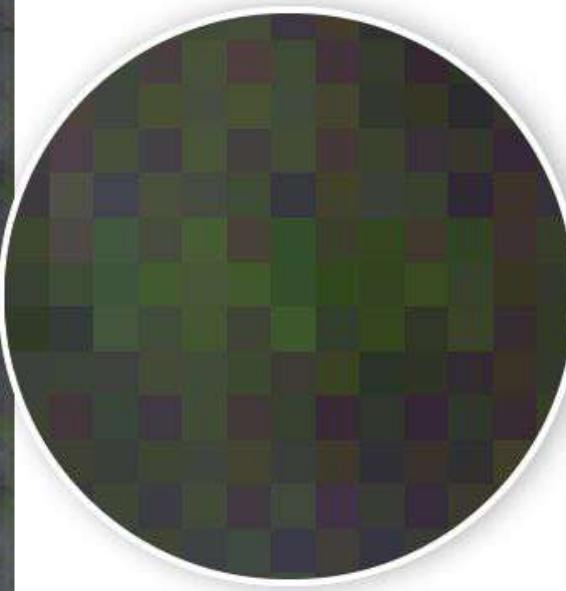
Transposed Convolution



Transposed Convolution



The Problems with Transposed Convolution

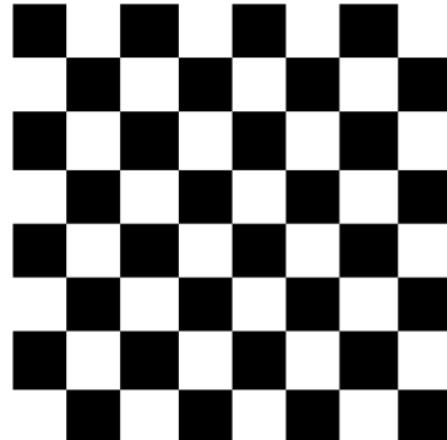


Checkerboard
Pattern

Available from: <http://doi.org/10.23915/distill.00003>

Summary

- Transposed convolutions upsample
- They have learnable parameters
- Problem: results have a checkerboard pattern



Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>

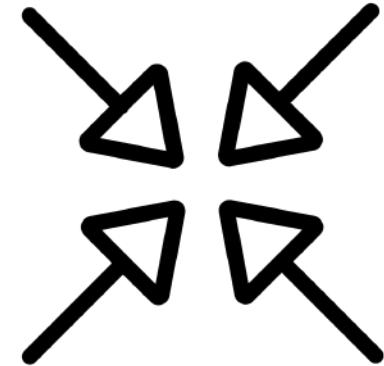


deeplearning.ai

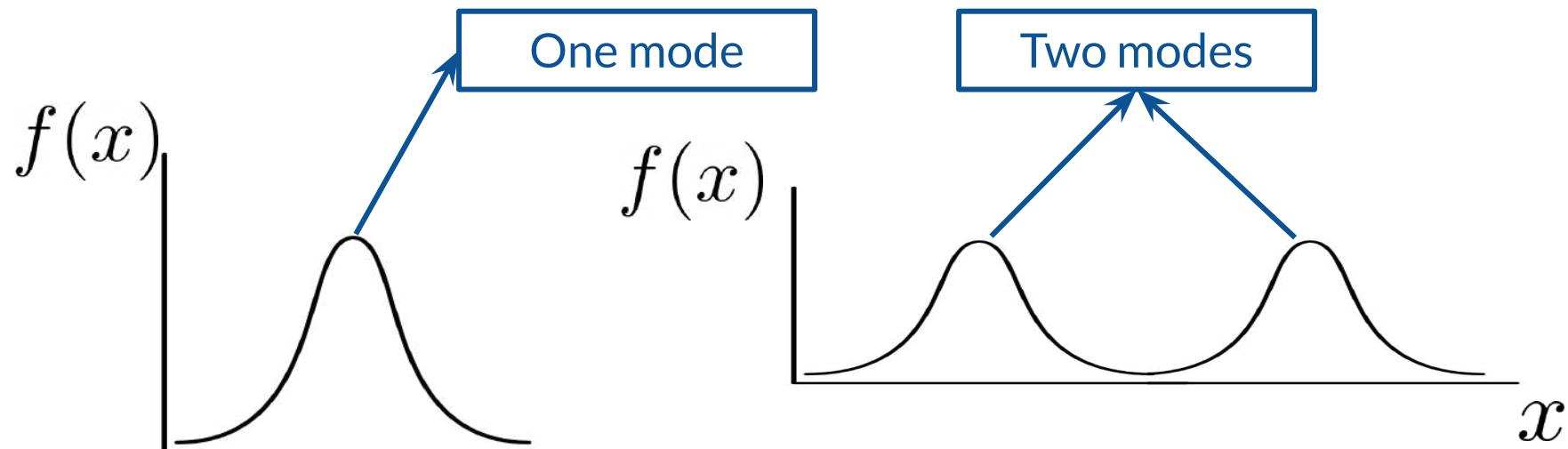
Mode Collapse

Outline

- Modes in distributions
- Mode collapse in GANs
- Intuition behind it during training

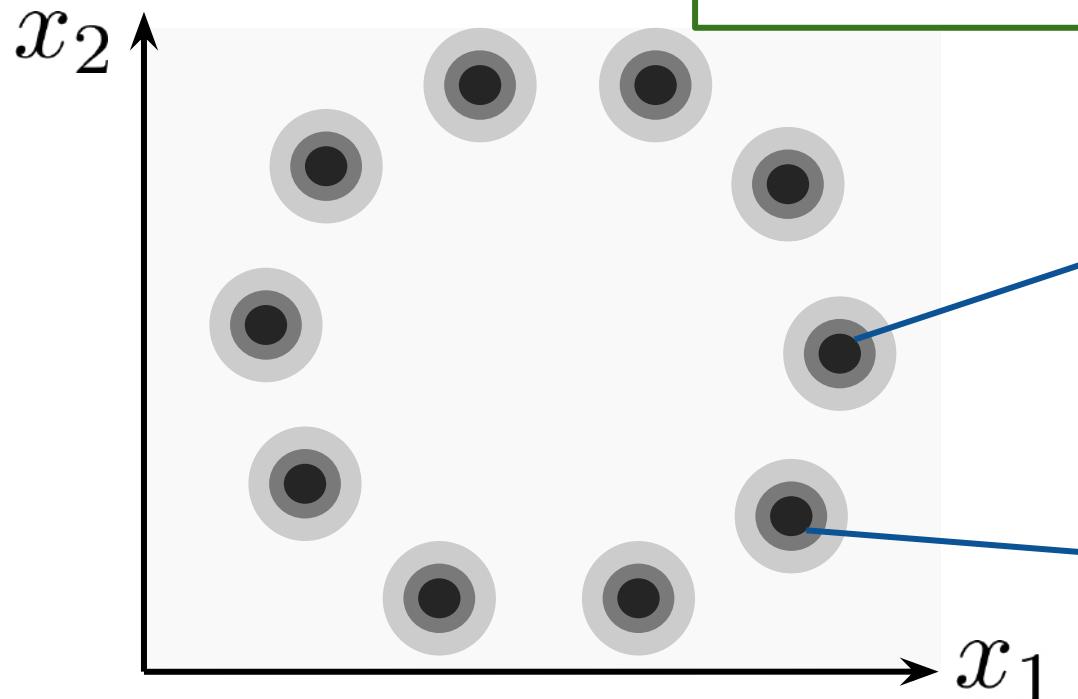


Mode Collapse

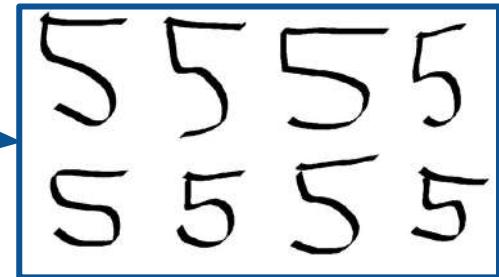
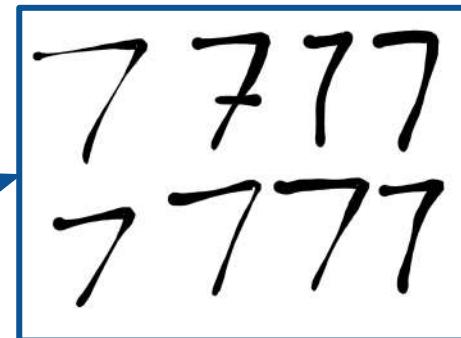


Any peak on the density function
is a mode!

Mode Collapse



10 different modes, 1 per digit



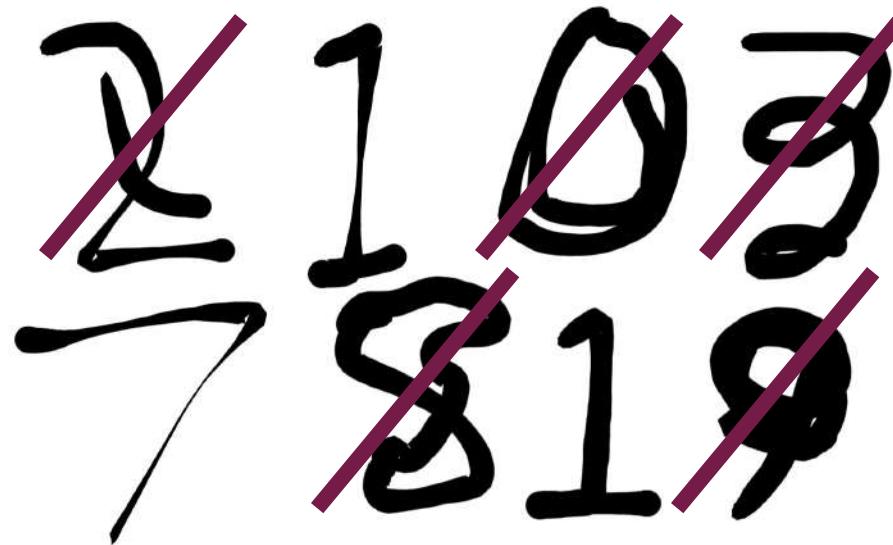
Mode Collapse



2103
7819

Discriminator

Mode Collapse



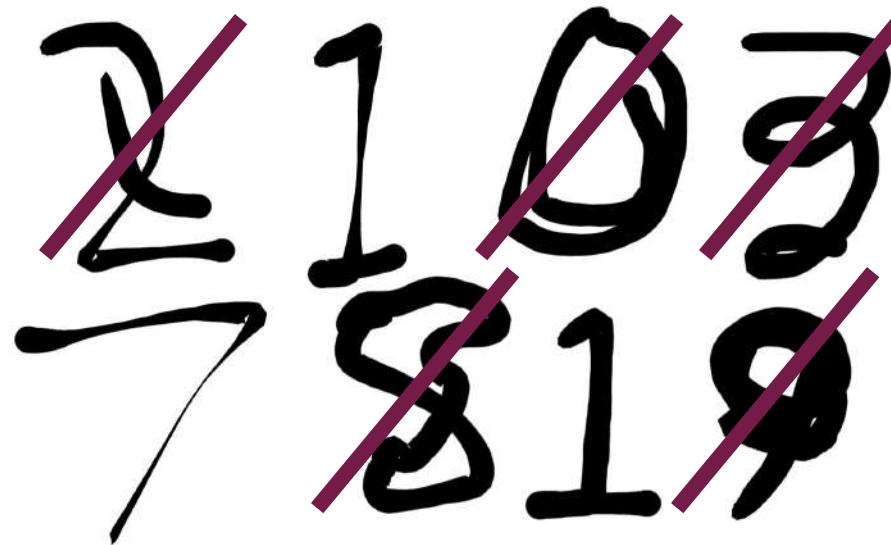
Fakes

Discriminator

Mode Collapse



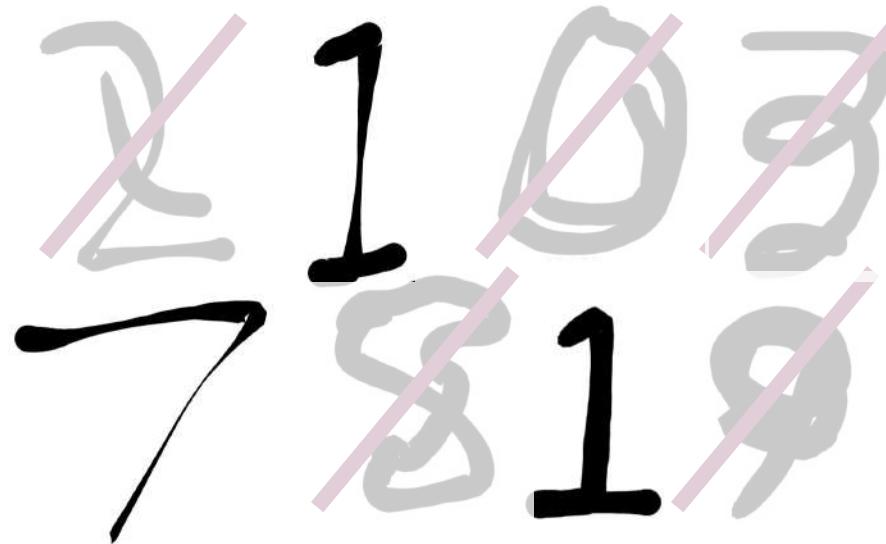
Generator



Mode Collapse



Generator



Fakes that
fooled the
discriminator

Mode Collapse



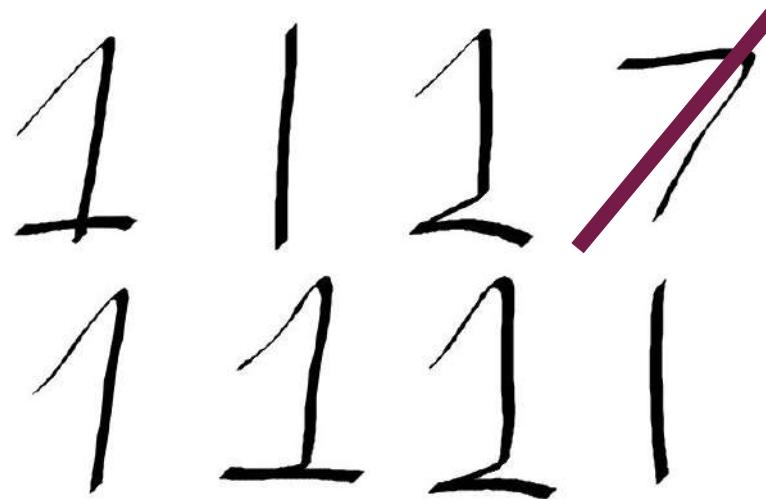
Generator

1 1 1 7
1 1 1 1

Mode Collapse



Discriminator

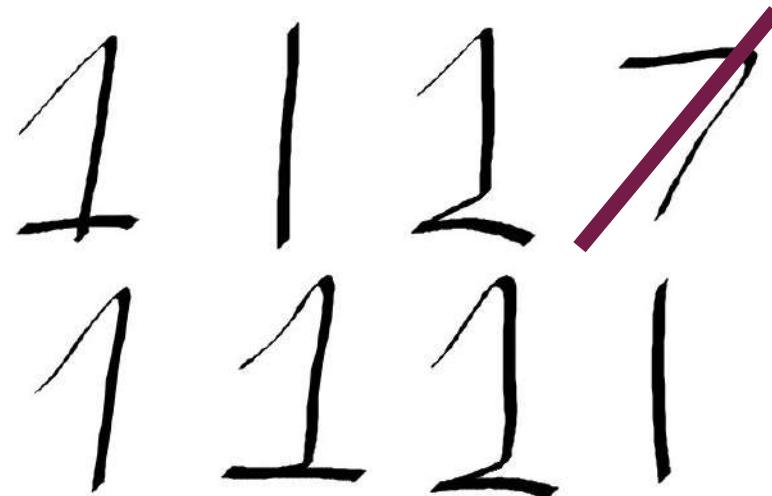


Fakes

Mode Collapse



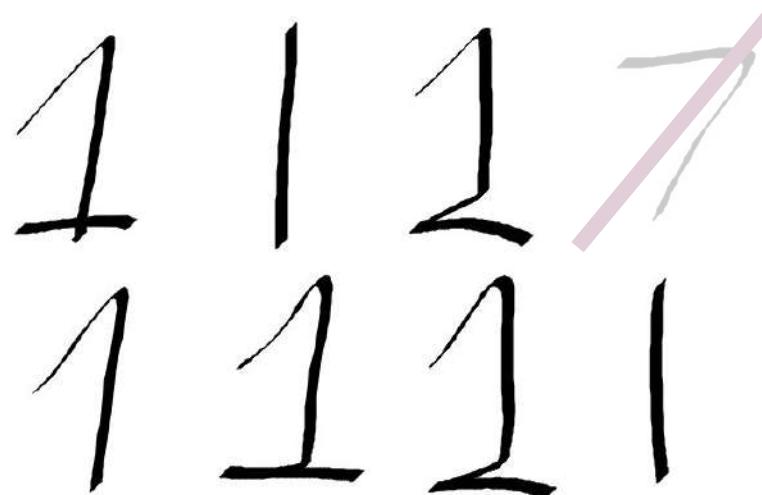
Generator



Mode Collapse



Generator

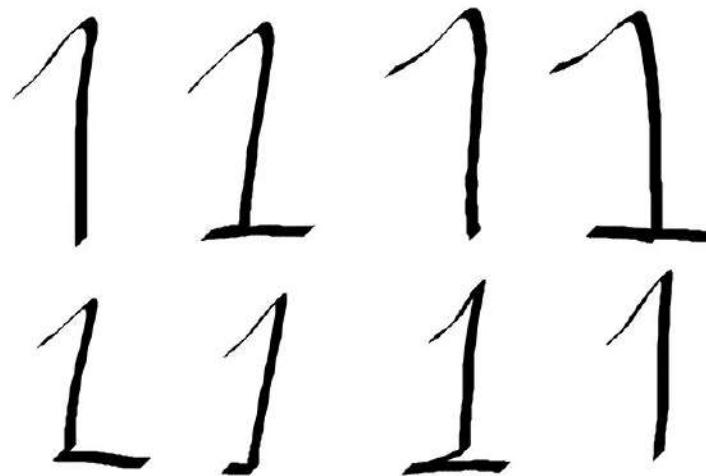


Fakes that
fooled the
discriminator

Mode Collapse



Generator



Summary

- Modes are peaks in the distribution of features
- Typical with real-world datasets
- Mode collapse happens when the generator gets stuck in one mode



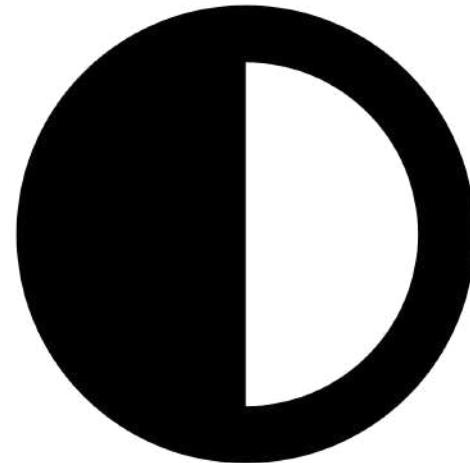


deeplearning.ai

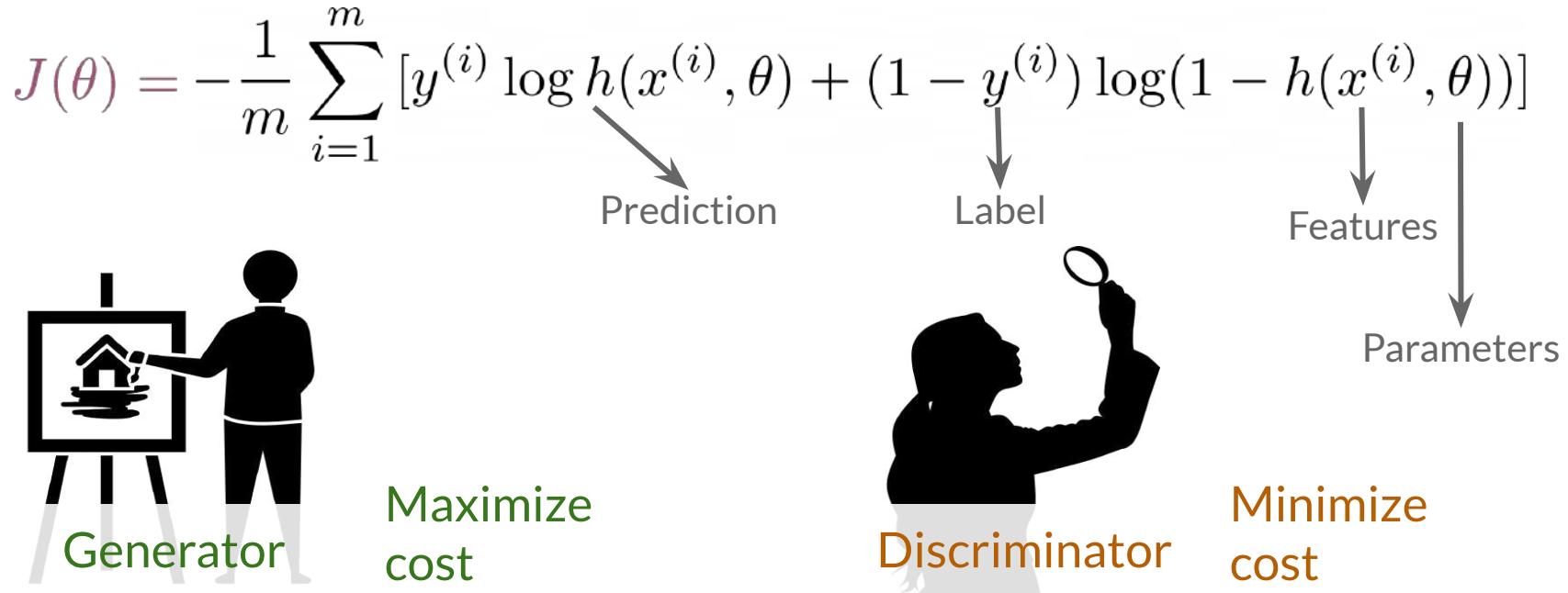
Problem with BCE Loss

Outline

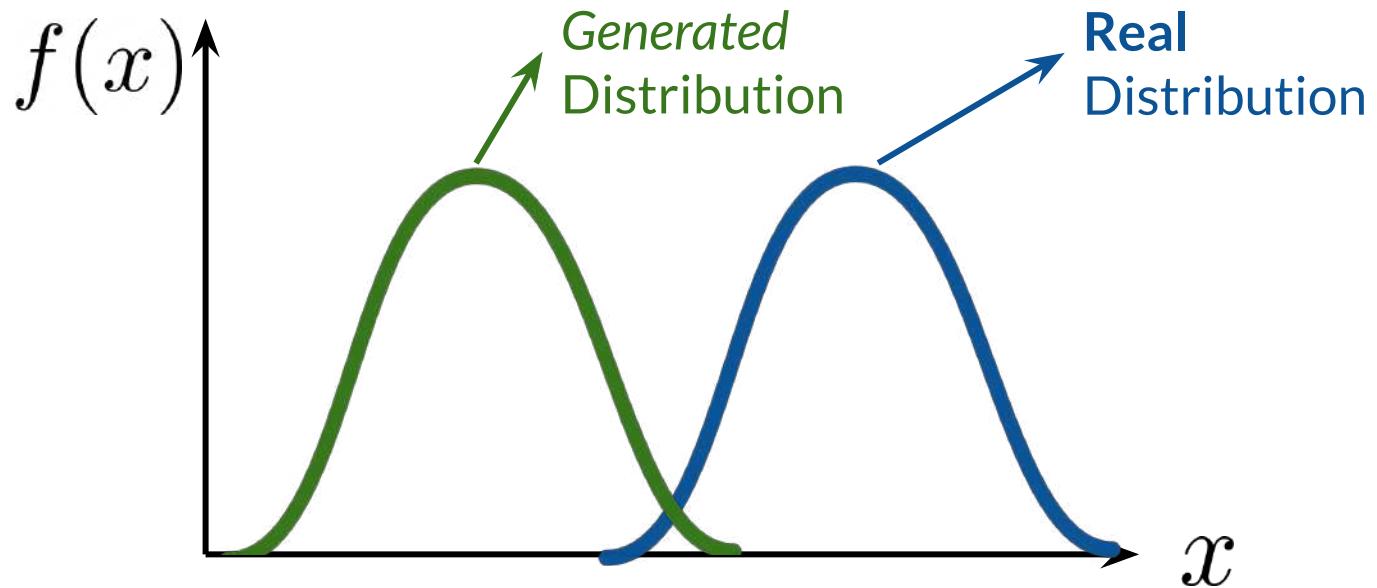
- BCE Loss and the end objective in GANs
- Problem with BCE Loss



BCE Loss in GANs

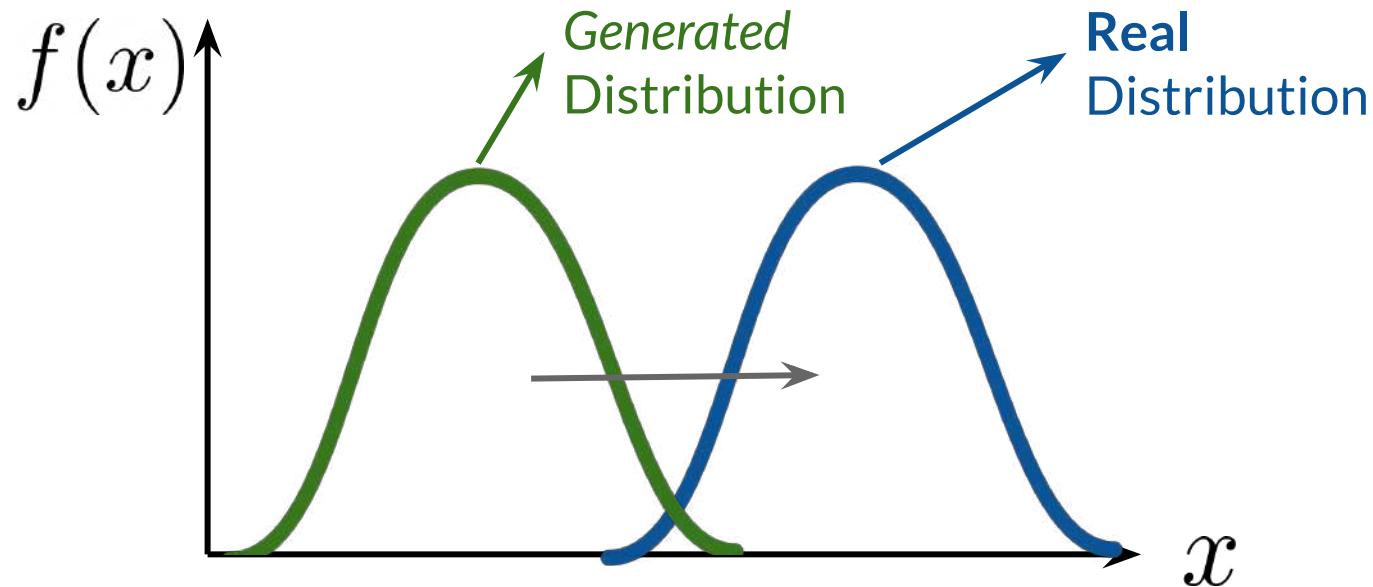


Objective in GANs



Objective in GANs

Make the generated and real distributions look similar



BCE Loss in GANs

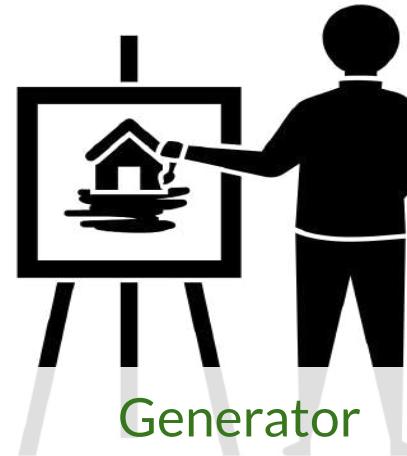
Criticizing is more straightforward



Discriminator

Single output

Easier to train
than the
generator



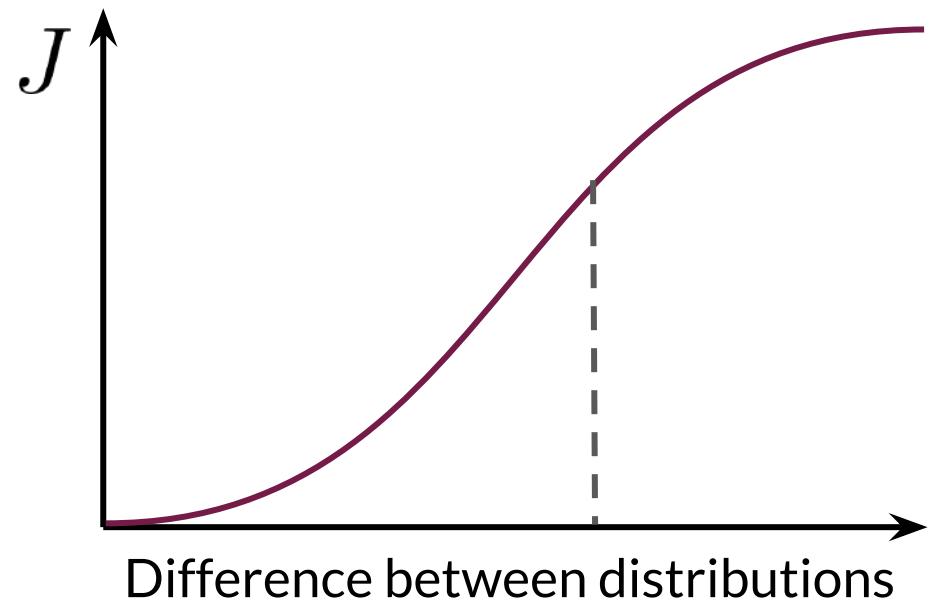
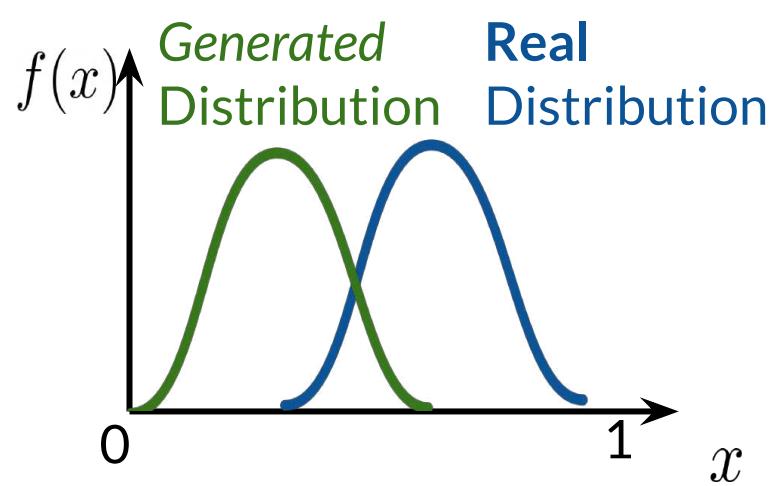
Generator

Complex
output

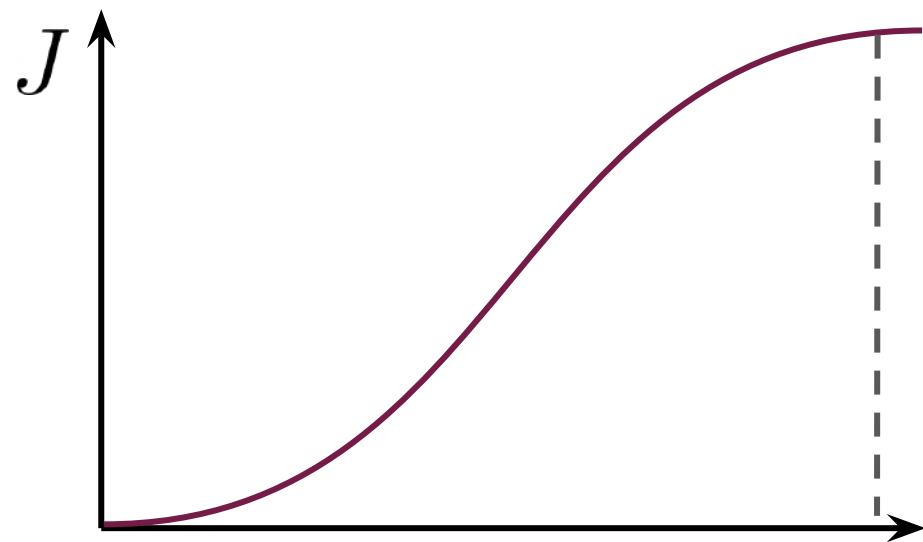
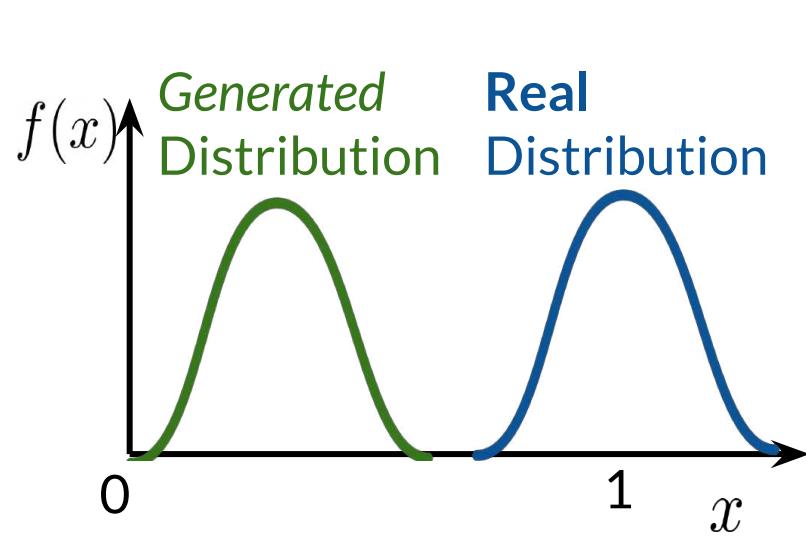
Difficult to
train

Often, the discriminator gets better than the generator

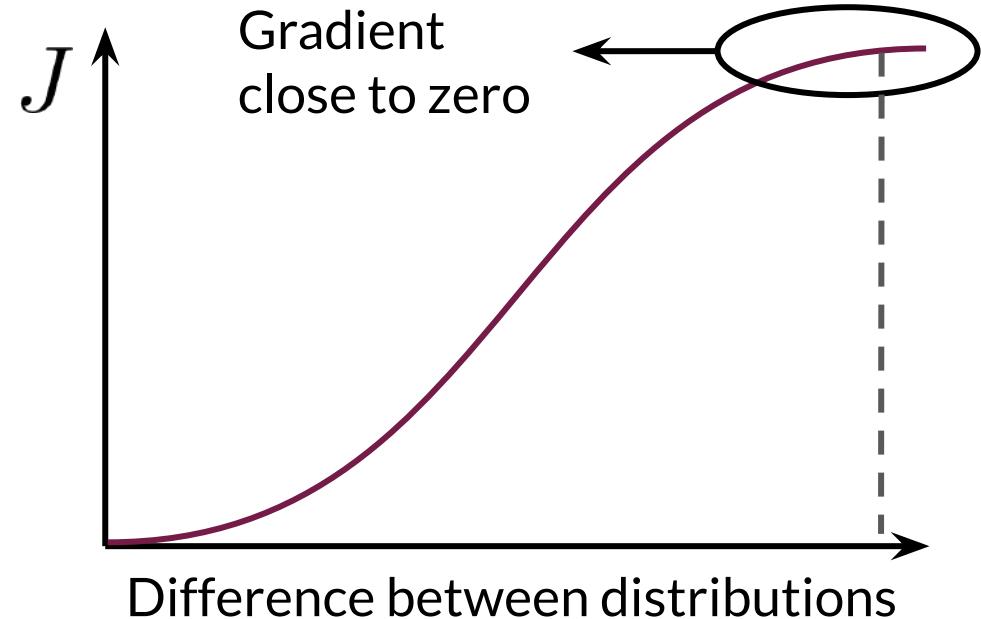
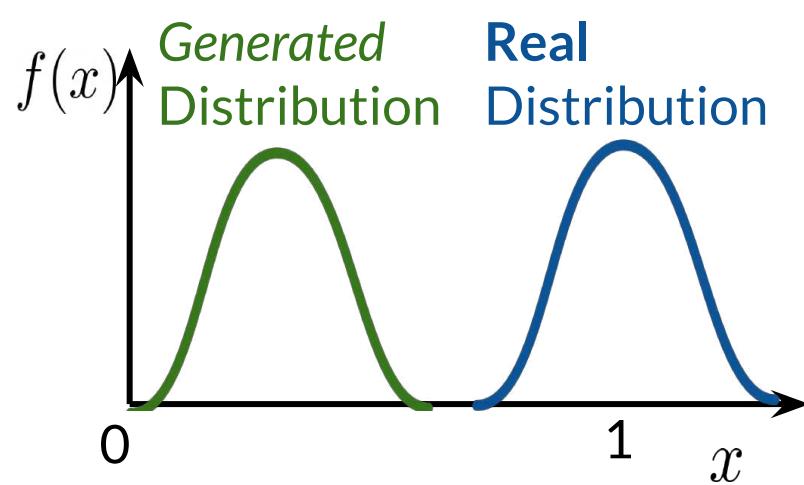
Problems with BCE Loss



Problems with BCE Loss

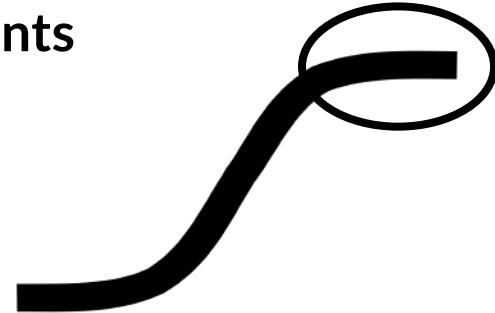


Problems with BCE Loss



Summary

- GANs try to make the real and generated distributions look similar
- When the discriminator improves too much, the function approximated by BCE Loss will contain flat regions
- Flat regions on the cost function = **vanishing gradients**



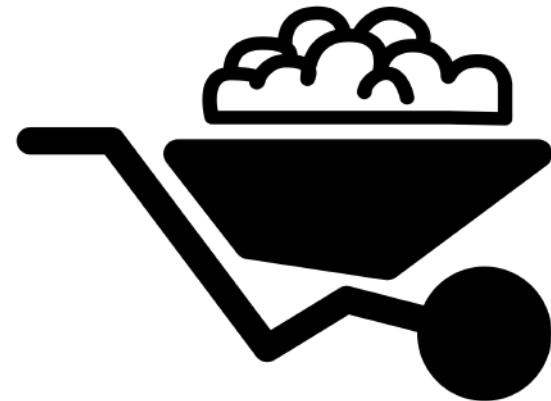


deeplearning.ai

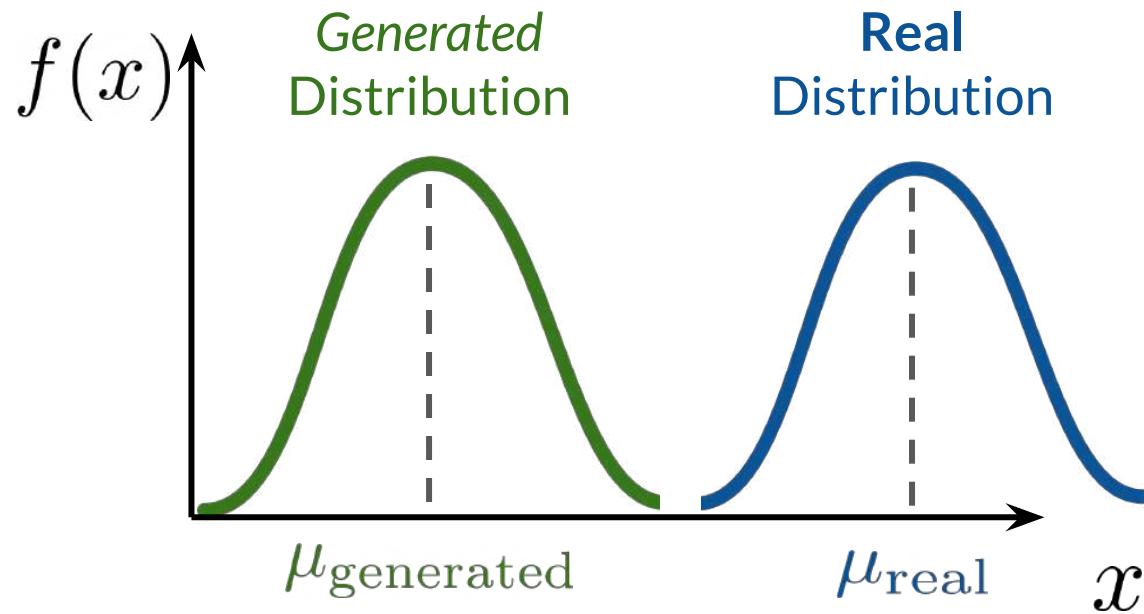
Earth Mover's Distance

Outline

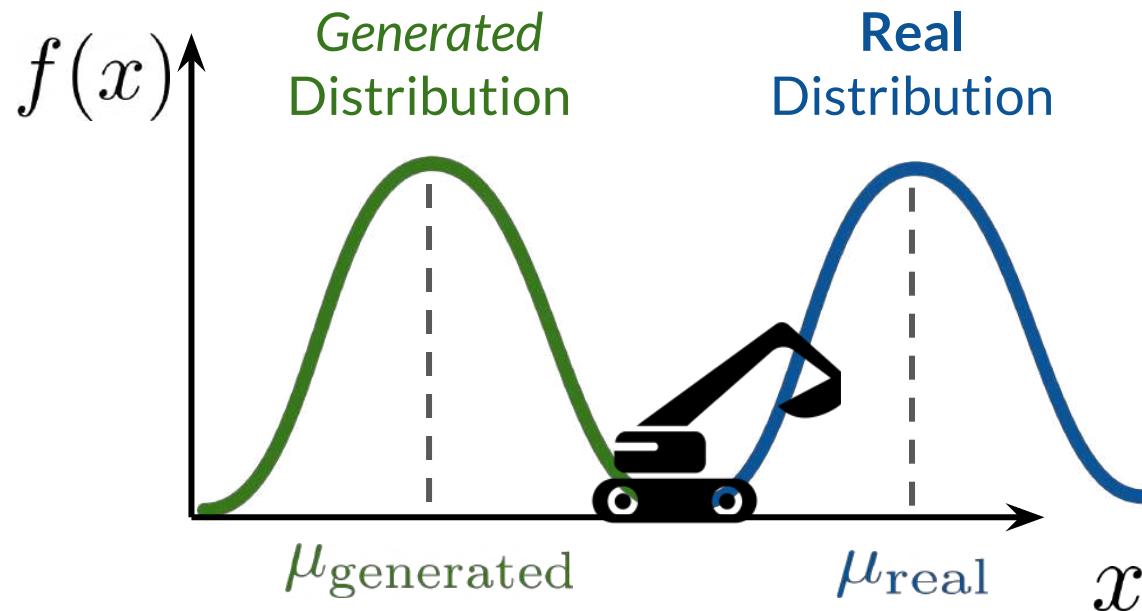
- Earth Mover's Distance (EMD)
- Why it solves the vanishing gradient problem of BCE Loss



Earth Mover's Distance



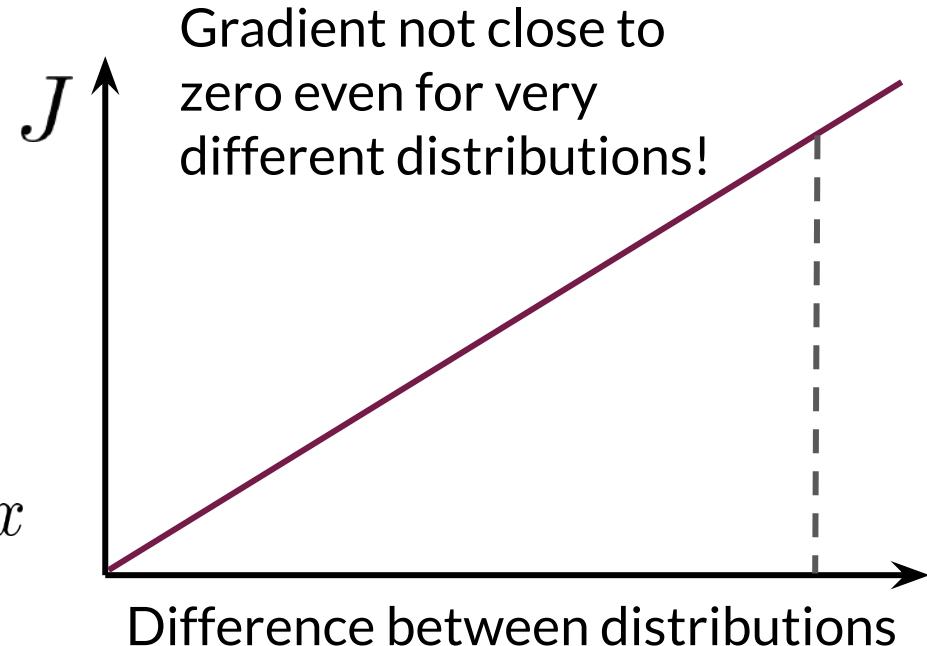
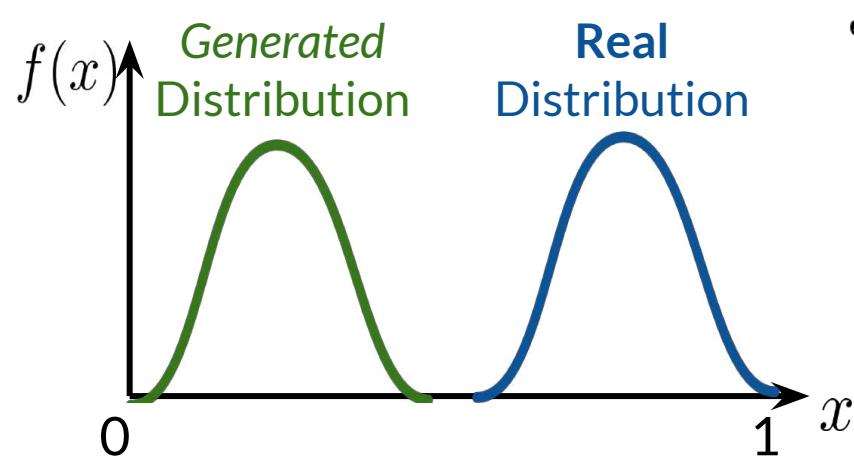
Earth Mover's Distance



Effort to make the *generated* distribution equal to the **real** distribution

Depends on the distance and amount moved

Earth Mover's Distance



Summary

- Earth mover's distance (EMD) is a function of amount and distance
- Doesn't have flat regions when the distributions are very different
- Approximating EMD solves the problems associated with BCE





deeplearning.ai

Wasserstein Loss

Outline

- BCE Loss Simplified
- W-Loss and its comparison with BCE Loss



BCE Loss Simplified

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h(x^{(i)}, \theta) + (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta))]$$

$\min \max$

d

g



Discriminator

Minimize
cost



Generator

Maximize
cost

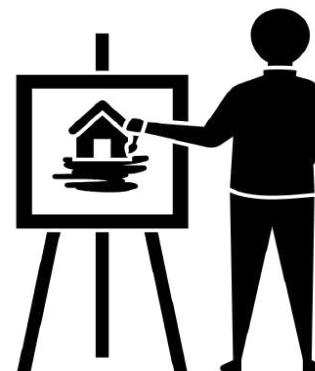
BCE Loss Simplified

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h(x^{(i)}, \theta) + (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta))]$$

$$\min_d \max_g -[\mathbb{E}(\log(d(x))) + \mathbb{E}()]$$



Minimize
cost



Maximize
cost

BCE Loss Simplified

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log h(x^{(i)}, \theta) + (1 - y^{(i)}) \log(1 - h(x^{(i)}, \theta))]$$

$$\min_d \max_g -[\mathbb{E}(\log(d(x))) + \mathbb{E}(1 - \log(d(g(z))))]$$



Minimize
cost



Maximize
cost

W-Loss

W-Loss approximates the Earth Mover's Distance

W-Loss

W-Loss approximates the Earth Mover's Distance

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))$$

W-Loss

W-Loss approximates the Earth Mover's Distance

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))$$



Maximize
the
distance

W-Loss

W-Loss approximates the Earth Mover's Distance

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))$$



Generator

Minimize
the
distance

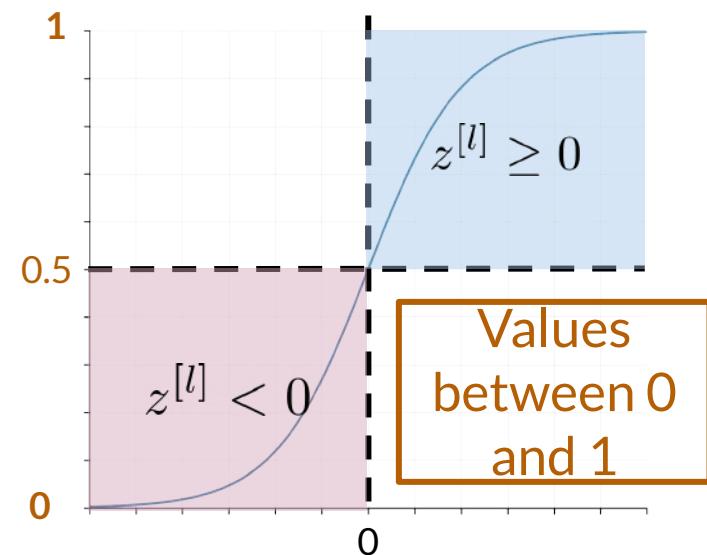


Critic

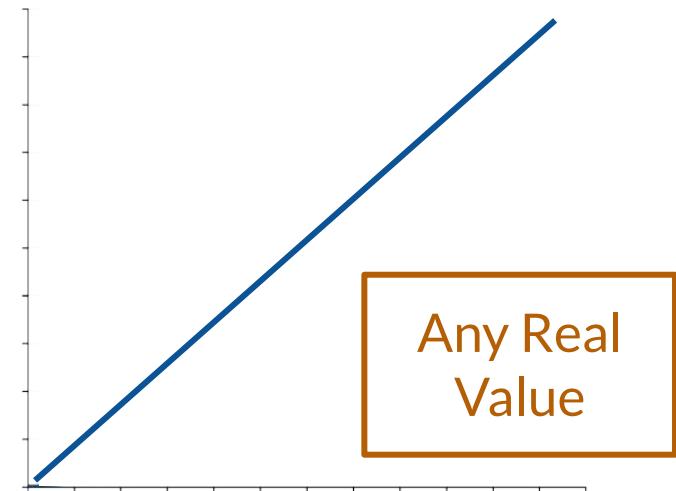
Maximize
the
distance

Discriminator Output

Discriminator output

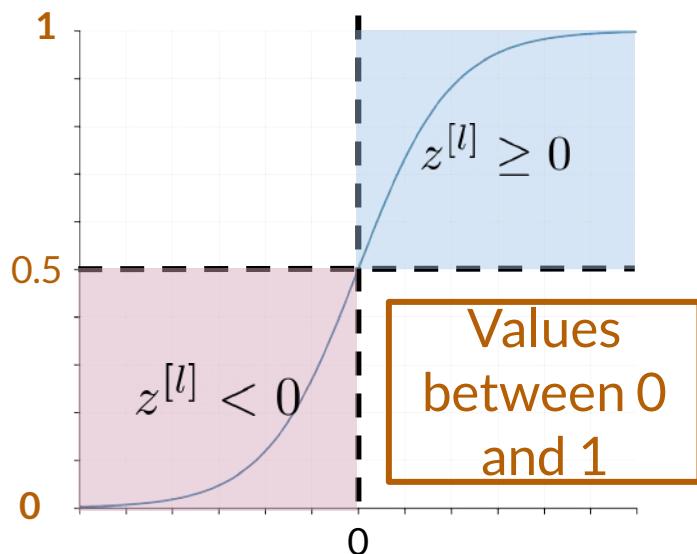


Discriminator output

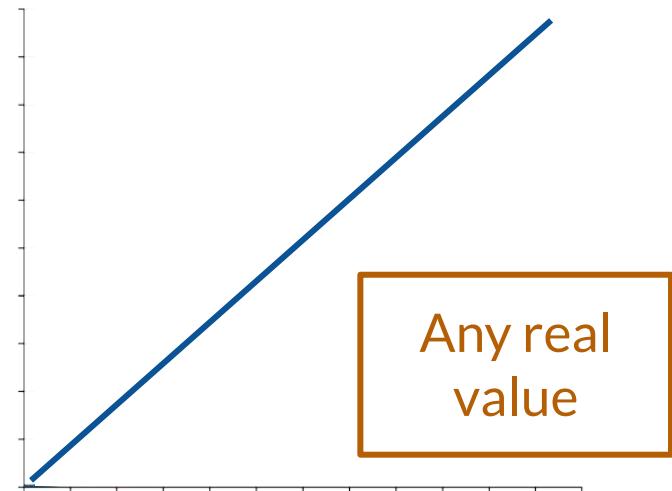


Discriminator Output

Discriminator output



Discriminator output
Critic



W-Loss vs BCE Loss

BCE Loss

Discriminator outputs between 0 and 1

$$-\left[\mathbb{E}(\log(d(x))) + \mathbb{E}(1 - \log(d(g(z))))\right]$$

W-Loss

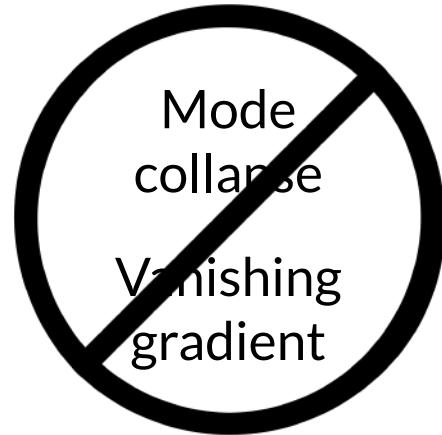
Critic outputs any number

$$\mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))$$

W-Loss helps with mode collapse and vanishing gradient problems

Summary

- W-Loss looks very similar to BCE Loss
- W-Loss prevents mode collapse and vanishing gradient problems





deeplearning.ai

Condition on Wasserstein Critic

Outline

- Continuity condition on the critic's neural network
- Why this condition matters



Condition on W-Loss

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))$$

Condition on W-Loss

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))$$

Condition on W-Loss

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))$$

Condition on W-Loss

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))$$

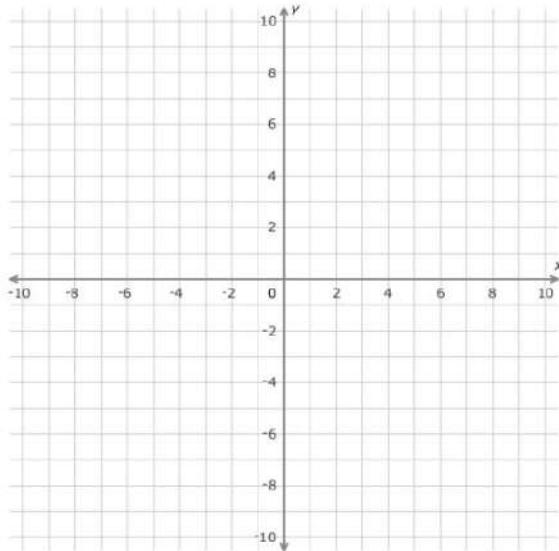
The diagram illustrates the components of the loss function. A green arrow points from the variable g to the term $\mathbb{E}(c(g(z)))$. A brown arrow points from the function c to the same term, indicating that both the function and its input $g(z)$ contribute to the calculation of the expectation.

Needs to be 1-Lipschitz Continuous

Condition on W-Loss

Critic needs to be **1-L Continuous**

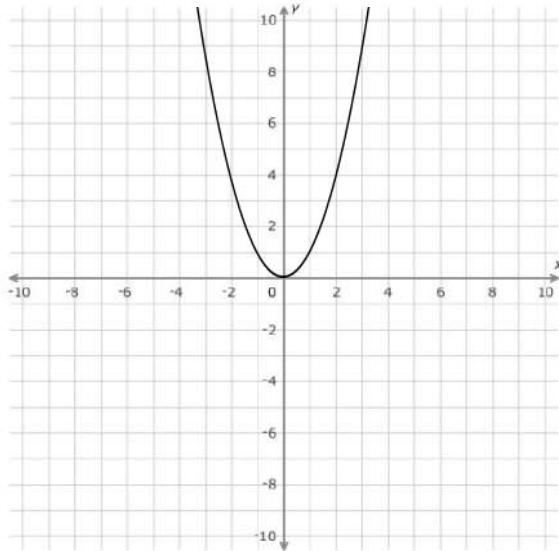
The norm of the gradient should be at most **1** for every point



Condition on W-Loss

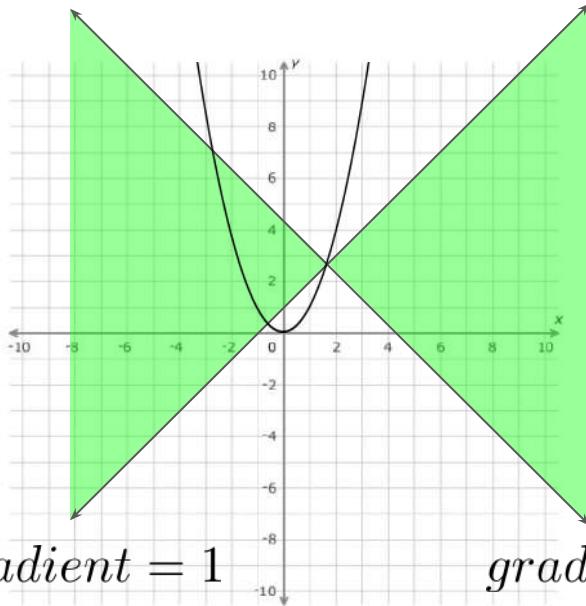
Critic needs to be **1-L Continuous**

The norm of the gradient should be at most **1** for every point



Condition on W-Loss

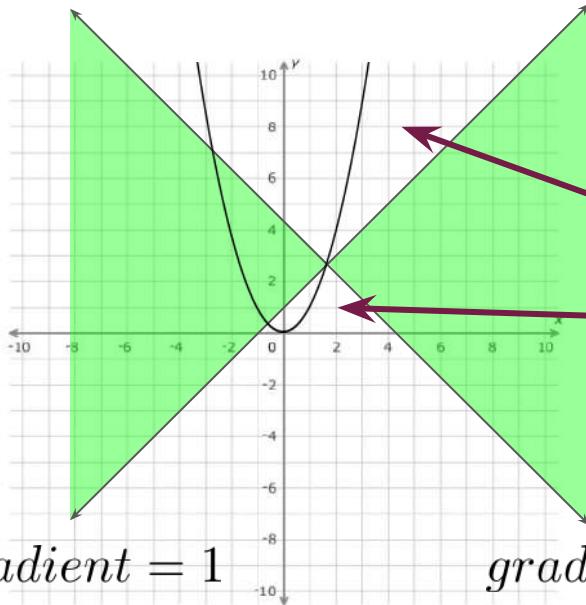
Critic needs to be **1-L Continuous**



The norm of the gradient should be at most **1** for every point

Condition on W-Loss

Critic needs to be **1-L Continuous**



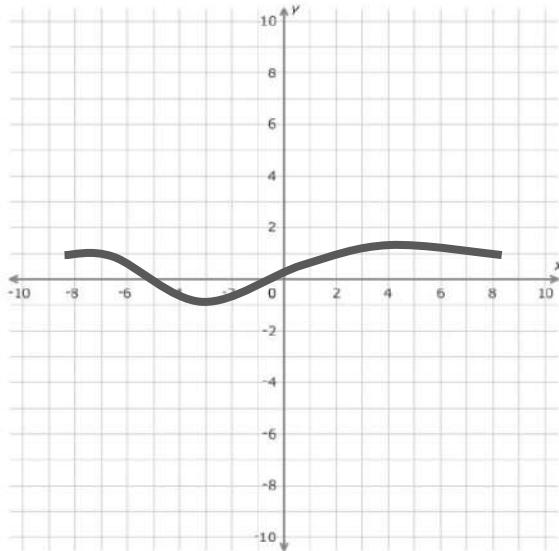
The norm of the gradient should be at most **1** for every point

Not 1-L Continuous

Condition on W-Loss

Critic needs to be **1-L Continuous**

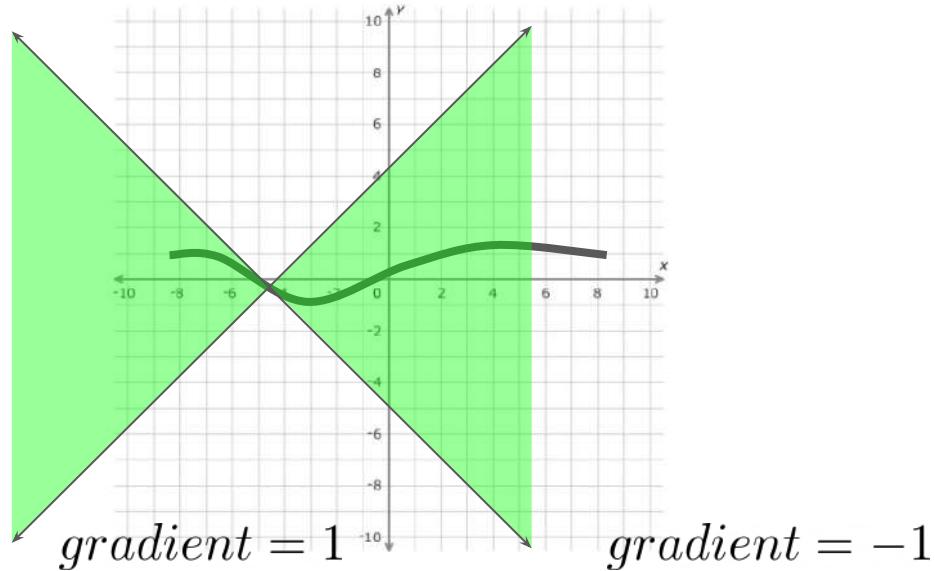
The norm of the gradient should be at most **1** for every point



Condition on W-Loss

Critic needs to be **1-L Continuous**

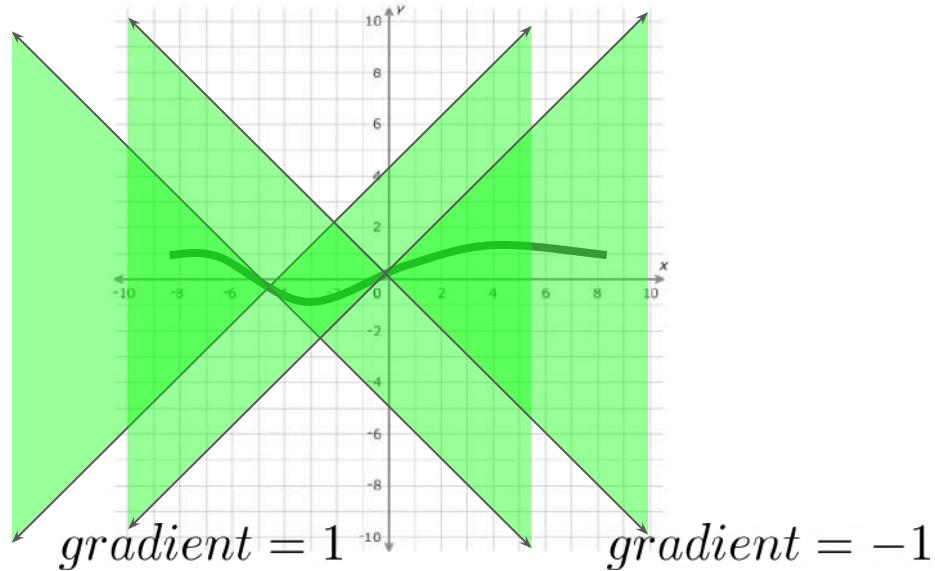
The norm of the gradient should be at most **1** for every point



Condition on W-Loss

Critic needs to be **1-L Continuous**

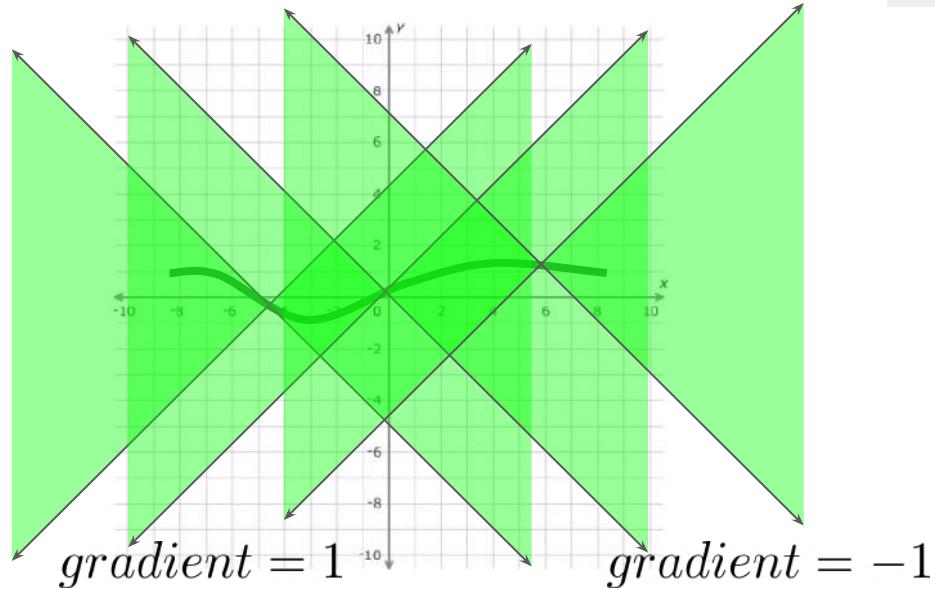
The norm of the gradient should be at most **1** for every point



Condition on W-Loss

Critic needs to be **1-L Continuous**

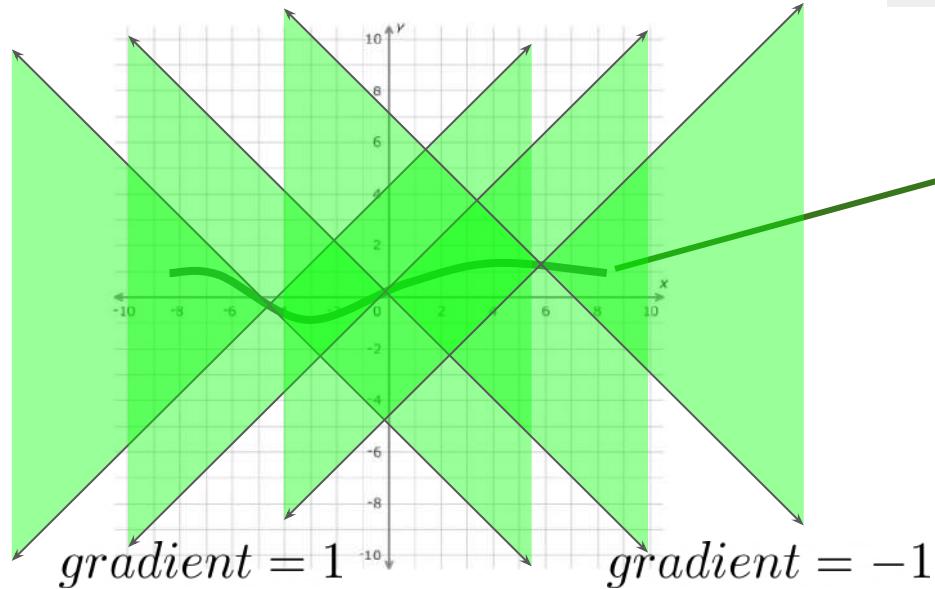
The norm of the gradient should be at most **1** for every point



Condition on W-Loss

Critic needs to be **1-L Continuous**

The norm of the gradient should be at most **1** for every point



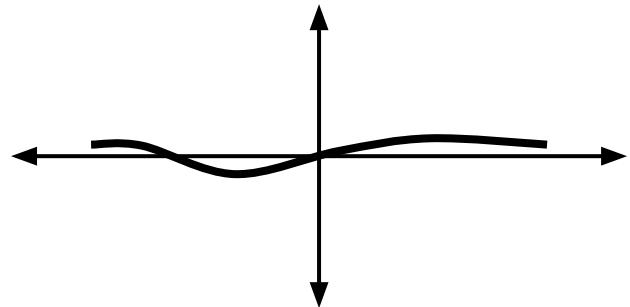
1-L Continuous

W-Loss is valid

Needed for training stable neural networks with W-Loss

Summary

- Critic's neural network needs to be 1-L Continuous when using W-Loss
- This condition ensures that W-Loss is validly approximating Earth Mover's Distance





deeplearning.ai

1-Lipschitz Continuity Enforcement

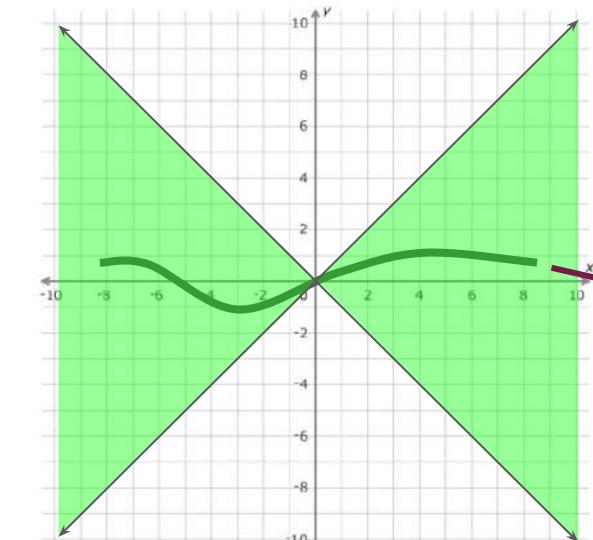
Outline

- Weight clipping and gradient penalty
- Advantages of gradient penalty



1-L Enforcement

Critic needs to be 1-L Continuous



Norm of the gradient at most 1

$$\|\nabla f(x)\|_2 \leq 1$$

Slope of the function
at most 1

1-L Enforcement: Weight Clipping

Weight clipping forces the weights of the critic to a fixed interval

Gradient descent to update weights

Clip the critic's weights

Limits the learning ability of the critic

1-L Enforcement: Gradient Penalty

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z))) + \lambda \text{reg}$$



Regularization of the
critic's gradient

1-L Enforcement: Gradient Penalty

Real

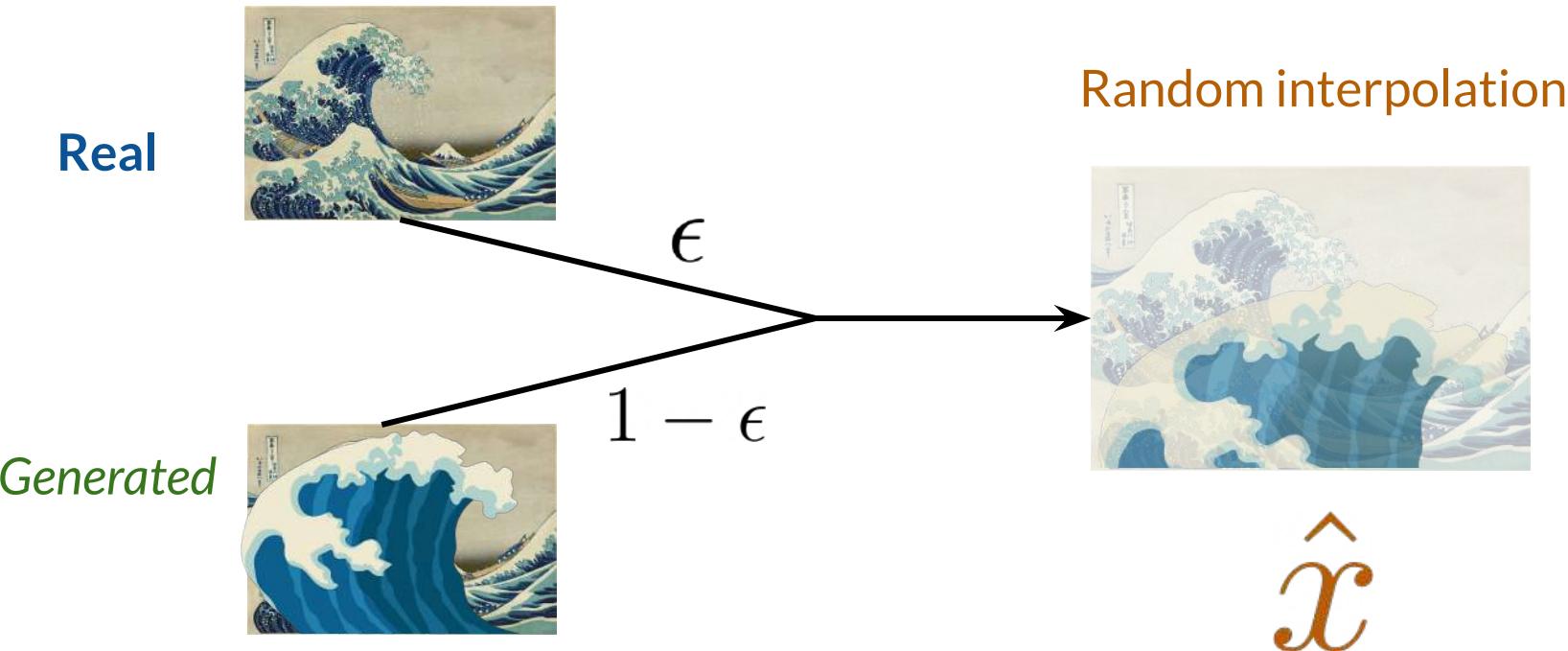


ϵ

Random interpolation



1-L Enforcement: Gradient Penalty



1-L Enforcement: Gradient Penalty

$$\mathbb{E}(\|\nabla c(\hat{x})\|_2 - 1)^2 \quad \text{Regularization term}$$

1-L Enforcement: Gradient Penalty

$$\mathbb{E}(\|\nabla c(\hat{x})\|_2 - 1)^2 \quad \text{Regularization term}$$

1-L Enforcement: Gradient Penalty

$$\mathbb{E}(\|\nabla c(\hat{x})\|_2 - 1)^2 \quad \text{Regularization term}$$

$$\epsilon x + (1 - \epsilon)g(z) \quad \text{Interpolation}$$

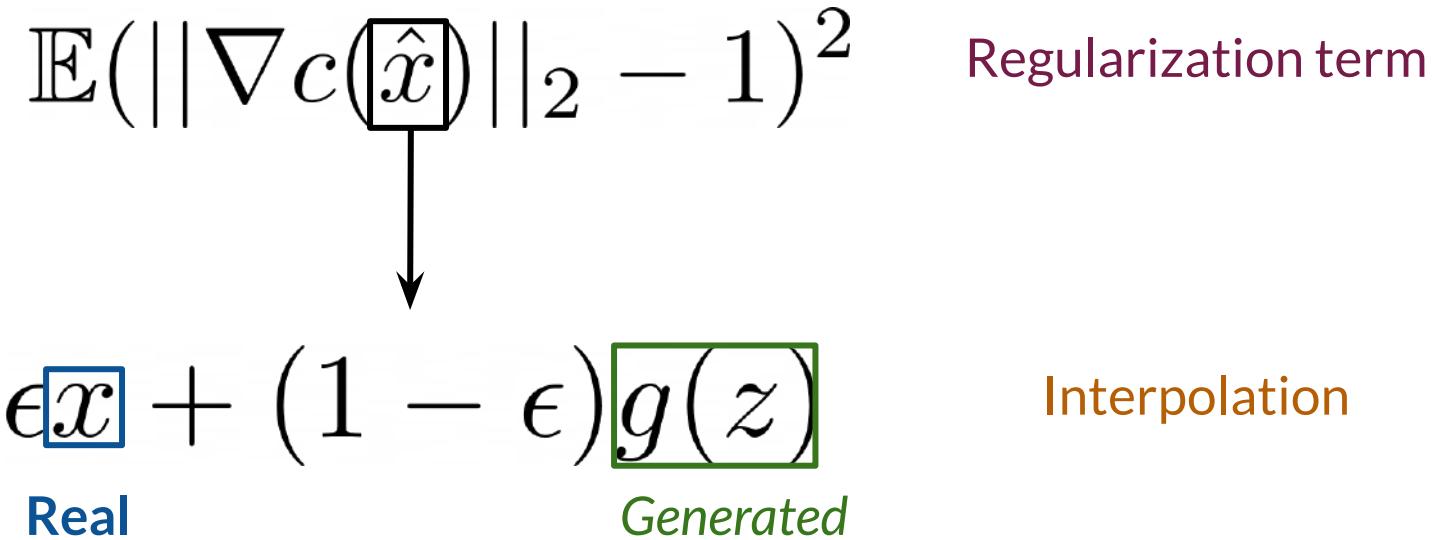
1-L Enforcement: Gradient Penalty

$$\mathbb{E}(\|\nabla c(\hat{x})\|_2 - 1)^2 \quad \text{Regularization term}$$

$$\epsilon \boxed{x} + (1 - \epsilon)g(z) \quad \text{Interpolation}$$

Real

1-L Enforcement: Gradient Penalty



Putting It All Together

$$\min_g \max_c \mathbb{E}(c(x)) - \mathbb{E}(c(g(z))) + \lambda \mathbb{E}(\|\nabla c(\hat{x})\|_2 - 1)^2$$

Putting It All Together

$$\min_g \max_c \boxed{\mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))} + \lambda \mathbb{E}(\|\nabla c(\hat{x})\|_2 - 1)^2$$

Makes the GAN less prone to mode collapse and vanishing gradient

Putting It All Together

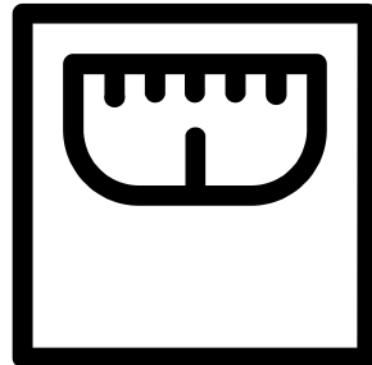
$$\min_g \max_c \boxed{\mathbb{E}(c(x)) - \mathbb{E}(c(g(z)))} + \lambda \mathbb{E}(\|\nabla c(\hat{x})\|_2 - 1)^2$$

Makes the GAN less prone to **mode collapse** and **vanishing gradient**

Tries to make the critic be 1-L Continuous, for the loss function to be **continuous and differentiable**

Summary

- Weight clipping and gradient penalty are ways to enforce 1-L continuity
- Gradient penalty tends to work better



Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>

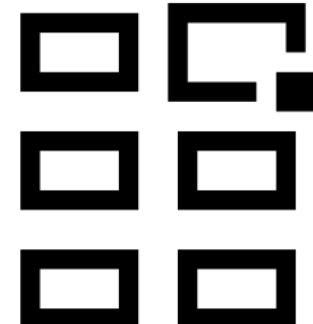


deeplearning.ai

Conditional Generation: Intuition

Outline

- Unconditional generation
- Conditional vs. unconditional generation



Unconditional Generation

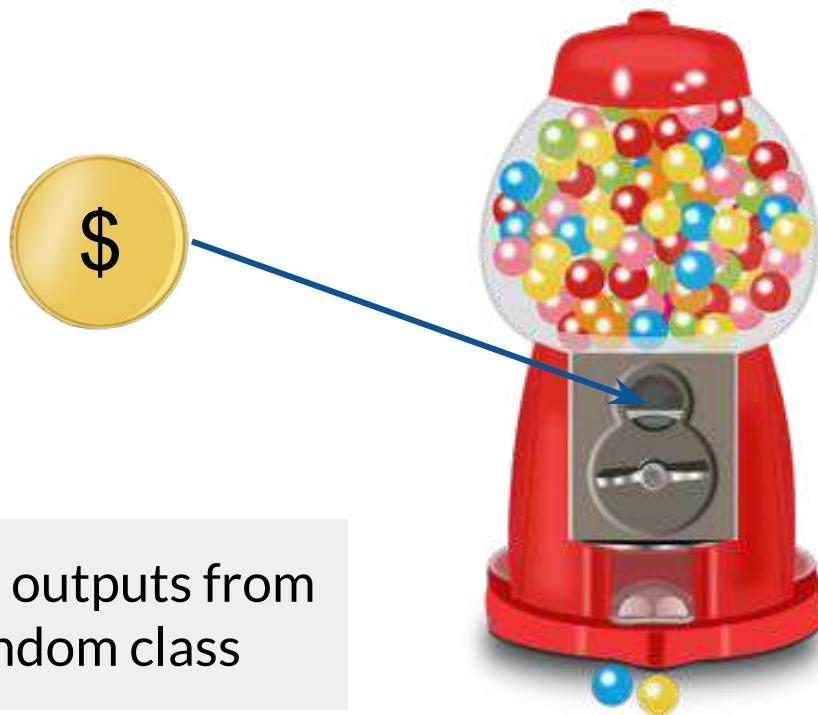
You get outputs from
a random class

Unconditional Generation



You get outputs from
a random class

Unconditional Generation



You get outputs from
a random class

Unconditional Generation



Unconditional Generation



You get outputs from
a random class

Unconditional Generation

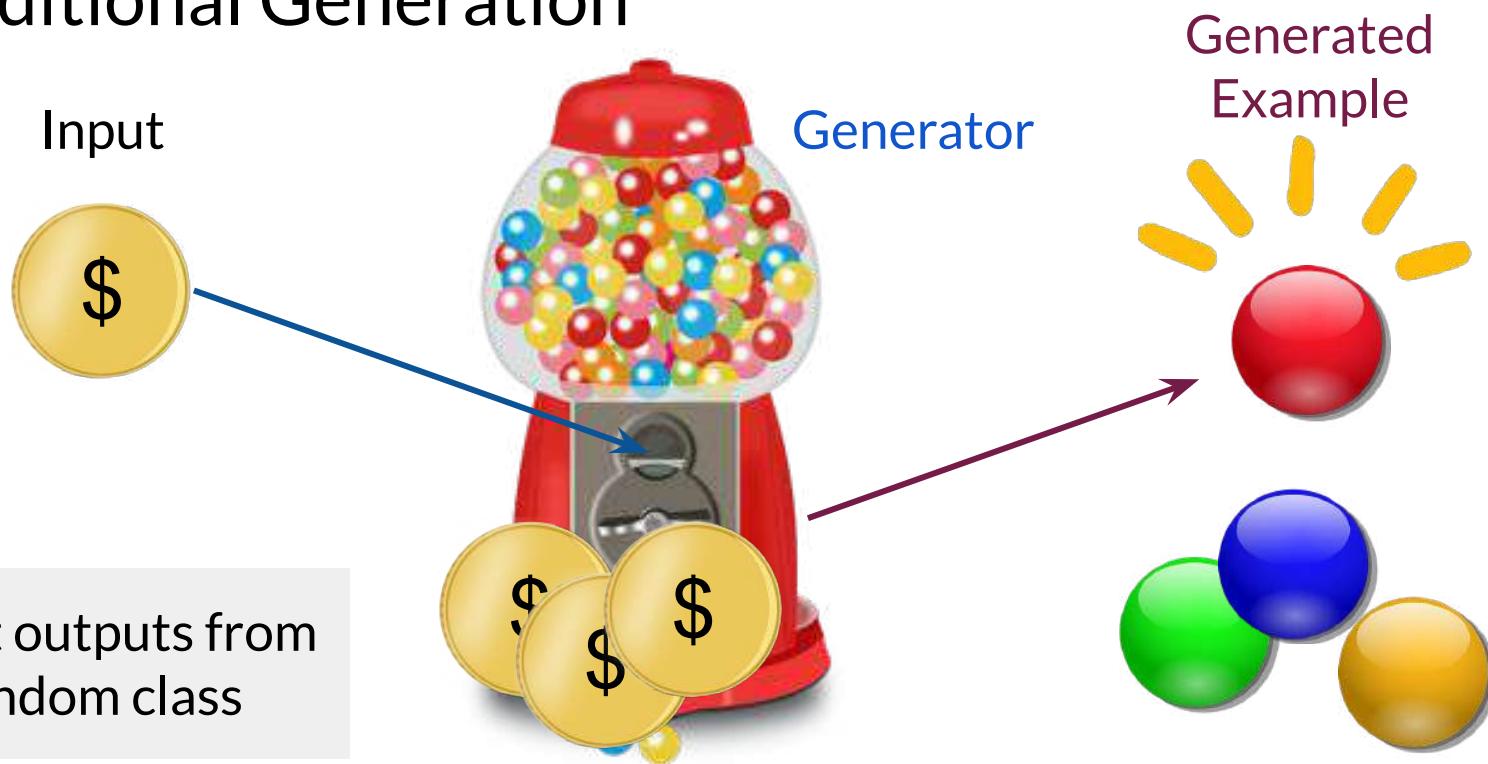


Unconditional Generation



You get outputs from
a random class

Unconditional Generation



Conditional Generation

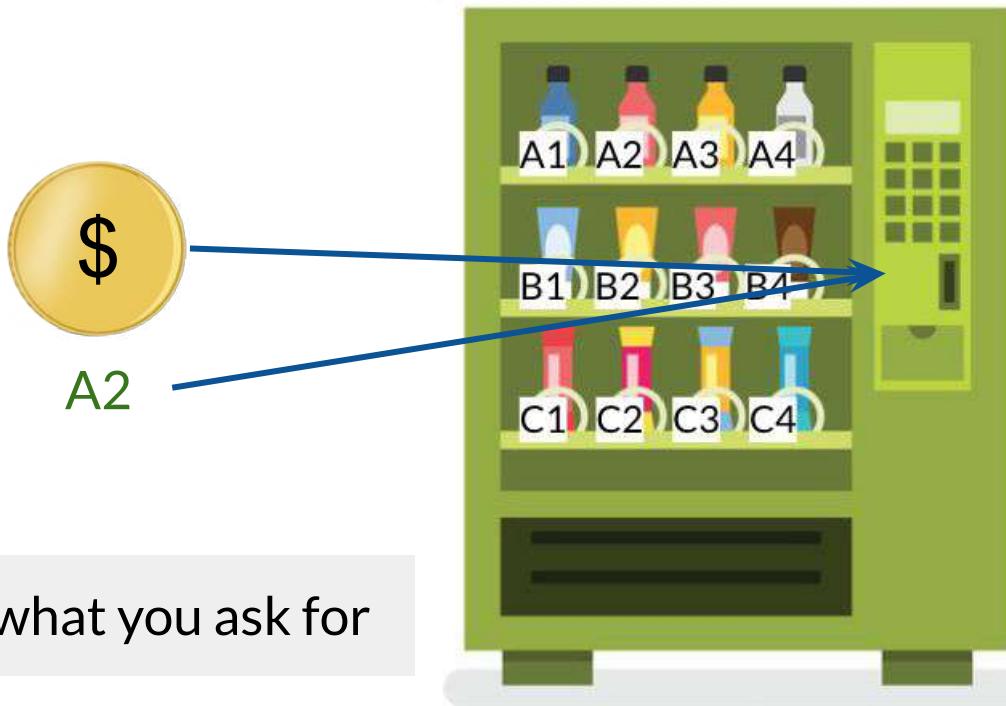
You get what you ask for

Conditional Generation



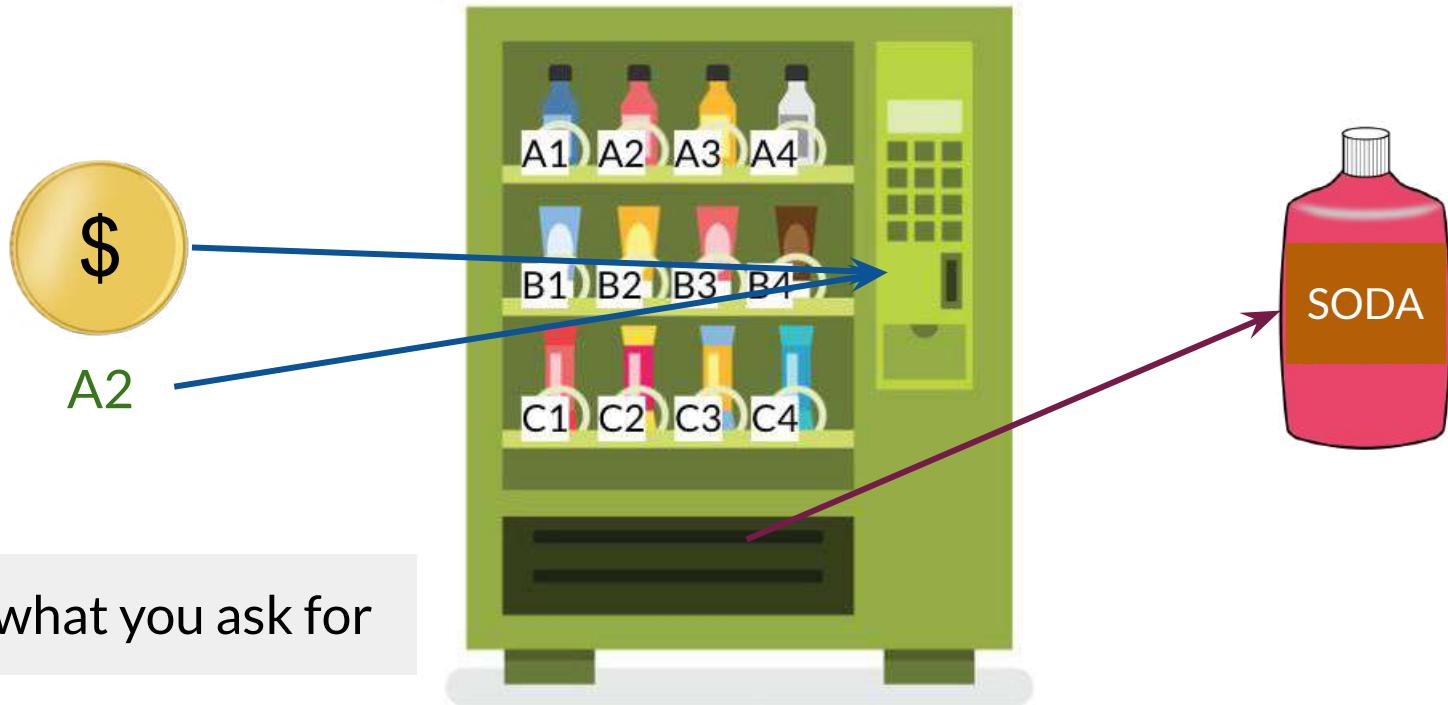
You get what you ask for

Conditional Generation



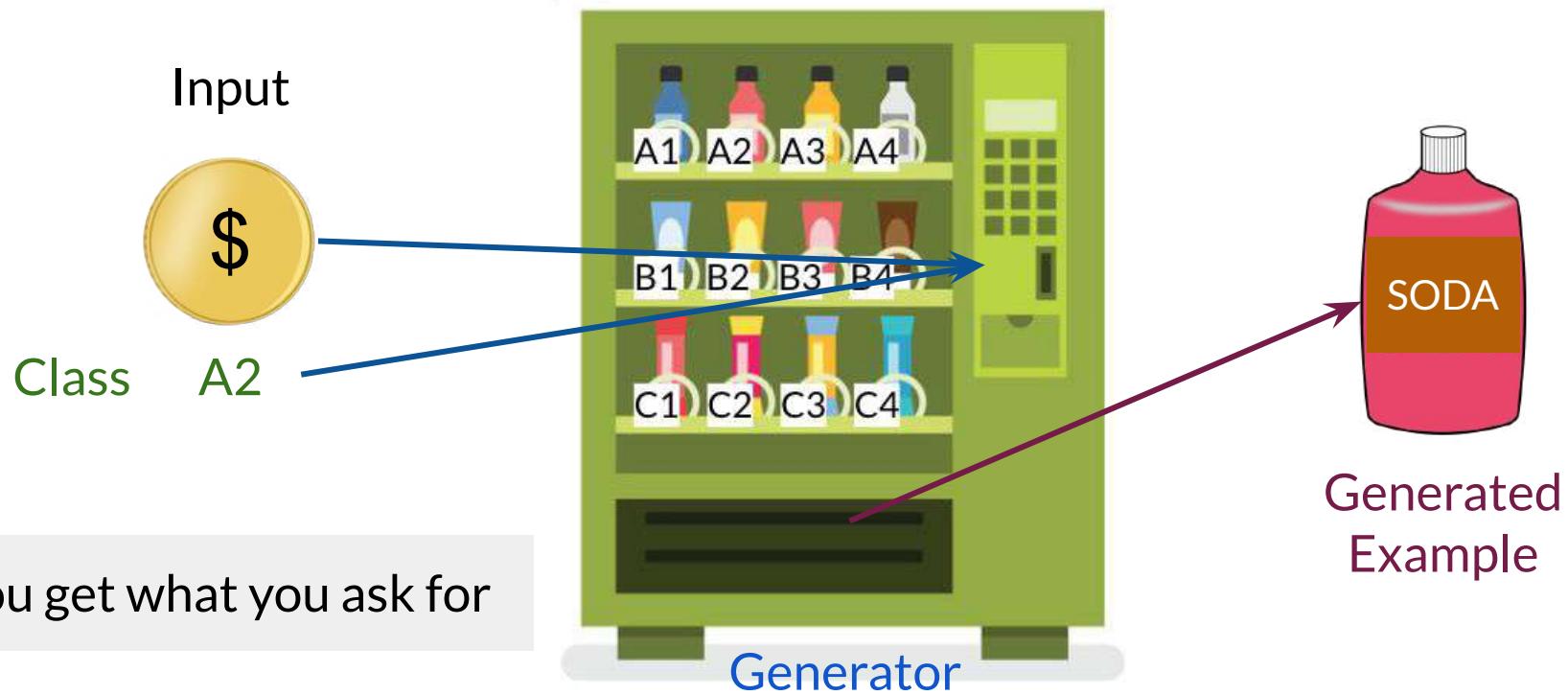
You get what you ask for

Conditional Generation



You get what you ask for

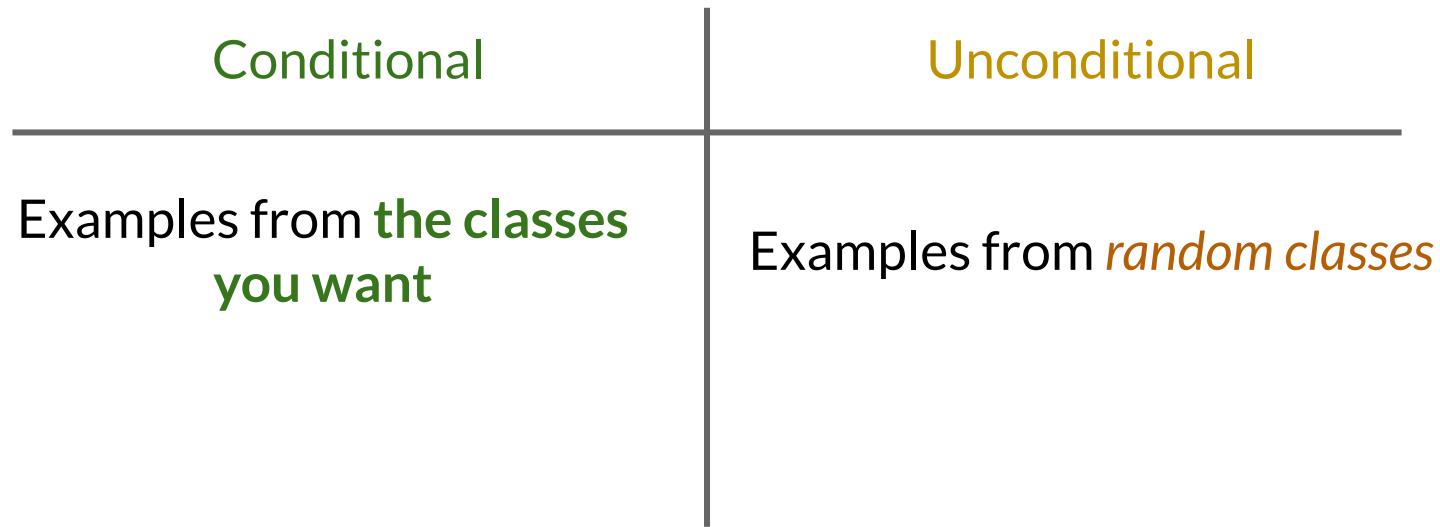
Conditional Generation



Conditional vs. Unconditional Generation



Conditional vs. Unconditional Generation

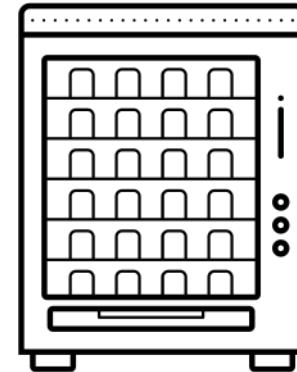


Conditional vs. Unconditional Generation

Conditional	Unconditional
Examples from the classes you want	Examples from random classes
Training dataset needs to be labeled	Training dataset doesn't need to be labeled

Summary

- Conditional generation requires labeled datasets
- Examples can be generated for the selected class



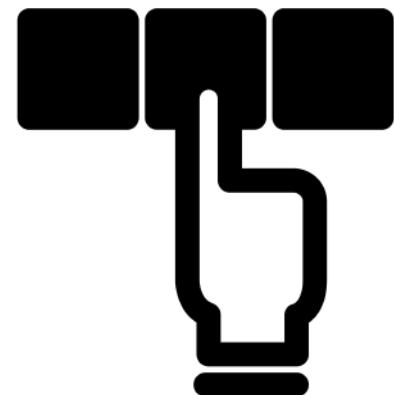


deeplearning.ai

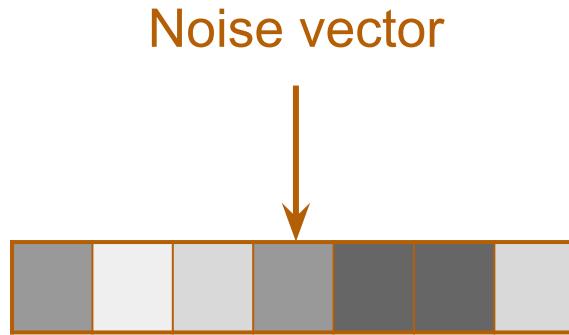
Conditional Generation: Inputs

Outline

- How to tell the generator what type of example to produce
- Input representation for the discriminator

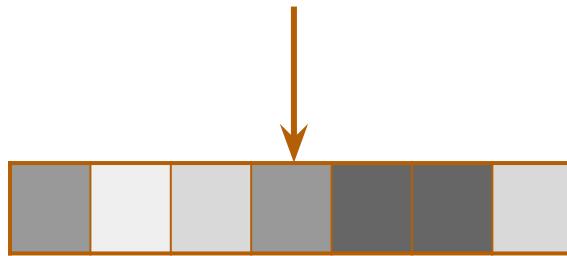


Generator Input

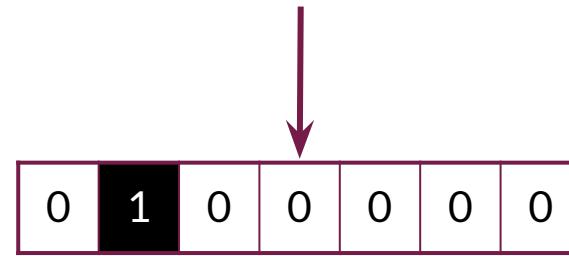


Generator Input

Noise vector

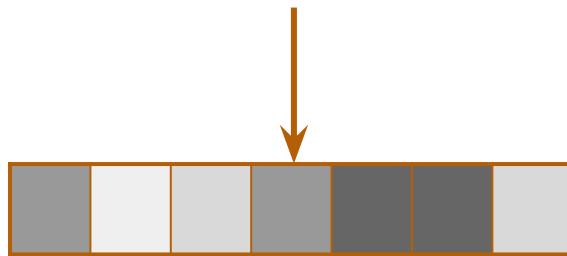


Class (one-hot) vector

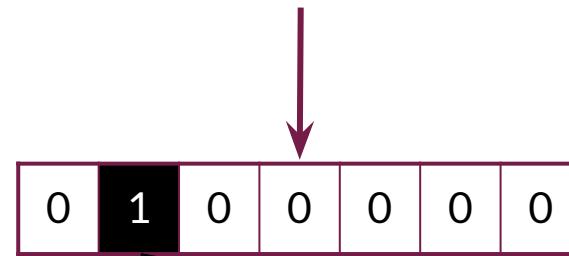


Generator Input

Noise vector



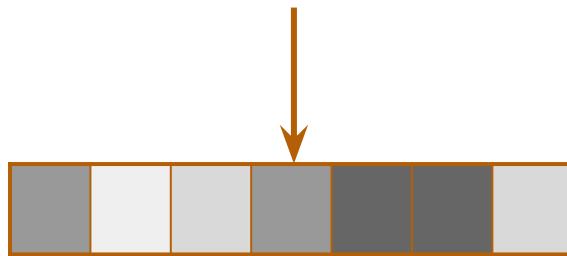
Class (one-hot) vector



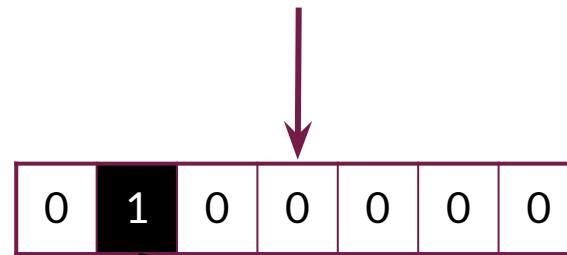
Husky

Generator Input

Noise vector



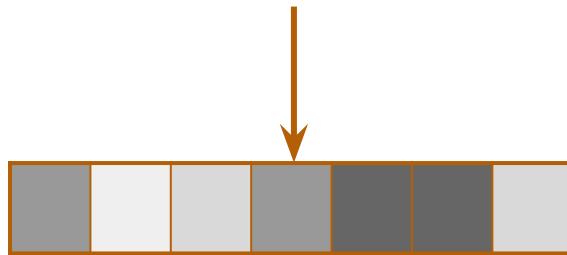
Class (one-hot) vector



Randomness in the
generation

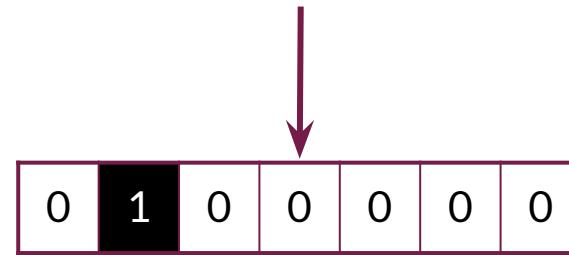
Generator Input

Noise vector



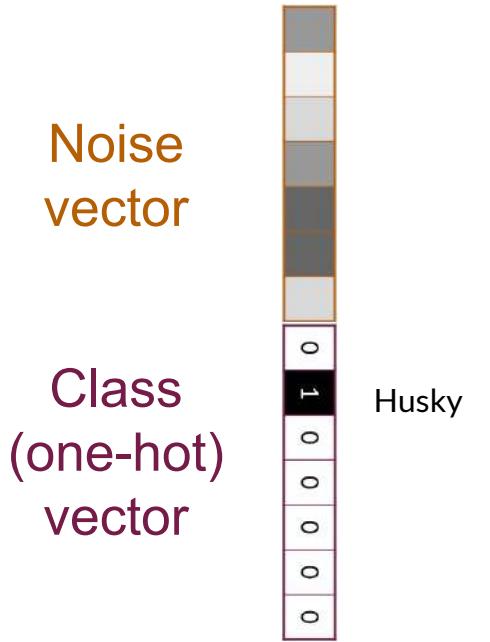
Randomness in the generation

Class (one-hot) vector

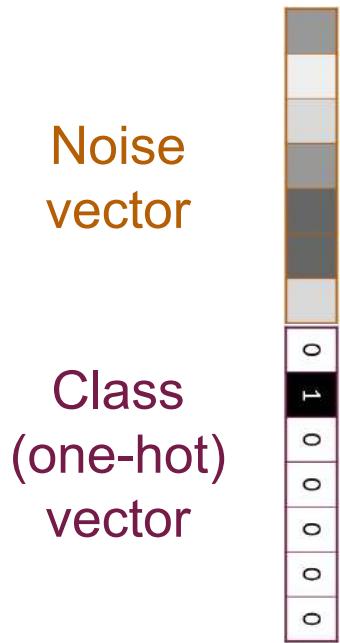


Control in the generation

Generator Input



Generator Input

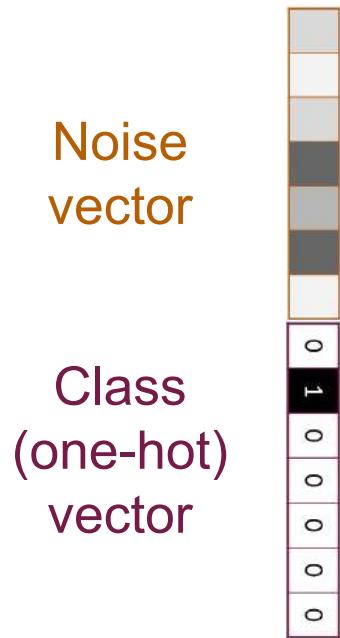


Generator

Output



Generator Input



Generator

Husky

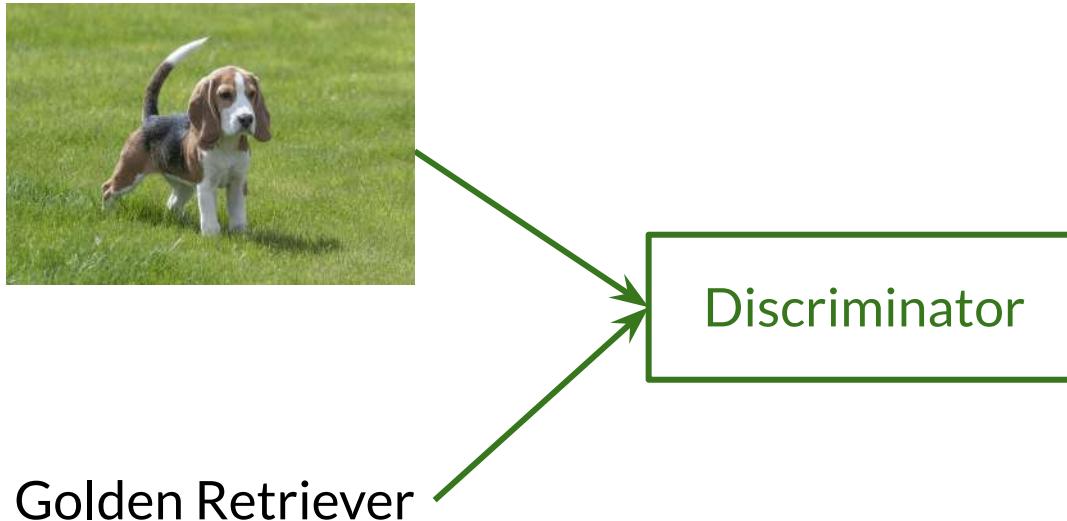
Output



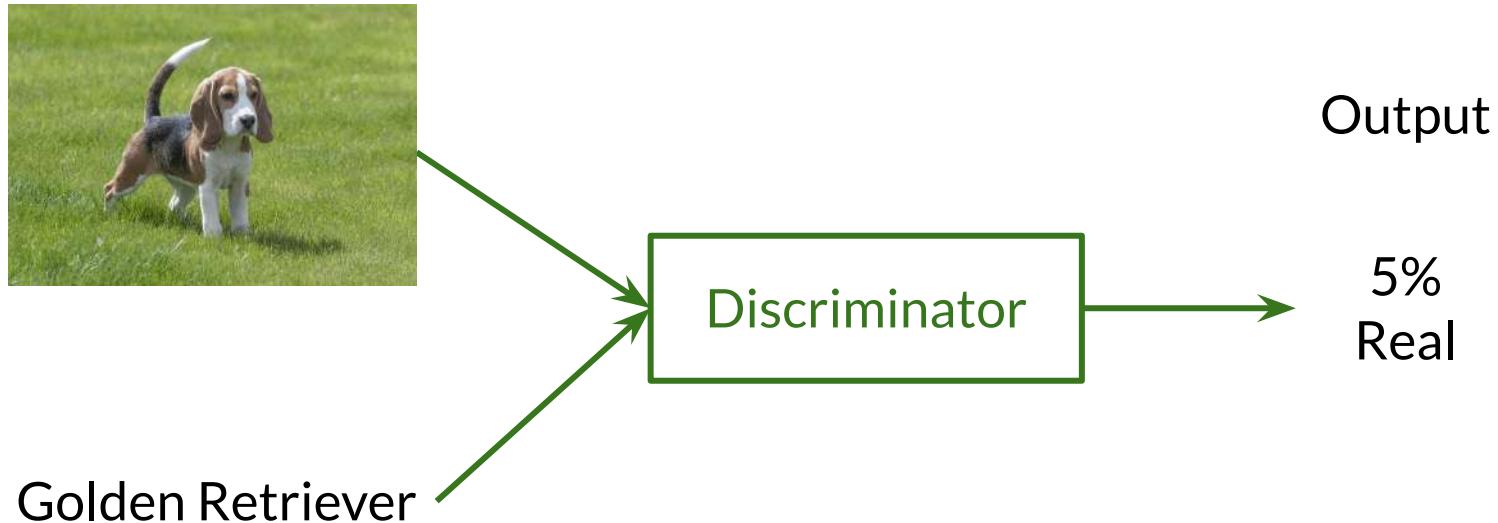
Discriminator Input

Discriminator

Discriminator Input



Discriminator Input



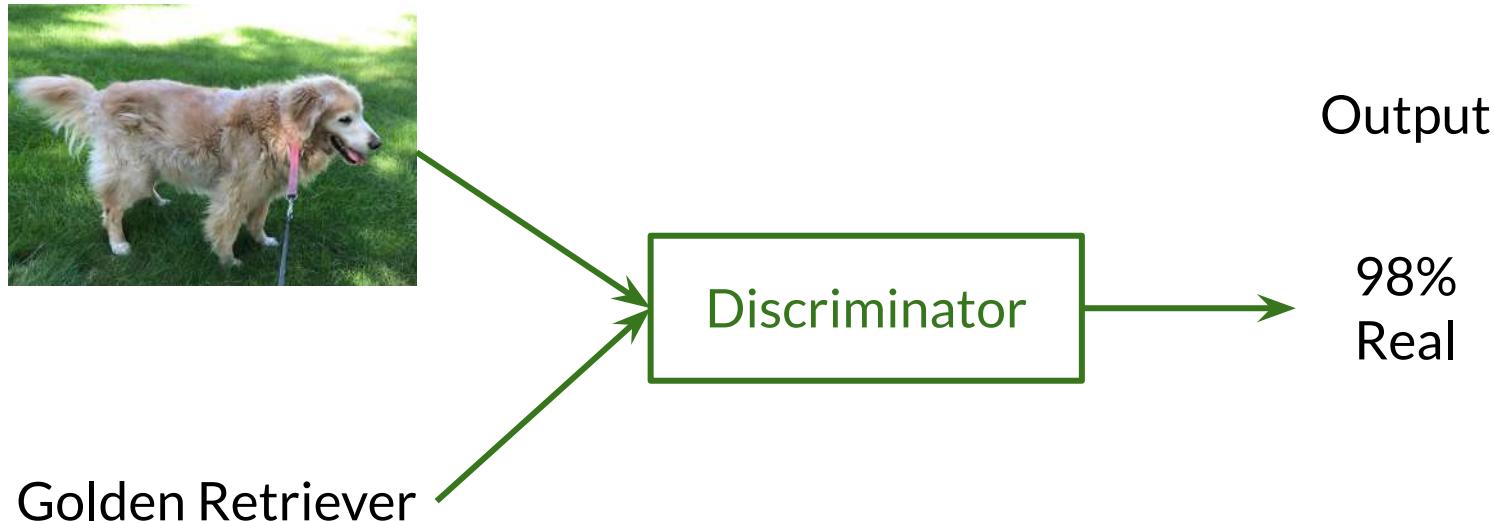
Discriminator Input



Golden Retriever

Discriminator

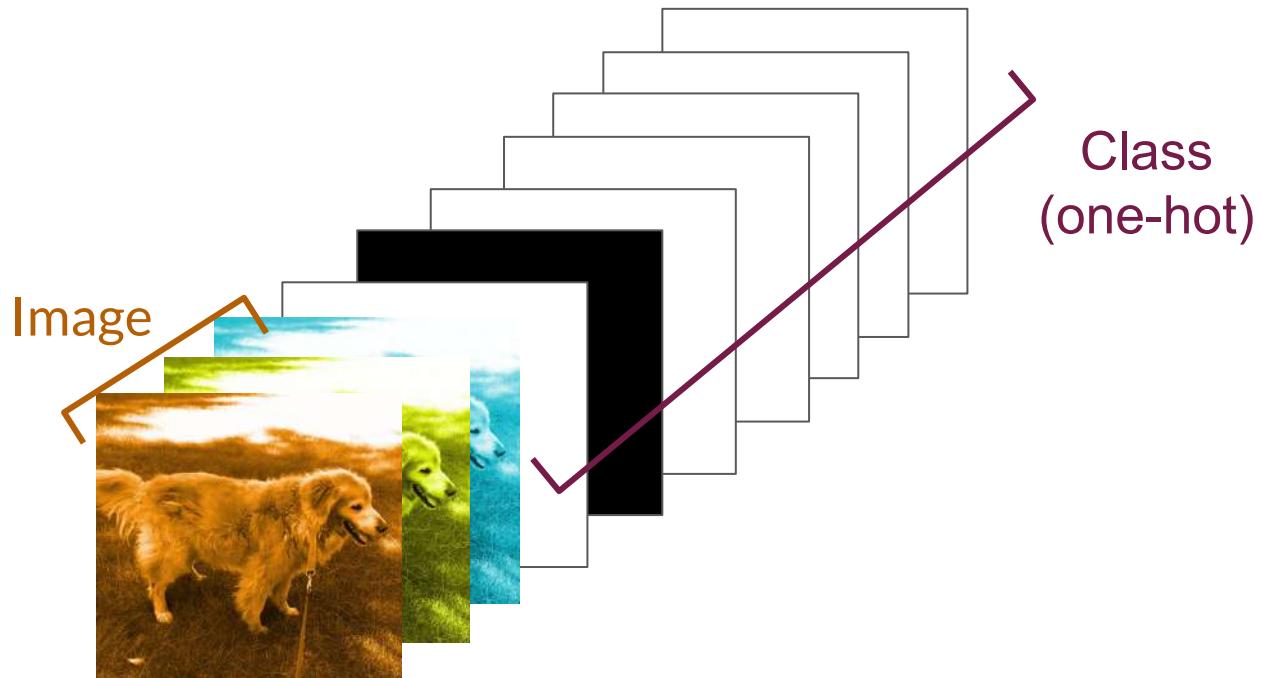
Discriminator Input



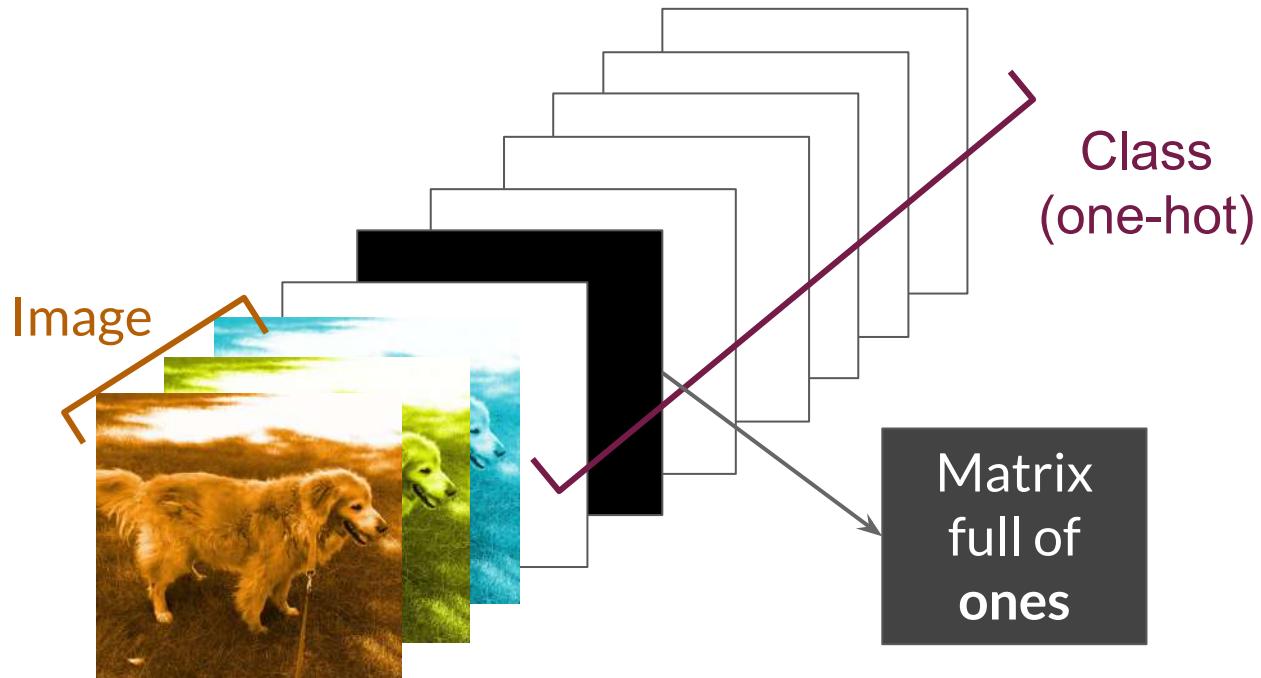
Discriminator Input



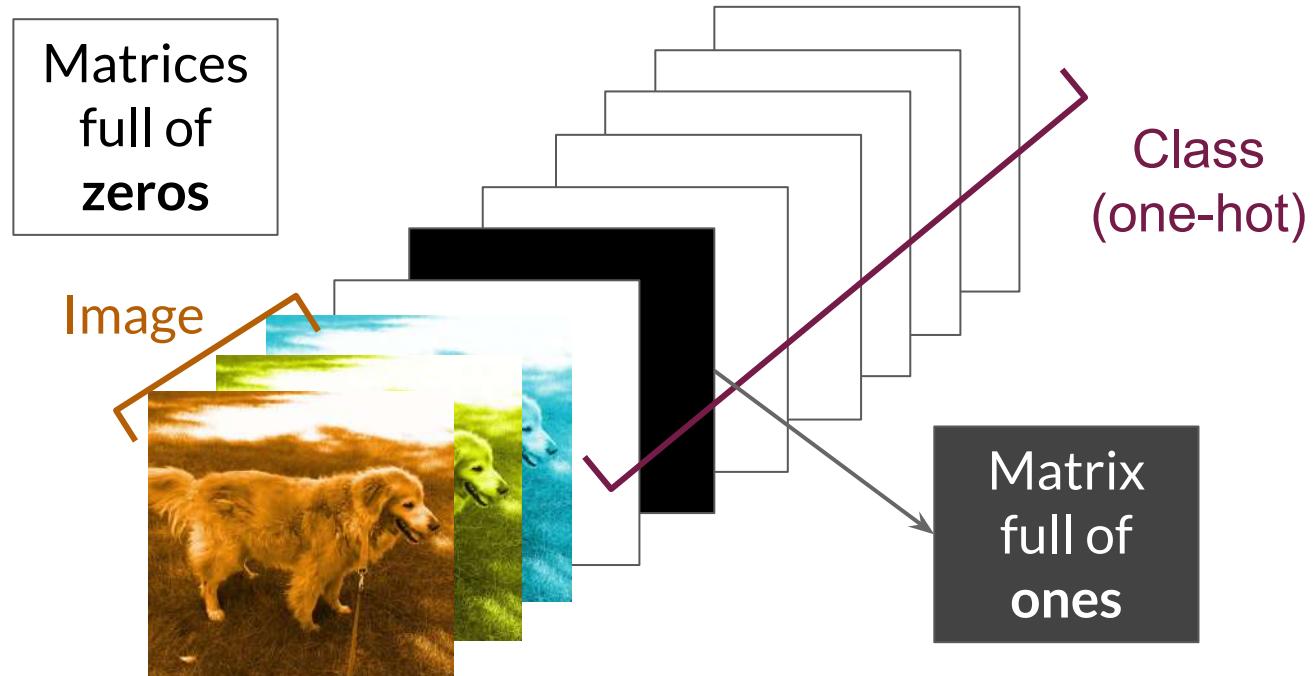
Discriminator Input



Discriminator Input

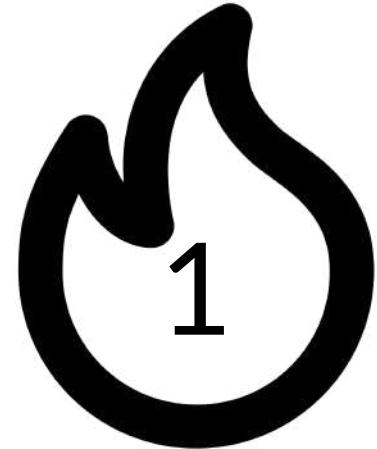


Discriminator Input



Summary

- The class is passed to the generator as one-hot vectors
- The class is passed to the discriminator as one-hot matrices
- The size of the vector and the number of matrices represent the number of classes



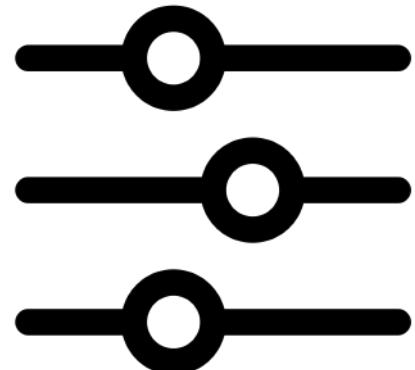


deeplearning.ai

Controllable Generation

Outline

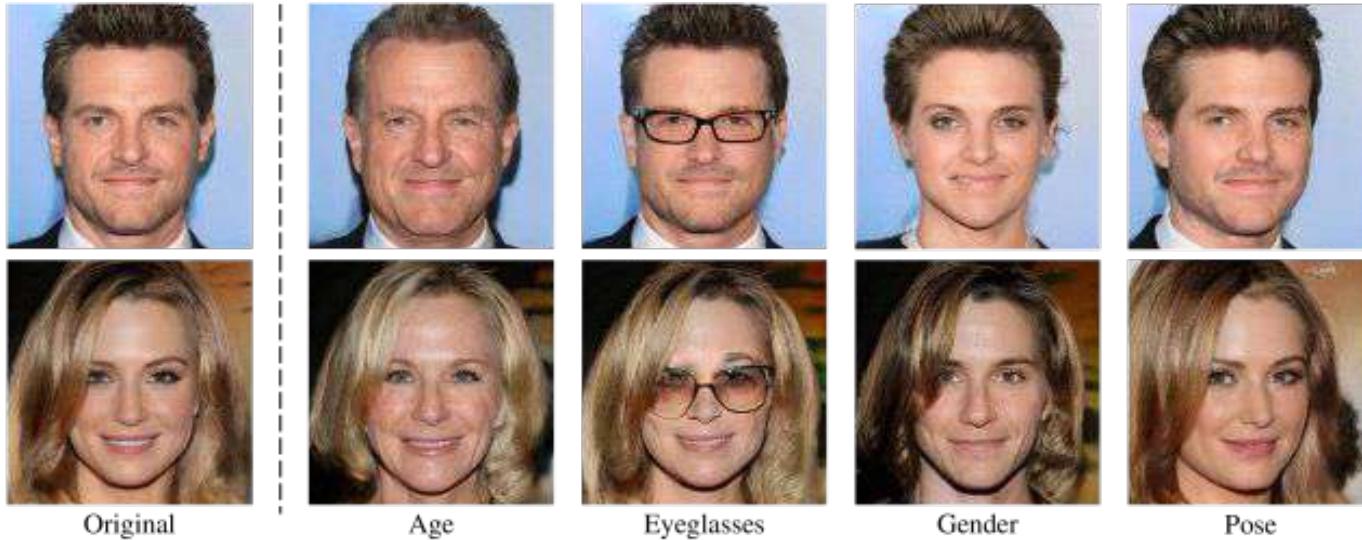
- What is controllable generation
- How it compares to conditional generation



Controllable Generation

Change specific features of the output

Controllable Generation



Change specific features of the output

Available from: <https://arxiv.org/abs/1907.10786>

Controllable Generation

Noise vector

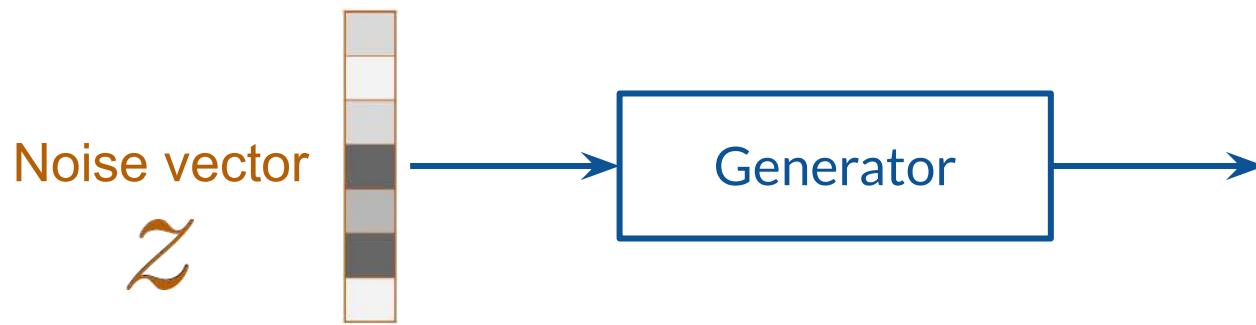
\tilde{z}



Tweak the input noise vector to get different features on the output

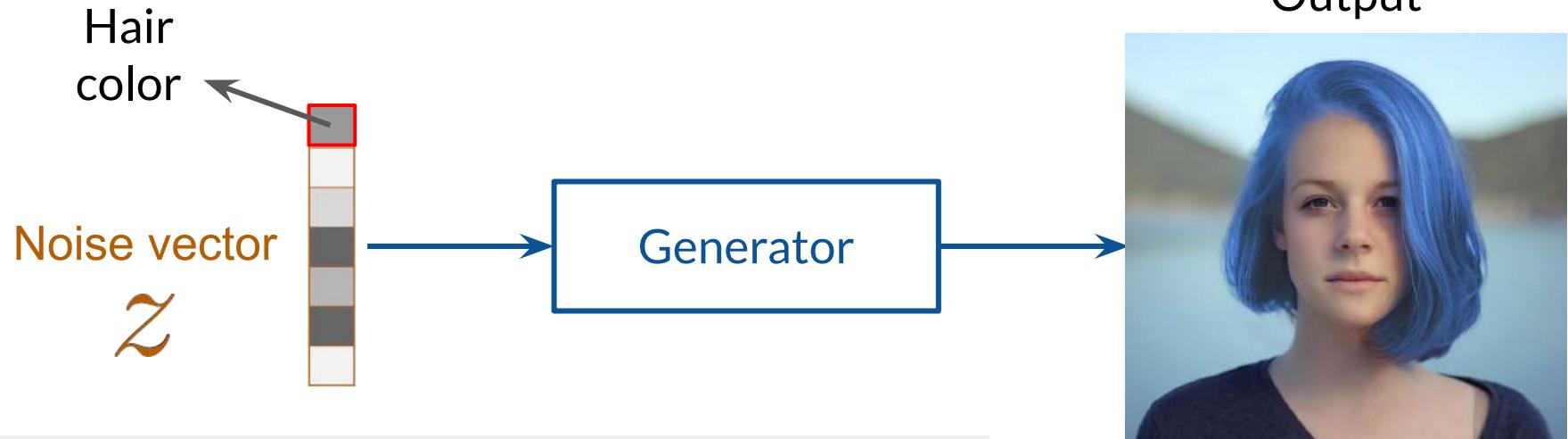
Controllable Generation

Controlled
Output



Tweak the input noise vector to get different features on the output

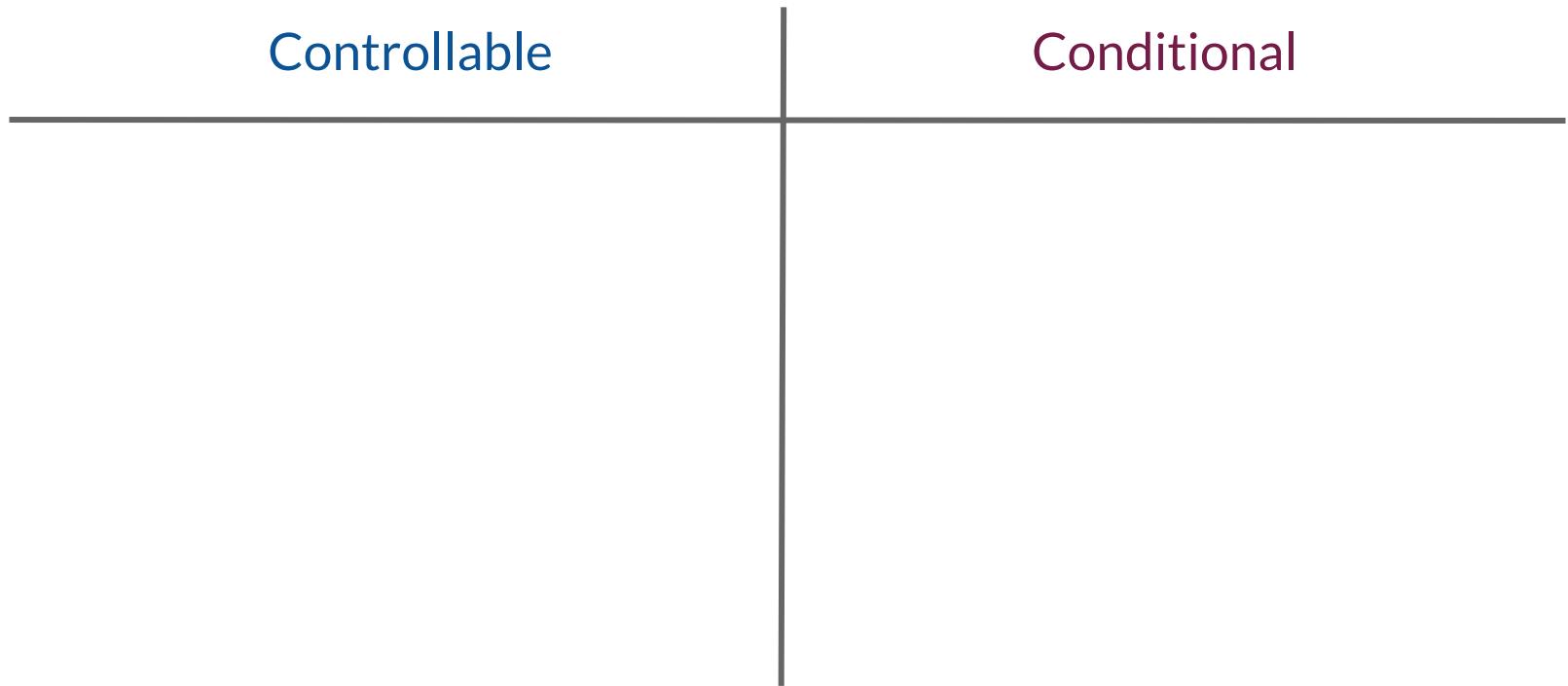
Controllable Generation



Tweak the input noise vector to get different features on the output

Controlled Output

Controllable Generation vs. Conditional Generation



Controllable Generation vs. Conditional Generation

Controllable	Conditional
Examples with the features that you want	Examples from <i>the classes you want</i>

Controllable Generation vs. Conditional Generation

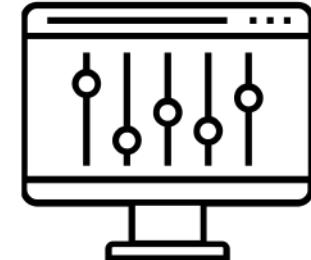
Controllable	Conditional
Examples with the features that you want	Examples from <i>the classes you want</i>
Training dataset doesn't need to be labeled	Training dataset <i>needs to be labeled</i>

Controllable Generation vs. Conditional Generation

Controllable	Conditional
Examples with the features that you want	Examples from <i>the classes you want</i>
Training dataset doesn't need to be labeled	Training dataset <i>needs to be labeled</i>
Manipulate the z vector input	<i>Append a class vector</i> to the input

Summary

- Controllable generation lets you control the features of the generated outputs
- It does not need a labeled training dataset
- The input vector is tweaked to get different features on the output



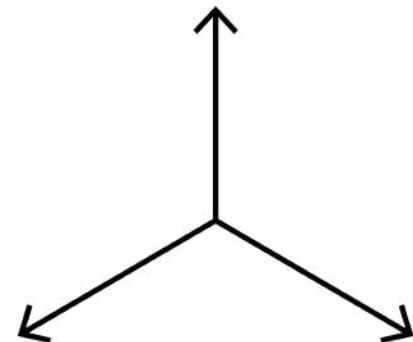


deeplearning.ai

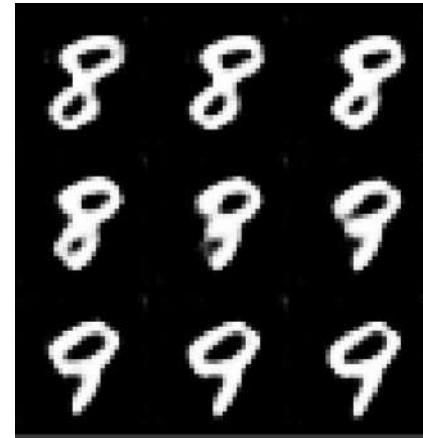
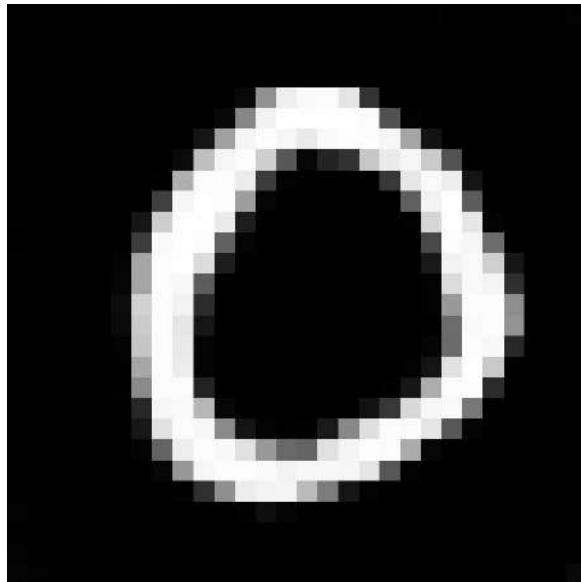
Vector Algebra in the Z-Space

Outline

- Interpolation in the Z-space
- Modifying the noise vector z to control desired features

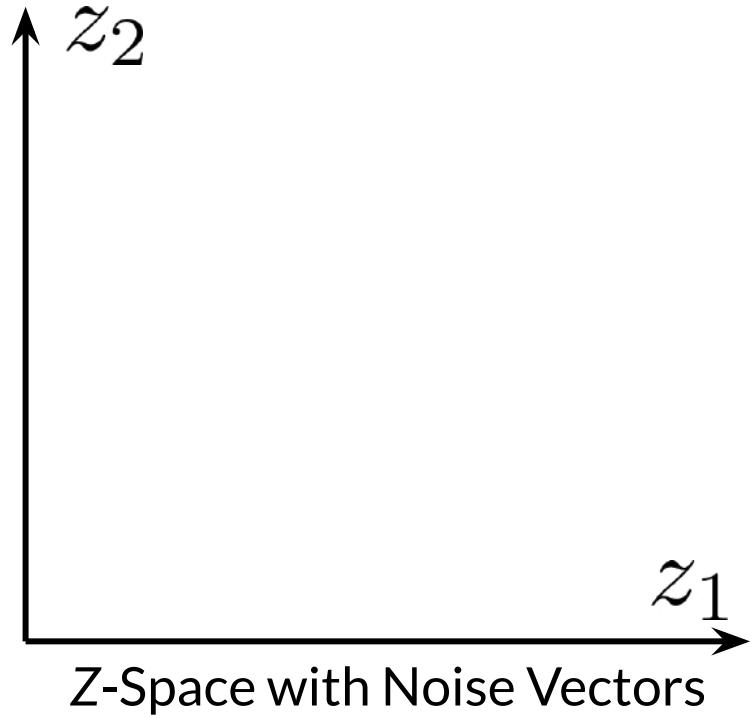


Interpolation Using the Z-Space

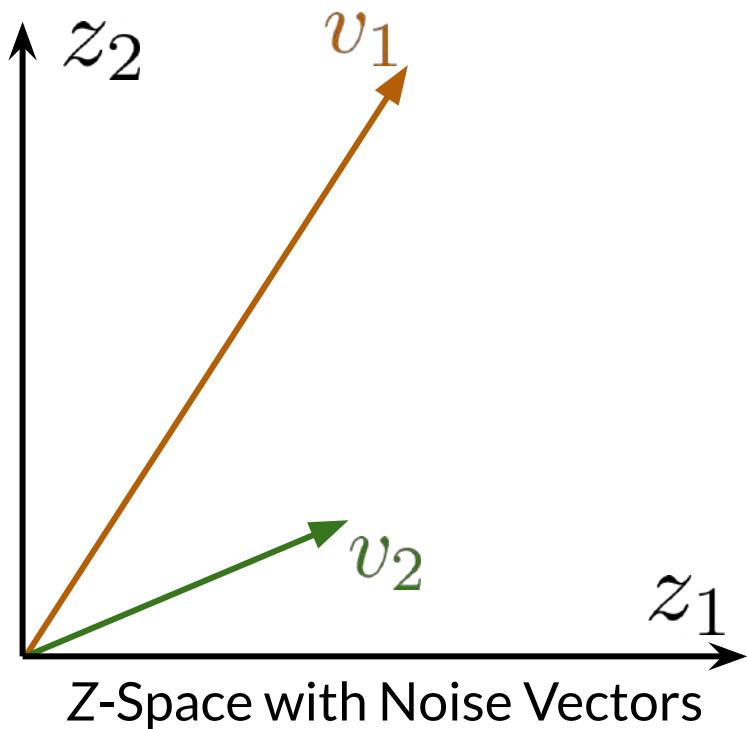


How an image morphs into another

Interpolation Using the Z-Space



Interpolation Using the Z-Space



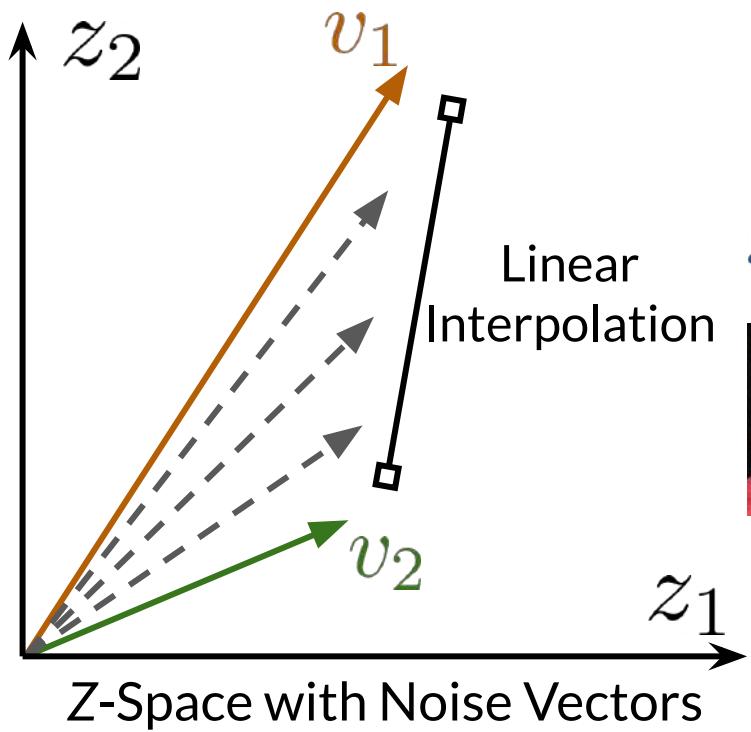
$g(v_1)$



$g(v_2)$



Interpolation Using the Z-Space



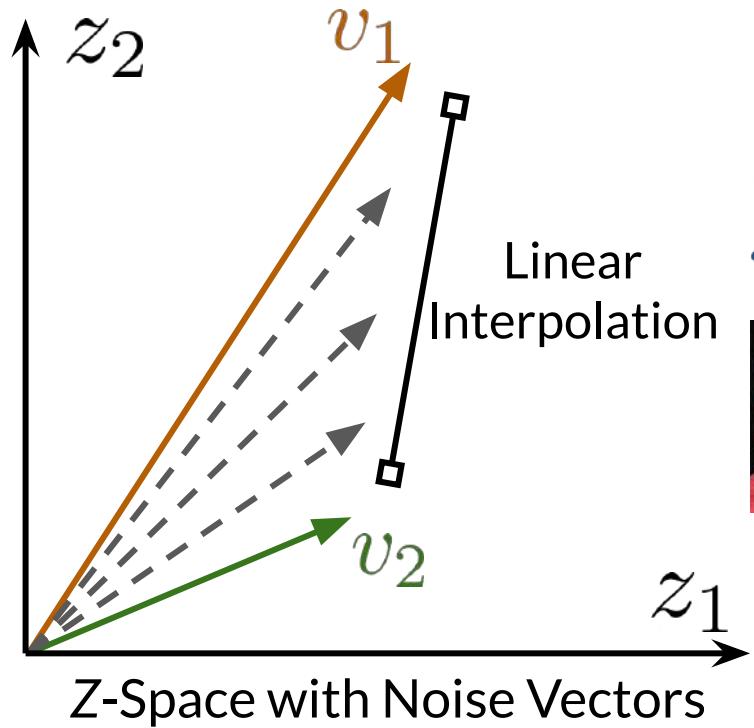
$g(v_1)$



$g(v_2)$



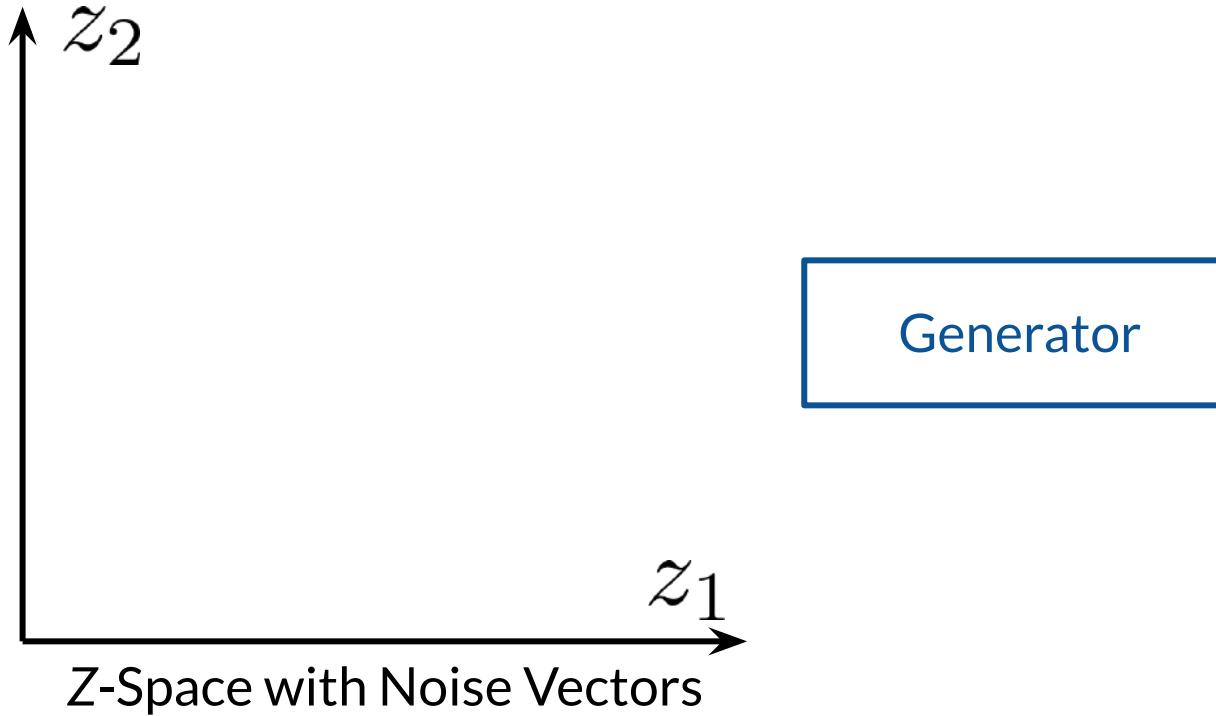
Interpolation Using the Z-Space



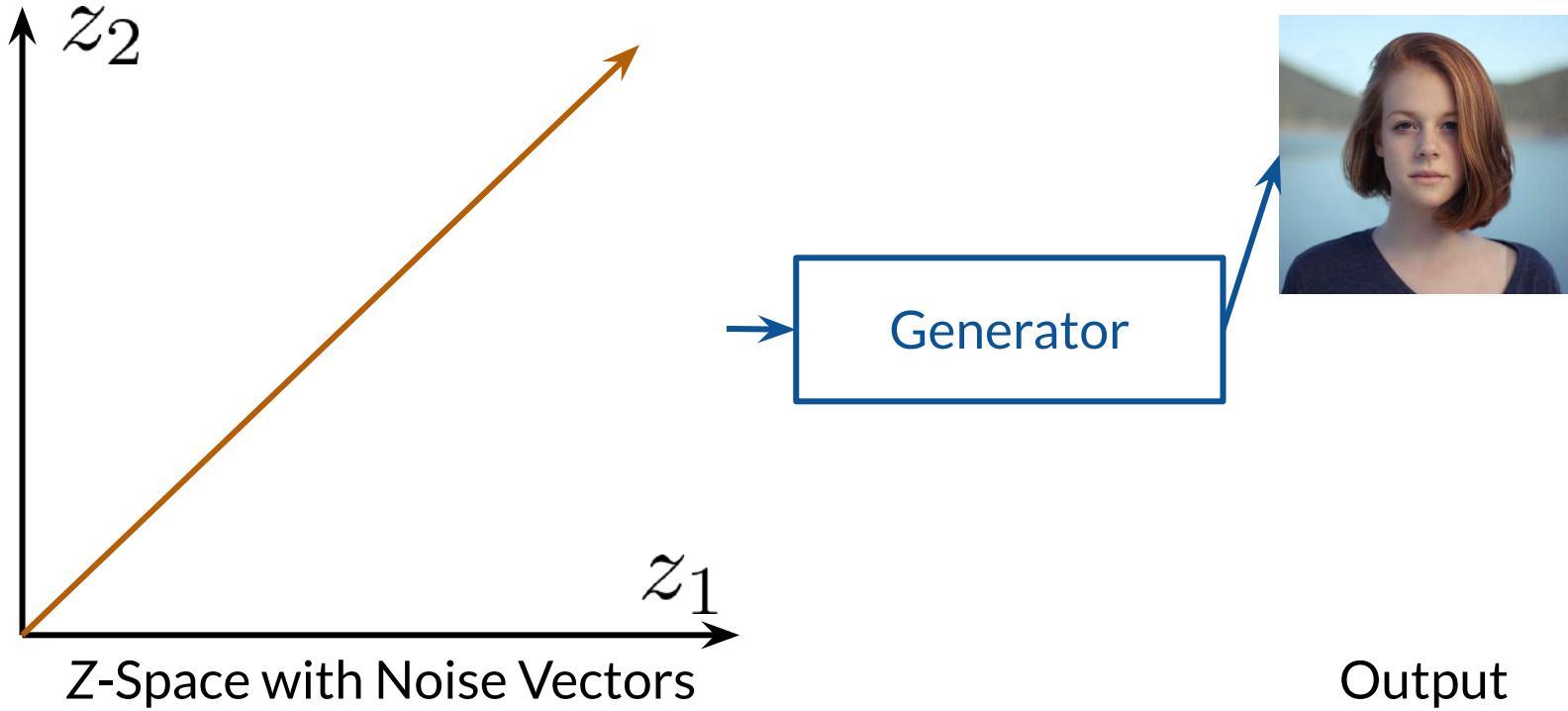
Intermediate images using
 $g(v_1)$ ← → $g(v_2)$



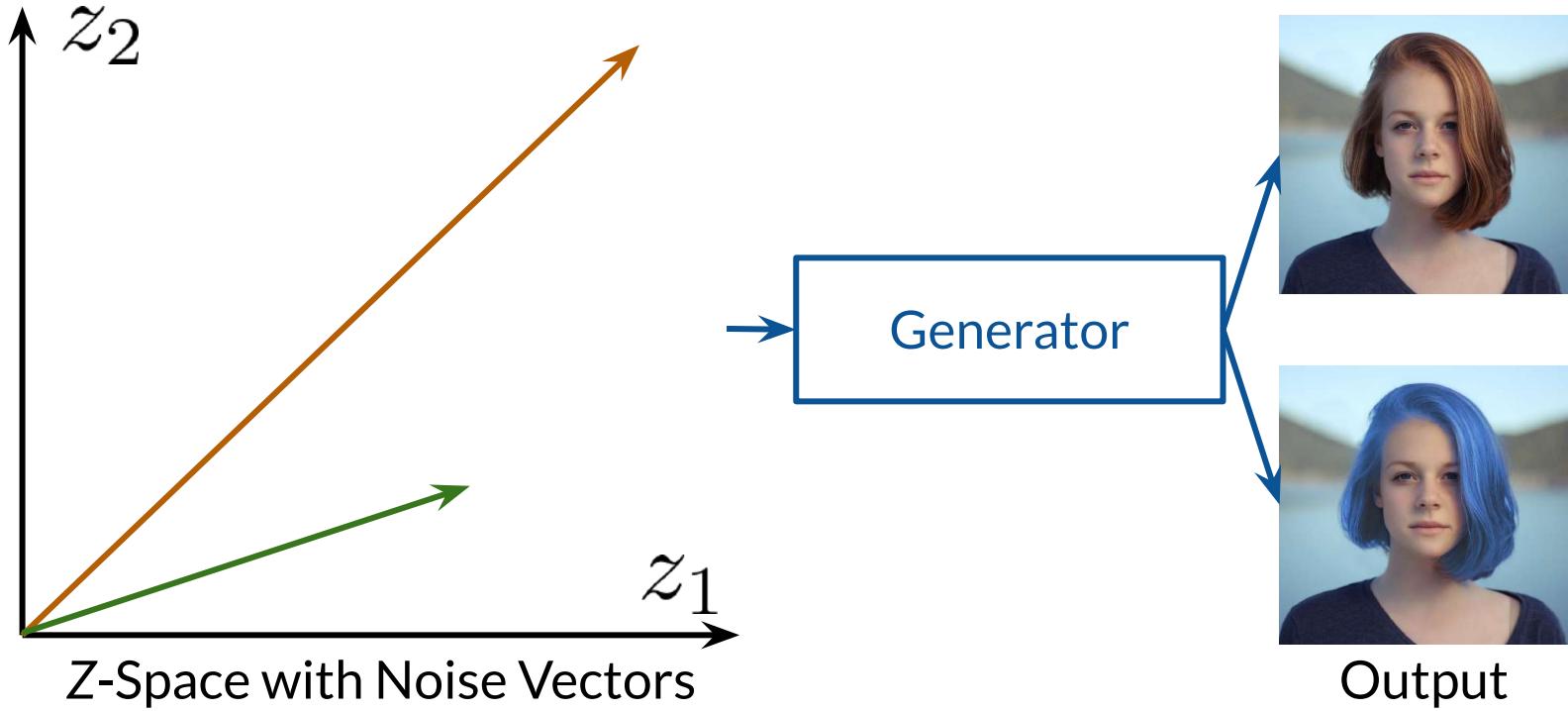
Z-Space and Controllable Generation



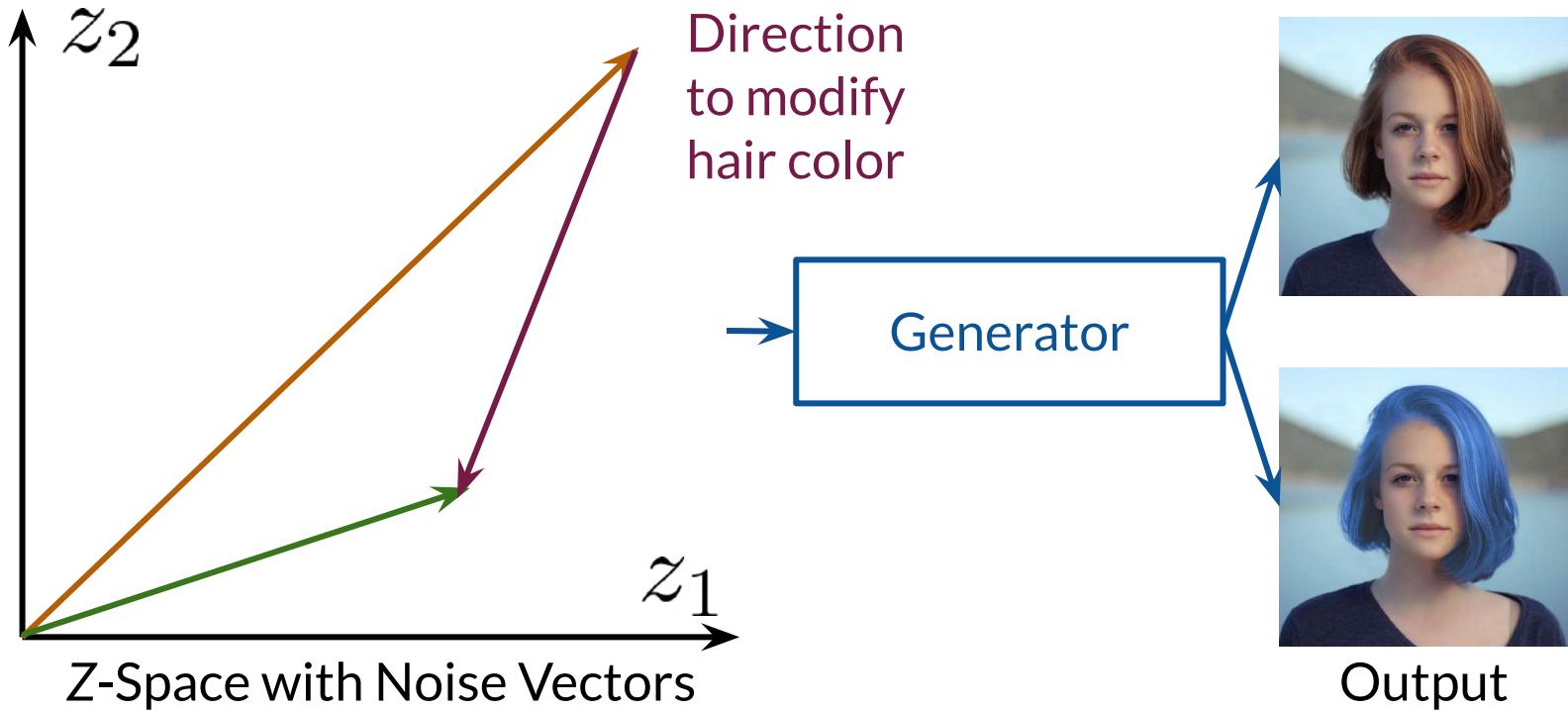
Z-Space and Controllable Generation



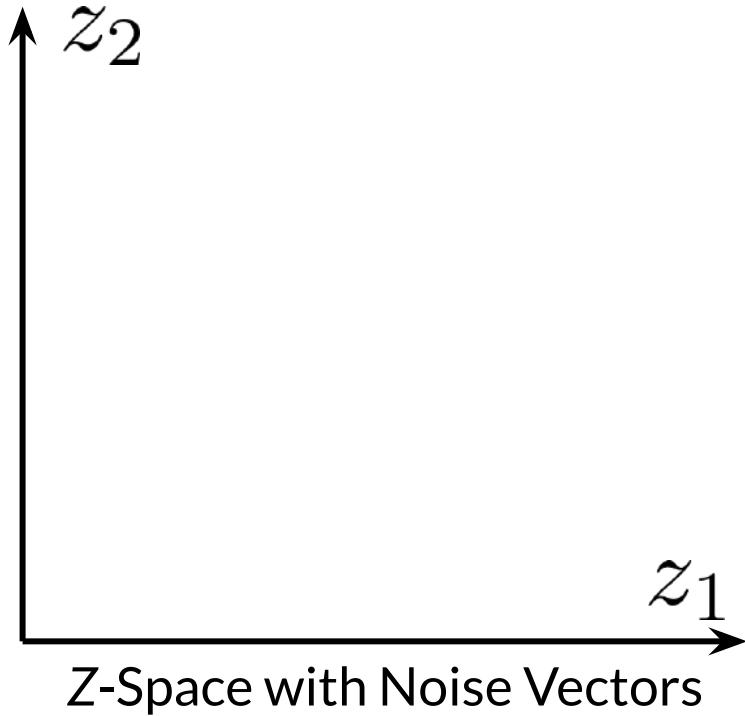
Z-Space and Controllable Generation



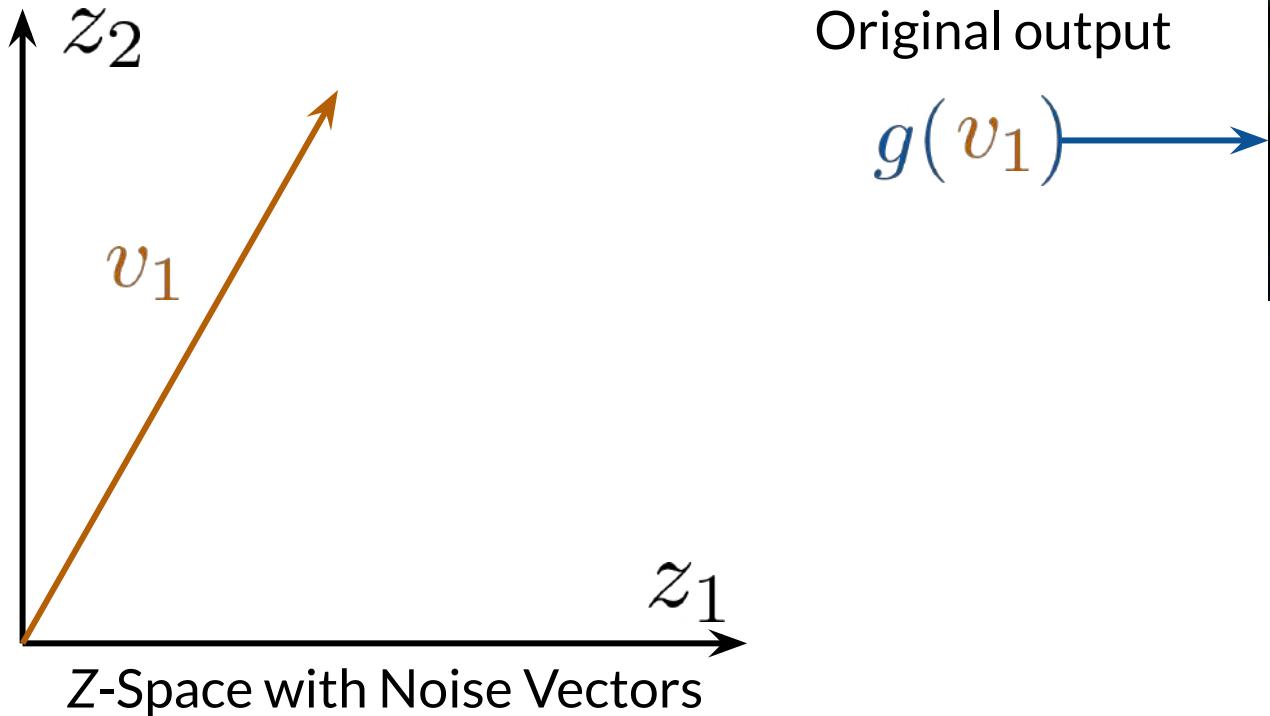
Z-Space and Controllable Generation



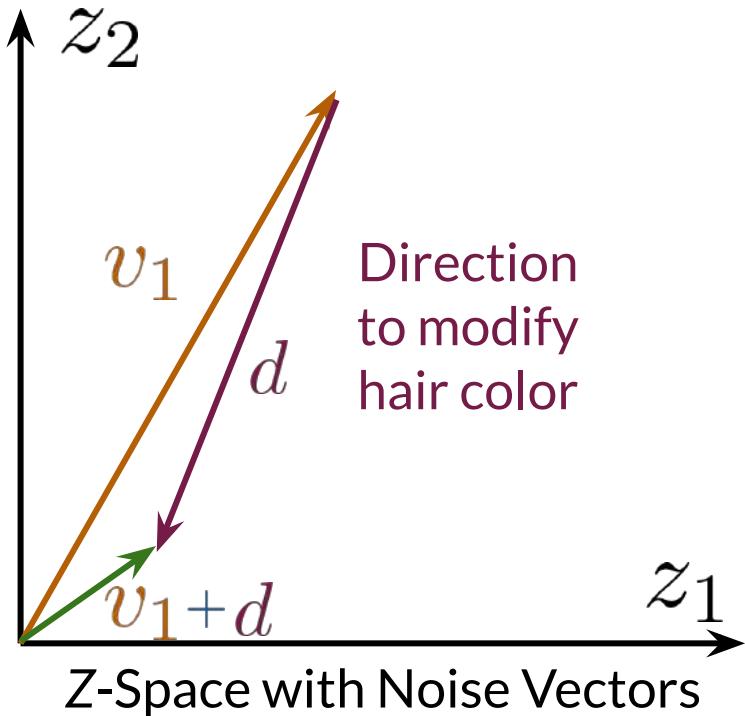
Z-Space and Controllable Generation



Z-Space and Controllable Generation



Z-Space and Controllable Generation

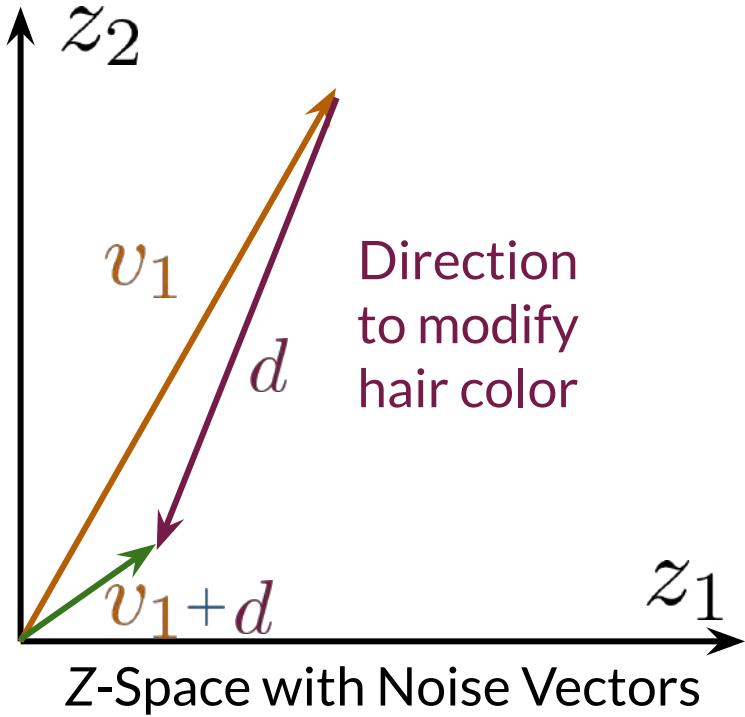


Original output

$$g(v_1) \longrightarrow$$



Z-Space and Controllable Generation



Original output

$$g(v_1) \rightarrow$$



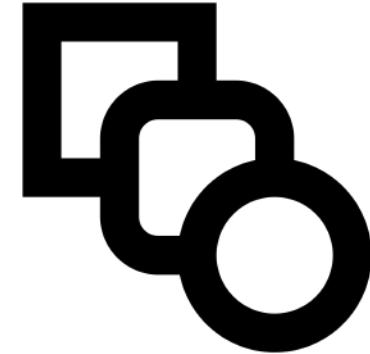
Controlled output

$$g(v_1 + d) \rightarrow$$



Summary

- To control output features, you need to find directions in the Z-space
- To modify your output, you move around in the Z-space





deeplearning.ai

Challenges with Controllable Generation

Outline

- Output feature correlation
- Z-space entanglement



Feature Correlation

Uncorrelated
Features



Add beard



Feature Correlation

Uncorrelated
Features



Add beard

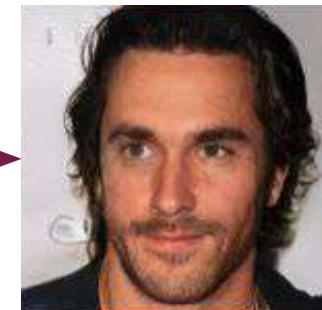


Correlated
Features



Add beard

Make more
masculine



Z-Space Entanglement

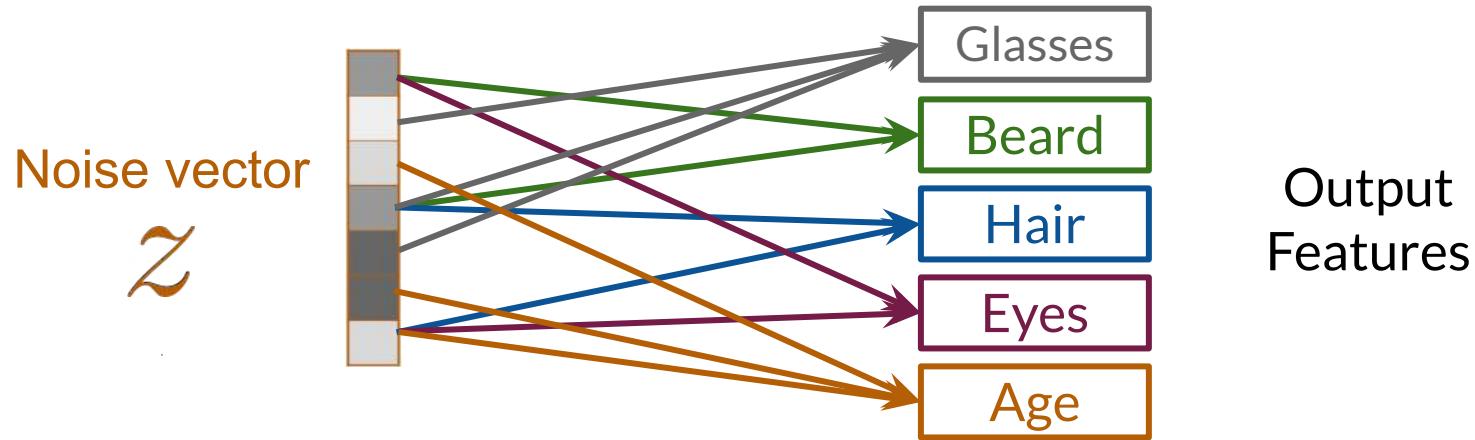
Noise vector

\tilde{z}

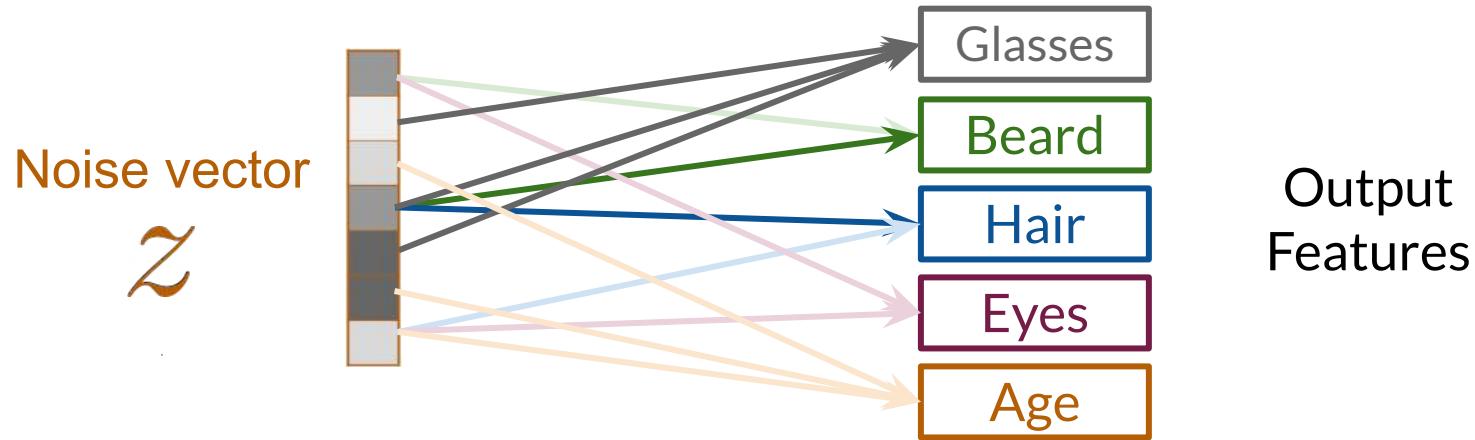


Output
Features

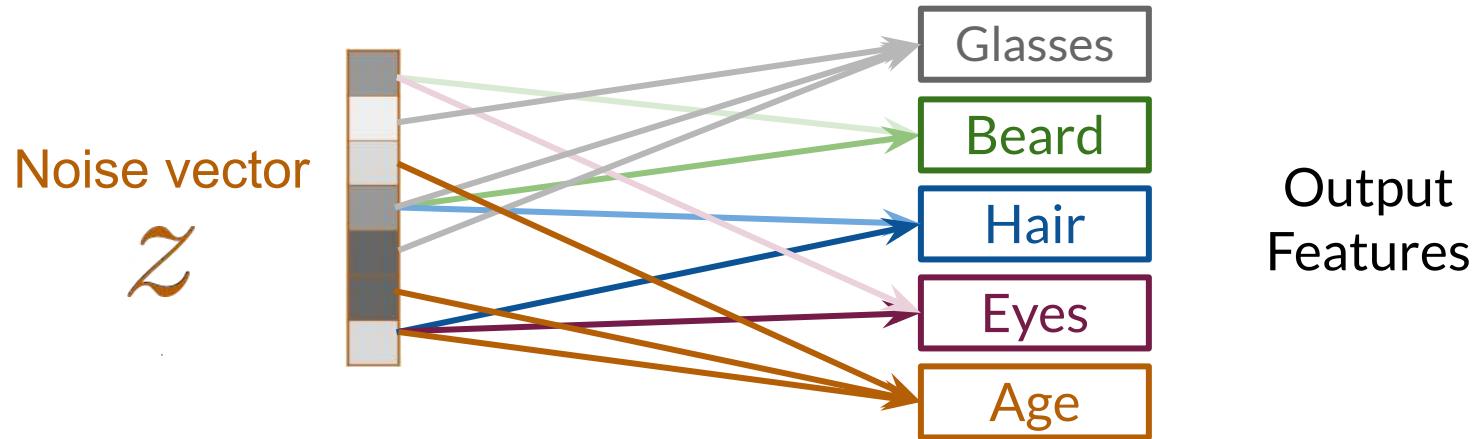
Z-Space Entanglement



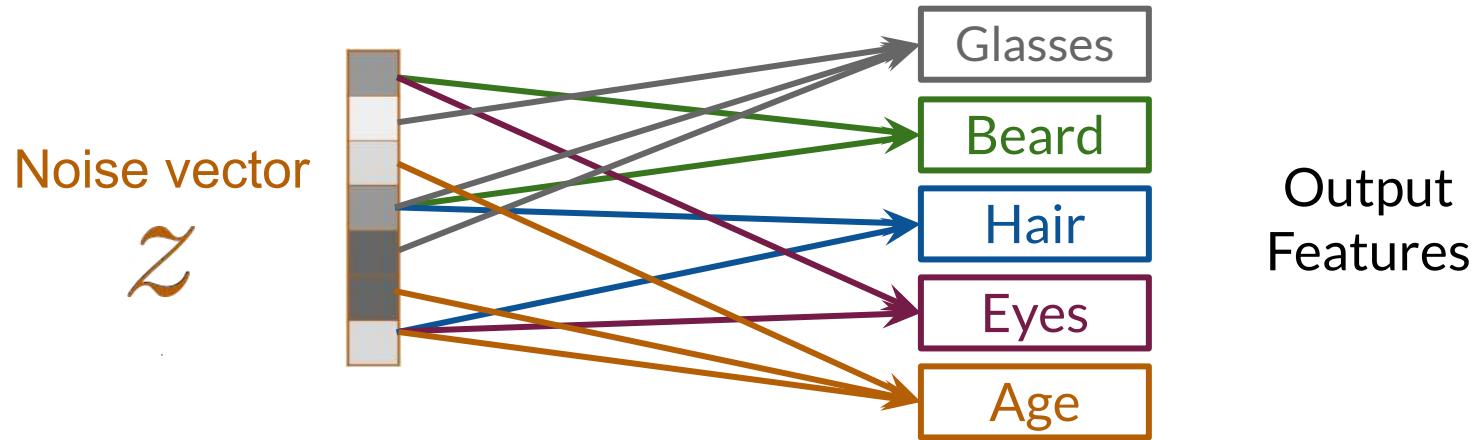
Z-Space Entanglement



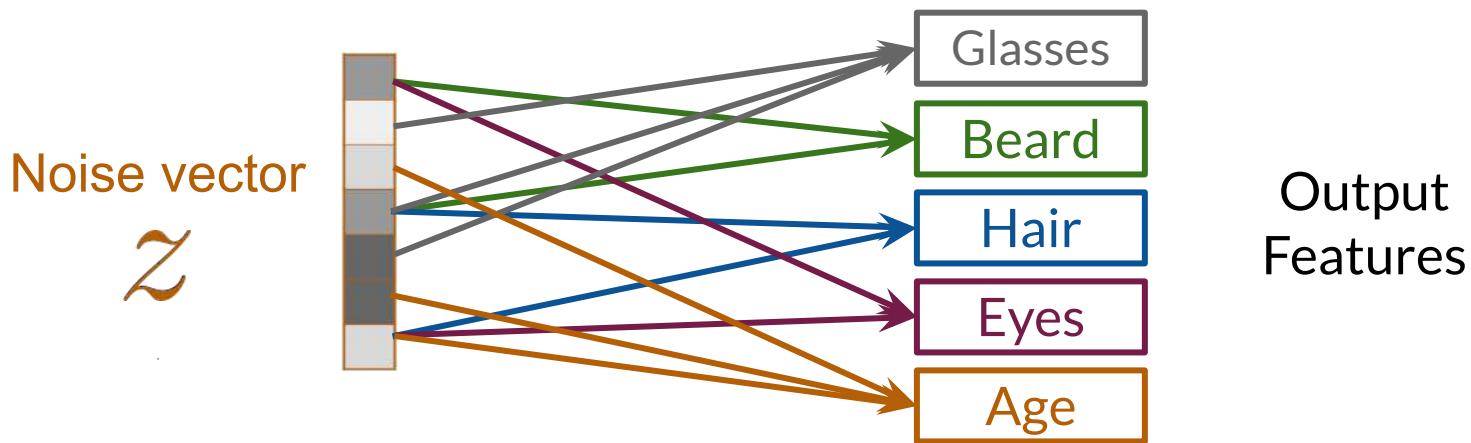
Z-Space Entanglement



Z-Space Entanglement

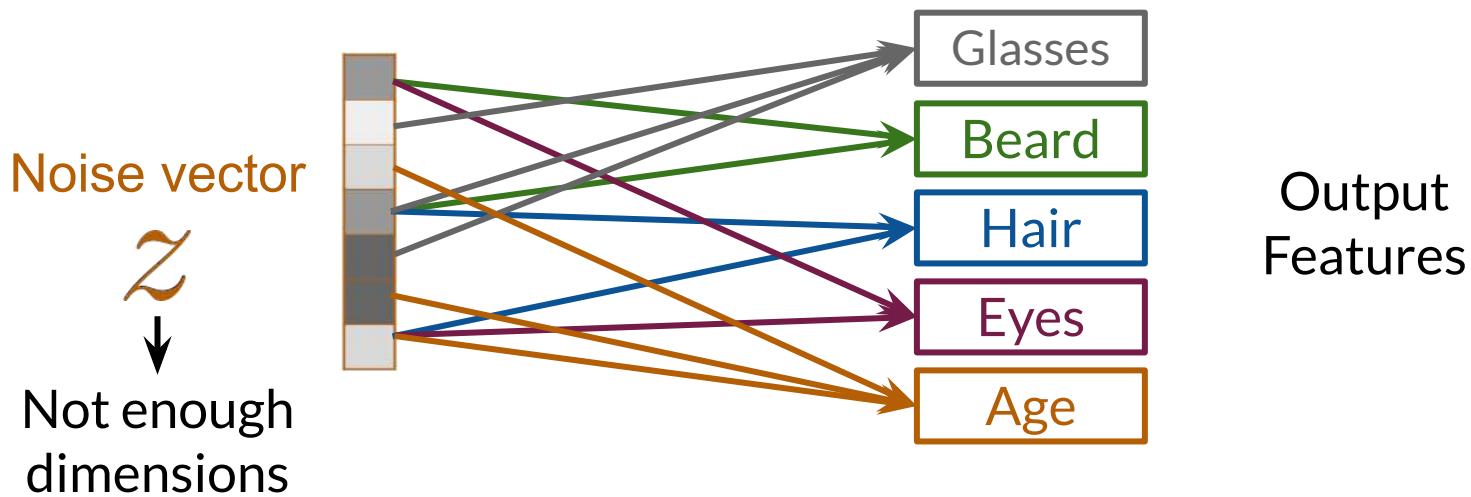


Z-Space Entanglement



It is not possible to control single output features

Z-Space Entanglement



It is not possible to control single output features

Summary

- When trying to control one feature, others that are correlated change
- Z-space entanglement makes controllability difficult, if not impossible
- Entanglement happens when z does not have enough dimensions



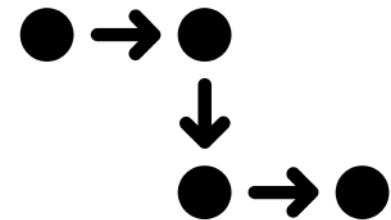


deeplearning.ai

Classifier Gradients

Outline

- How to use classifiers to find directions in the Z-space
- Requirements to use this method



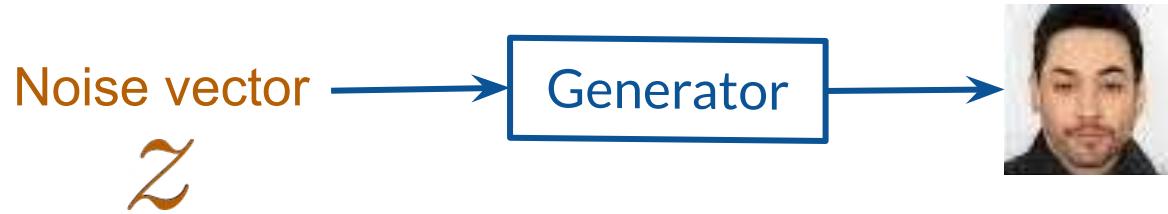
Classifier Gradients

Classifier Gradients

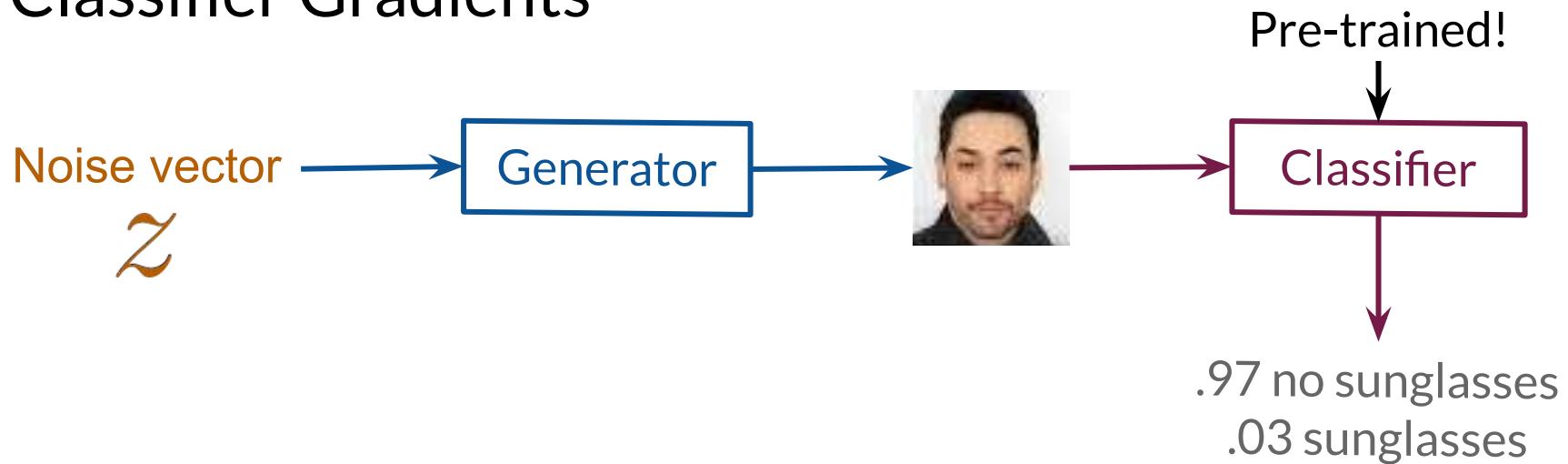
Noise vector

\tilde{z}

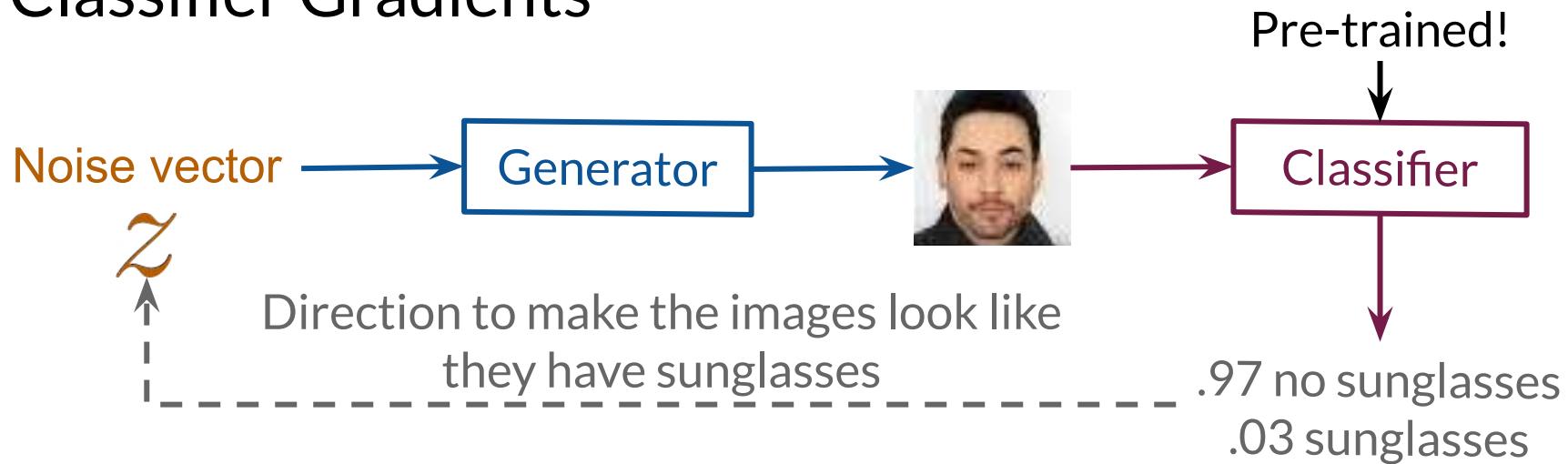
Classifier Gradients



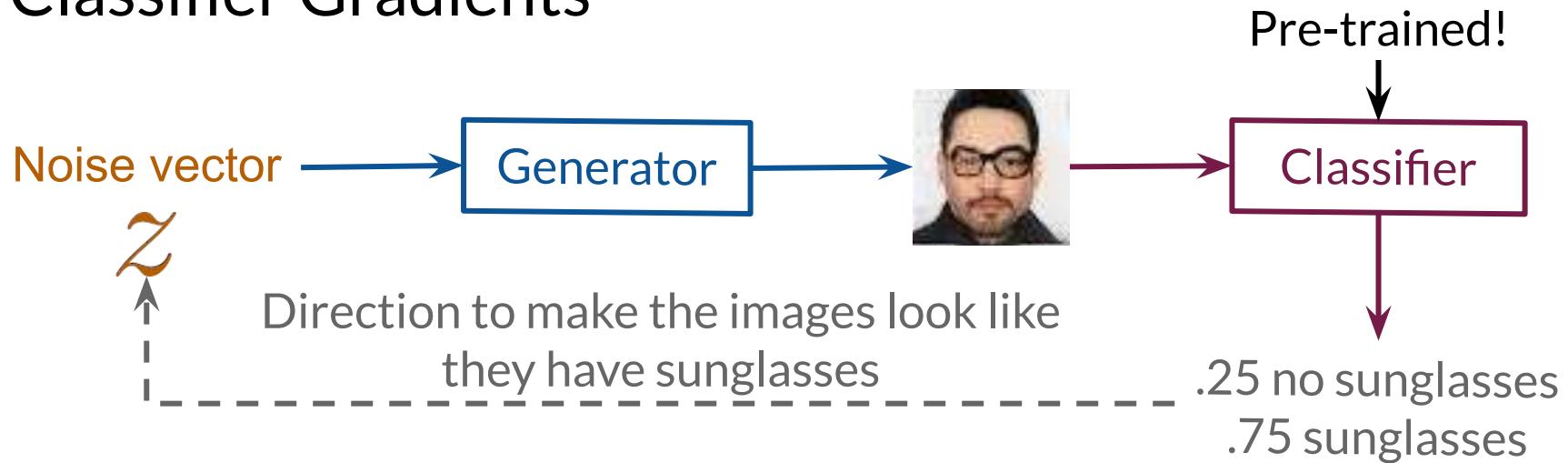
Classifier Gradients



Classifier Gradients

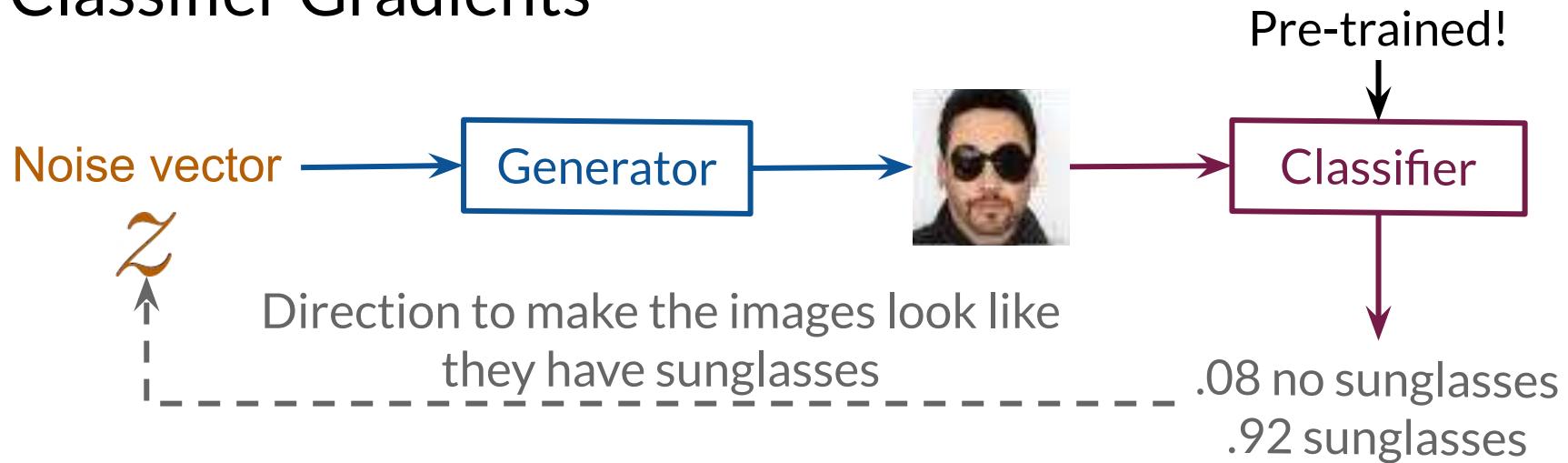


Classifier Gradients



Modify just the **noise vector** until the feature emerges

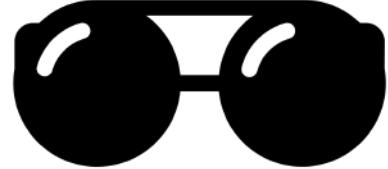
Classifier Gradients



Modify just the **noise vector** until the feature emerges

Summary

- Classifiers can be used to find directions in the Z-space
- To find directions, the updates are done just to the noise vector





deeplearning.ai

Disentanglement

Outline

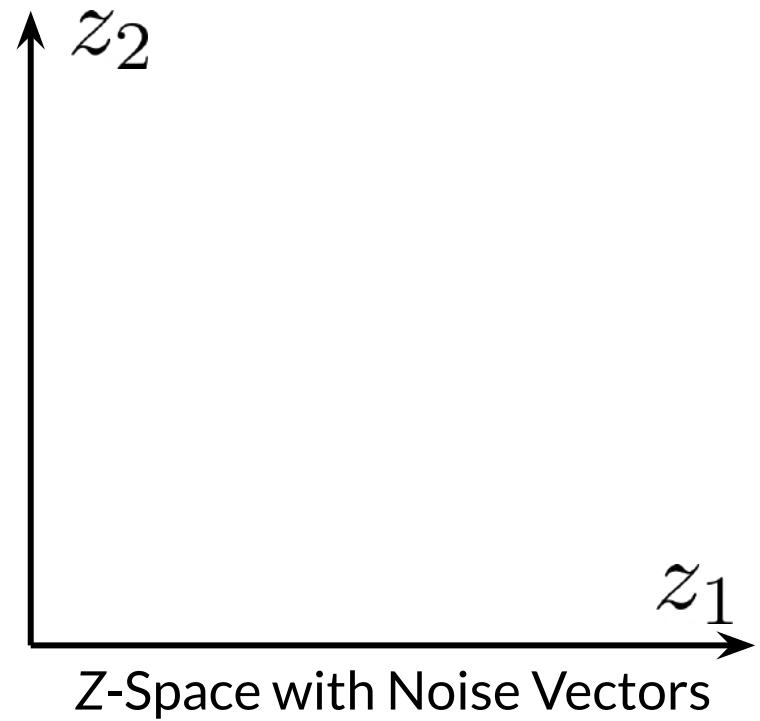
- What a disentangled Z-space means
- Ways to encourage disentangled Z-spaces



Disentangled Z-Space

$$v_1 = [1, 2, 3, \dots]$$

$$v_2 = [5, 6, 7, \dots]$$

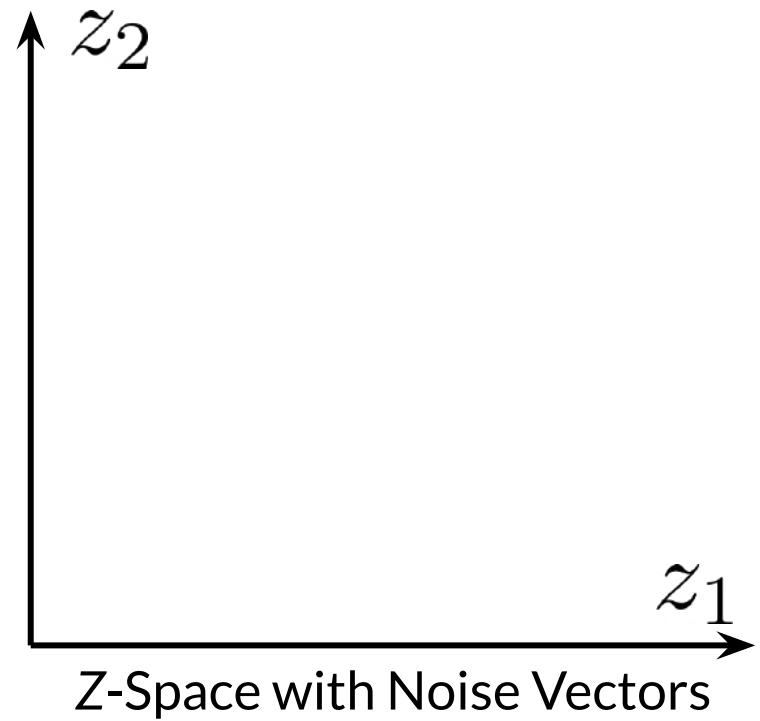


Disentangled Z-Space

$$v_1 = [\underset{\text{Hair color}}{\textcolor{brown}{1}}, 2, 3, \dots]$$

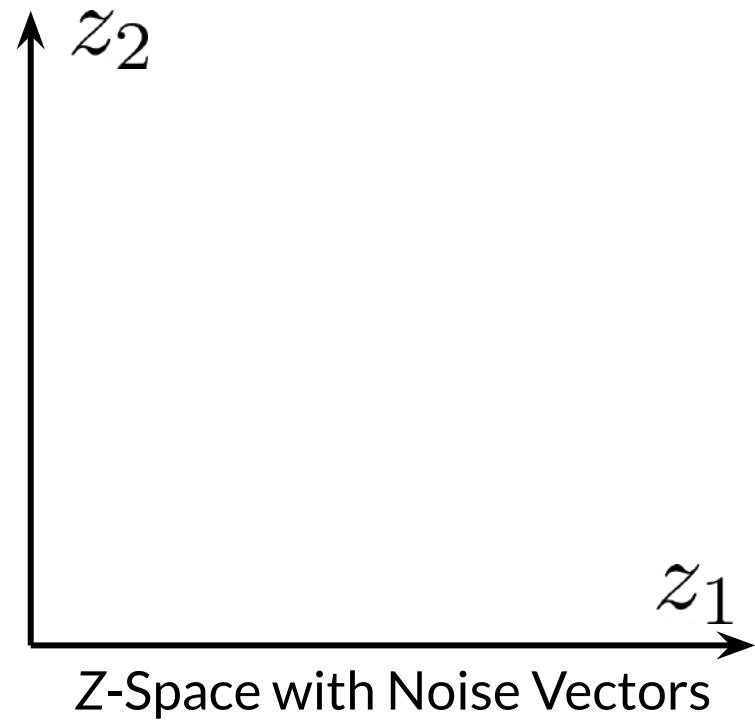
$$v_2 = [\underset{\text{Hair color}}{\textcolor{brown}{5}}, 6, 7, \dots]$$

Hair
color



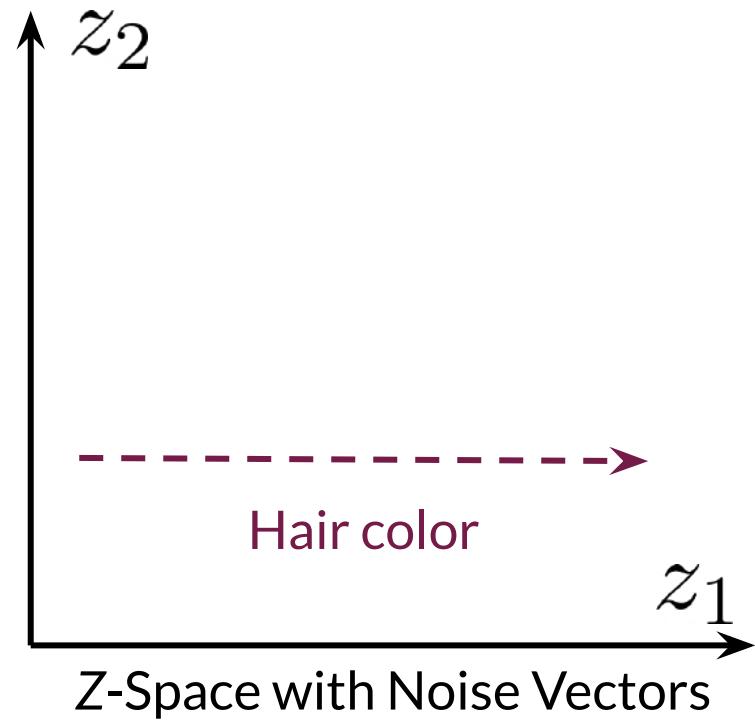
Disentangled Z-Space

$$\begin{array}{c} z_1 \quad z_2 \\ v_1 = [\textcolor{red}{1}, \textcolor{blue}{2}, 3, \dots] \\ v_2 = [\textcolor{red}{5}, \textcolor{blue}{6}, 7, \dots] \\ \text{Hair color} \qquad \qquad \text{Hair length} \end{array}$$



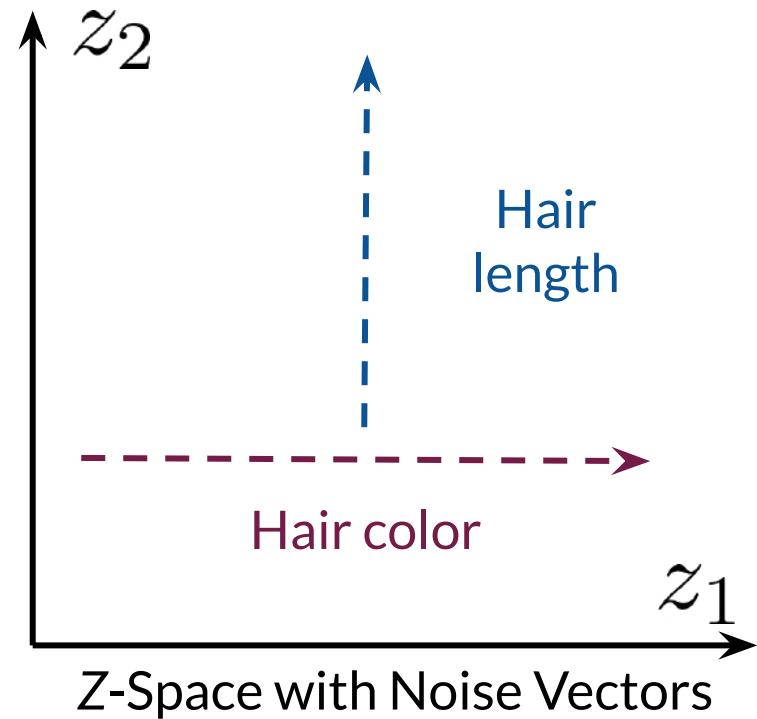
Disentangled Z-Space

$$v_1 = [\underset{\text{Hair color}}{\textcolor{red}{1}}, \underset{\text{Hair length}}{\textcolor{blue}{2}}, 3, \dots]$$
$$v_2 = [\underset{\text{Hair color}}{\textcolor{red}{5}}, \underset{\text{Hair length}}{\textcolor{blue}{6}}, 7, \dots]$$



Disentangled Z-Space

$$\begin{matrix} & z_1 & z_2 \\ v_1 = & [\textcolor{pink}{1}, & 2, & 3, \dots] \\ v_2 = & [\textcolor{pink}{5}, & 6, & 7, \dots] \\ & \text{Hair color} & \text{Hair length} \end{matrix}$$

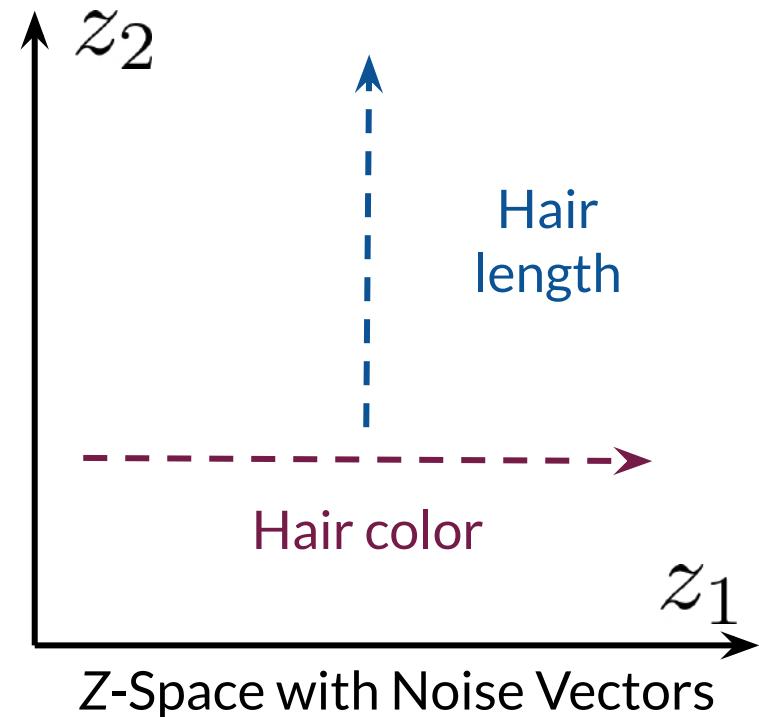


Disentangled Z-Space

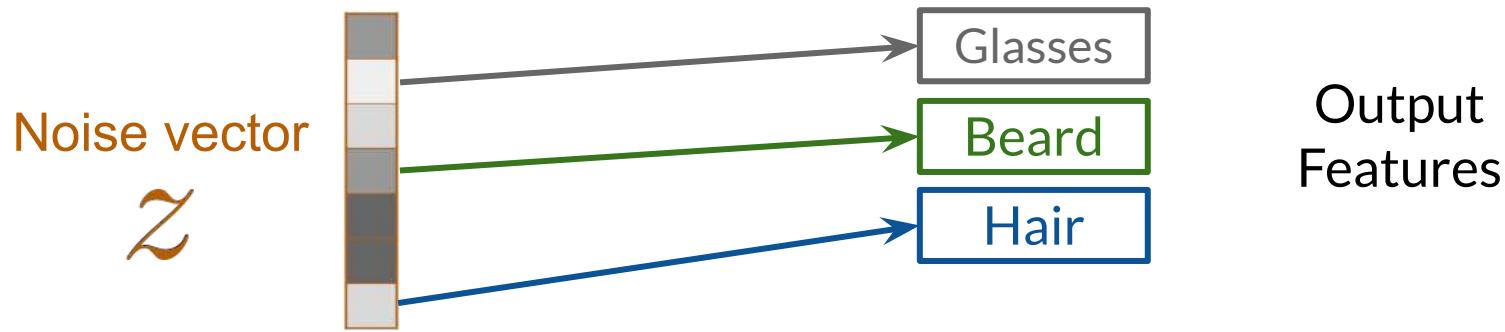
$$v_1 = [\underset{\text{Hair color}}{\textcolor{red}{z_1}}, \underset{\text{Hair length}}{\textcolor{blue}{z_2}}, 3, \dots]$$

$$v_2 = [\underset{\text{Hair color}}{\textcolor{red}{5}}, \underset{\text{Hair length}}{\textcolor{blue}{6}}, 7, \dots]$$

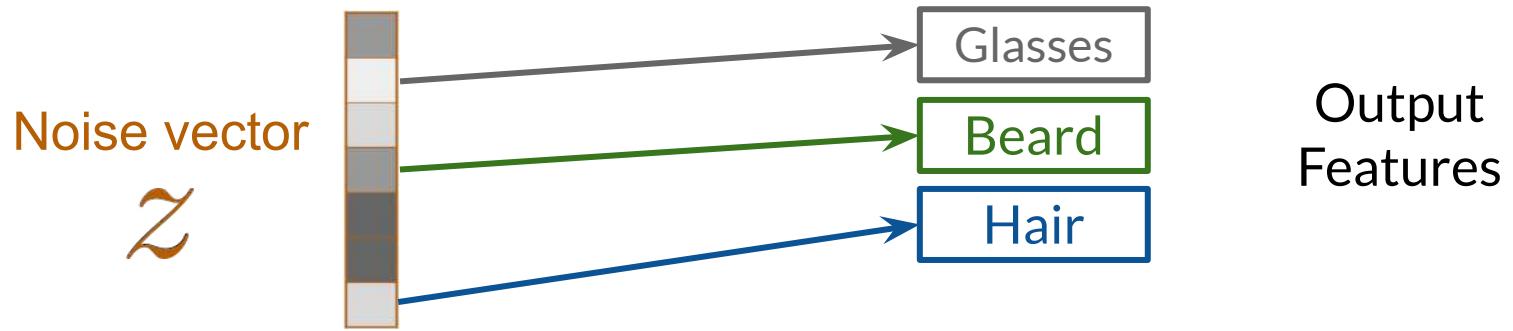
Latent factors of variation



Disentangled Z-Space

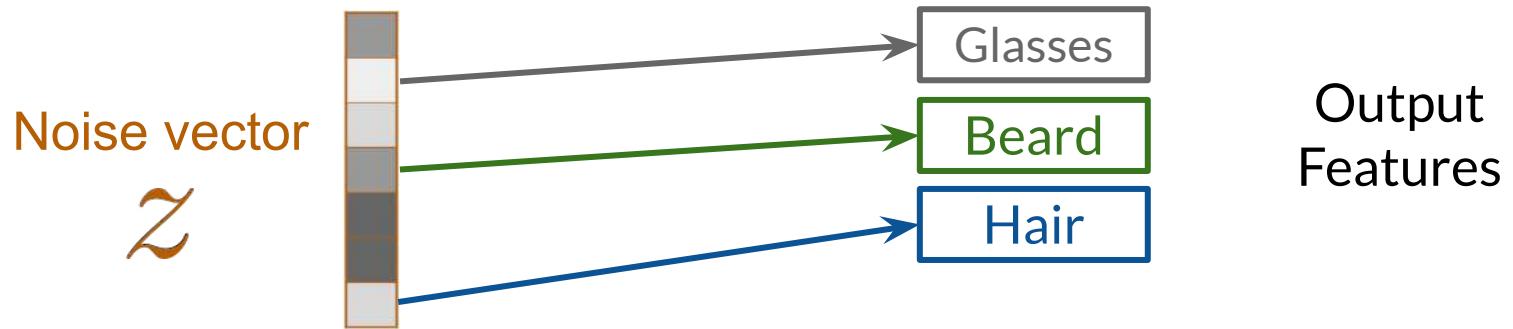


Disentangled Z-Space

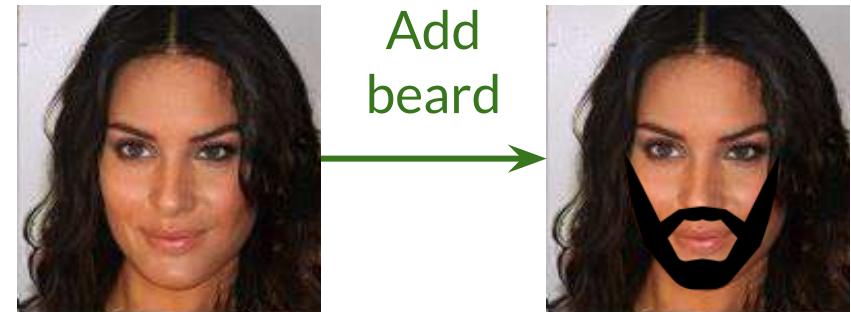


Changes to one feature
don't affect the others

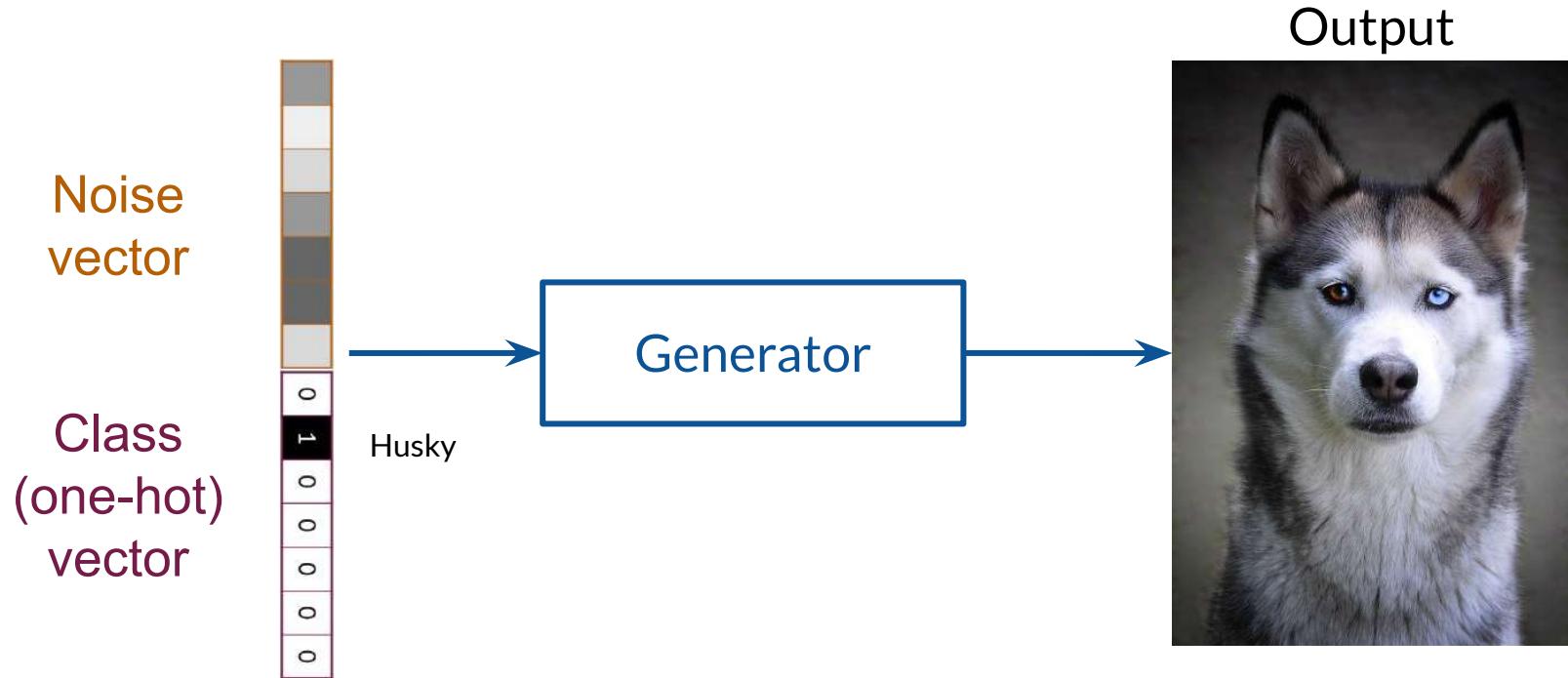
Disentangled Z-Space



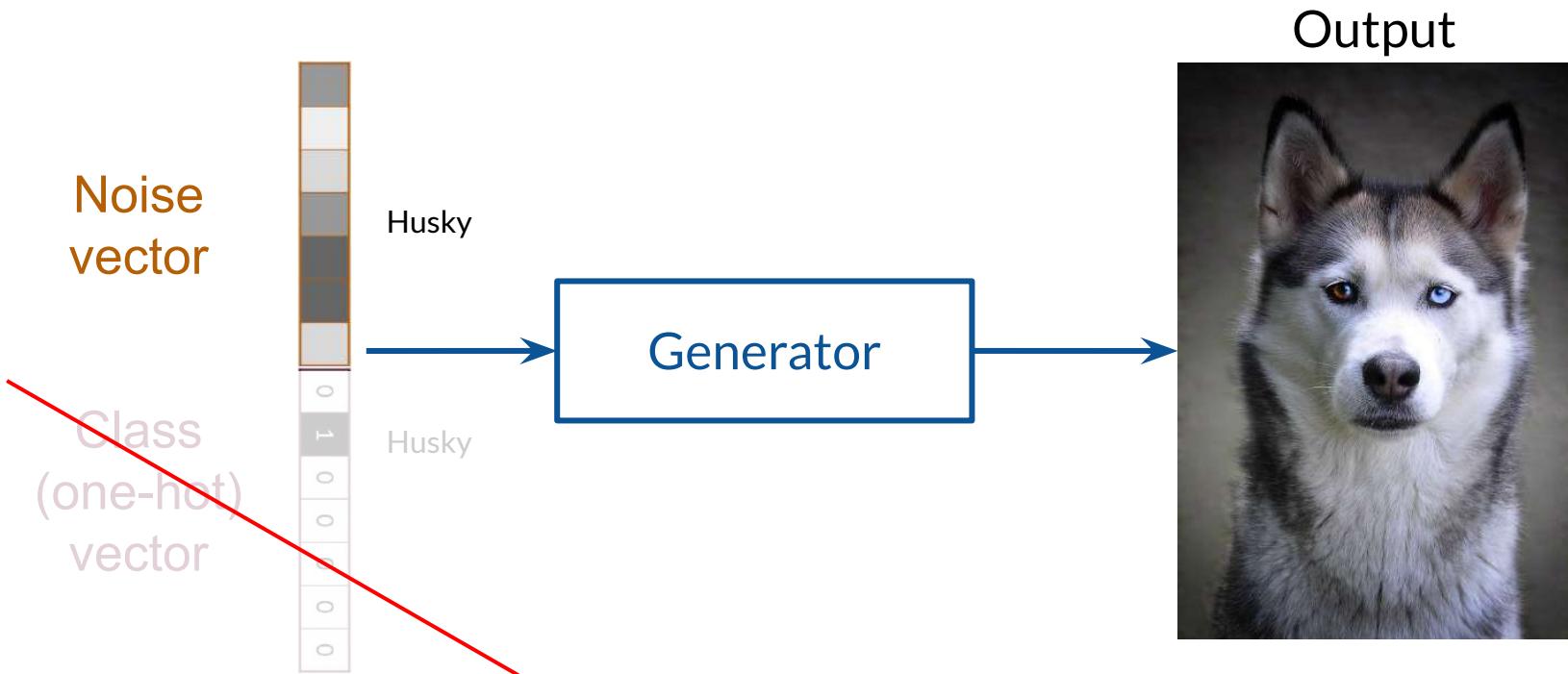
Changes to one feature
don't affect the others



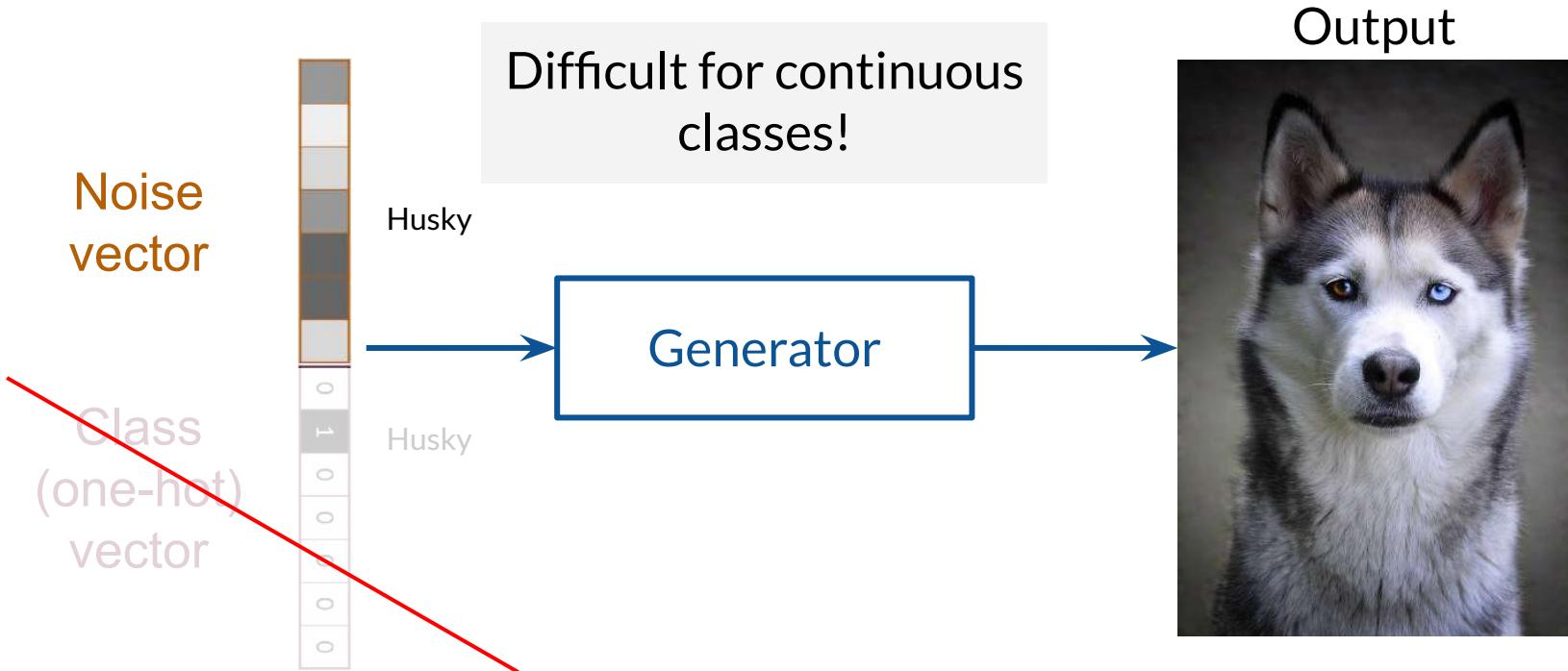
Encourage Disentanglement: Supervision



Encourage Disentanglement: Supervision



Encourage Disentanglement: Supervision



Encourage Disentanglement: Loss Function

$$v_1 = [1, 2, 3, \dots]$$

$$v_2 = [5, 6, 7, \dots]$$

Encourage Disentanglement: Loss Function

$$v_1 = [1, 2, 3, \dots]$$

$$v_2 = [5, 6, 7, \dots]$$

$$L_{\text{new}} = L_{\text{original}} + \text{reg}_d$$

The diagram illustrates the formula for the new loss function. It shows the sum of two terms: L_{original} and reg_d . The term L_{original} is enclosed in a green box and has a green arrow pointing down to the text "Original loss". The term reg_d is enclosed in an orange box and has an orange arrow pointing down to the text "Regularization".

Original loss Regularization

Encourage Disentanglement: Loss Function

$$v_1 = [1, 2, 3, \dots]$$

$$v_2 = [5, 6, 7, \dots]$$

$$L_{\text{new}} = L_{\text{original}} + \text{reg}_d$$

Original loss Regularization

Can be any loss function
(e.g. BCE, W-Loss)

Encourage Disentanglement: Loss Function

$$\begin{aligned} z_1 & \quad z_2 \\ v_1 = [& \textcolor{pink}{1}, \textcolor{lightblue}{2}, 3, \dots] \\ v_2 = [& \textcolor{pink}{5}, \textcolor{lightblue}{6}, 7, \dots] \end{aligned}$$

$$L_{\text{new}} = \boxed{L_{\text{original}}} + \boxed{\text{reg}_d}$$

Original loss Regularization

Can be any loss function
(e.g. BCE, W-Loss)

Encourage Disentanglement: Loss Function

$$v_1 = [z_1, z_2, 3, \dots]$$

z_1 z_2

$v_1 = [1, 2, 3, \dots]$

$v_2 = [5, 6, 7, \dots]$

Output feature #1

Output feature #2

$$L_{\text{new}} = L_{\text{original}} + \text{reg}_d$$

Original loss Regularization

Can be any loss function
(e.g. BCE, W-Loss)

Summary

- Disentangled Z-spaces let you control individual features by corresponding z values directly to them
- There are supervised and unsupervised methods to achieve disentanglement



Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>



deeplearning.ai

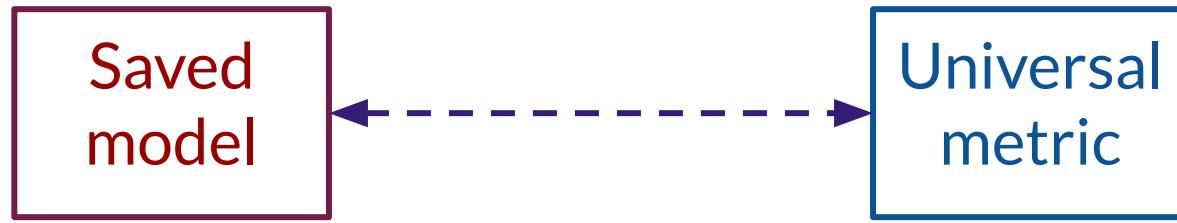
Evaluation

Outline

- Why evaluating GANs is hard
- Two properties: fidelity and diversity



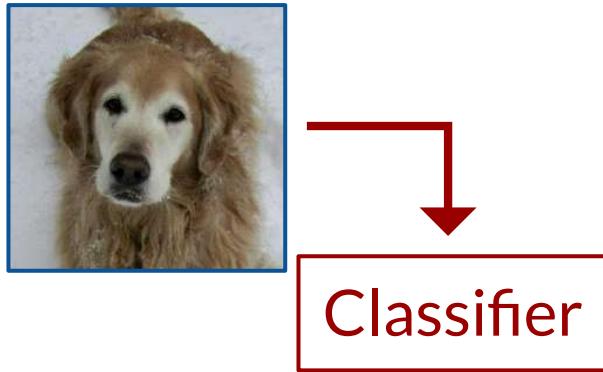
Why is evaluating GANs hard?



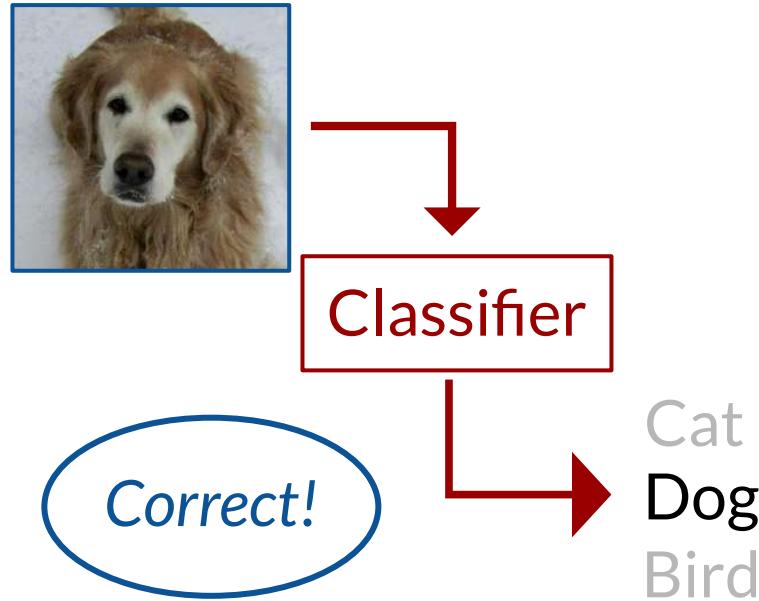
Why is evaluating GANs hard?

Classifier

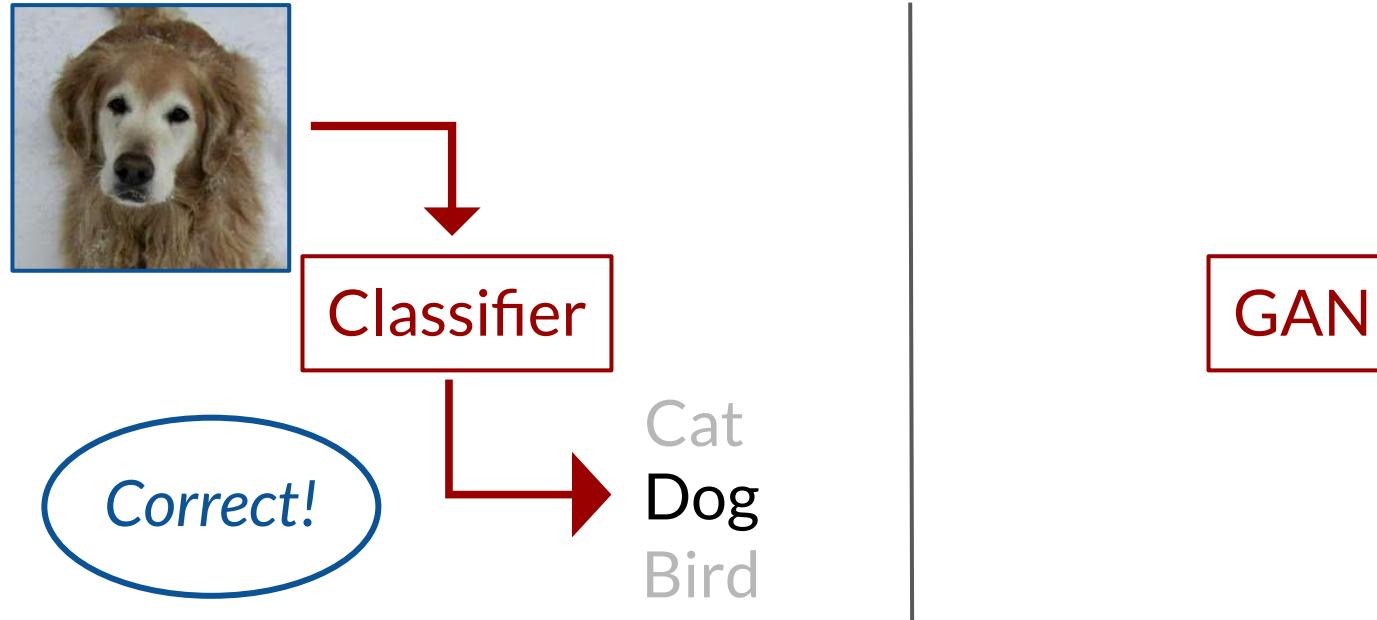
Why is evaluating GANs hard?



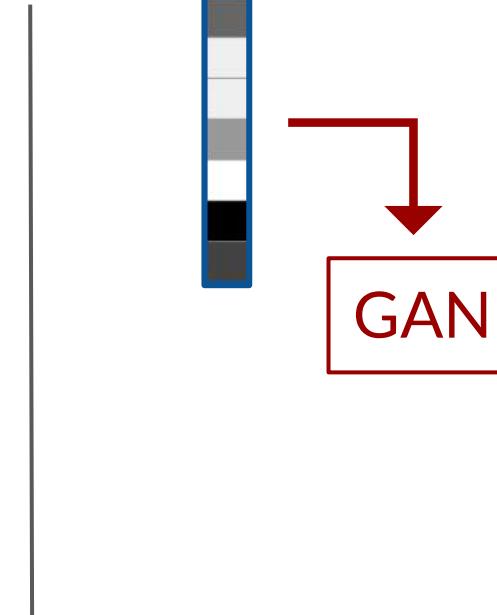
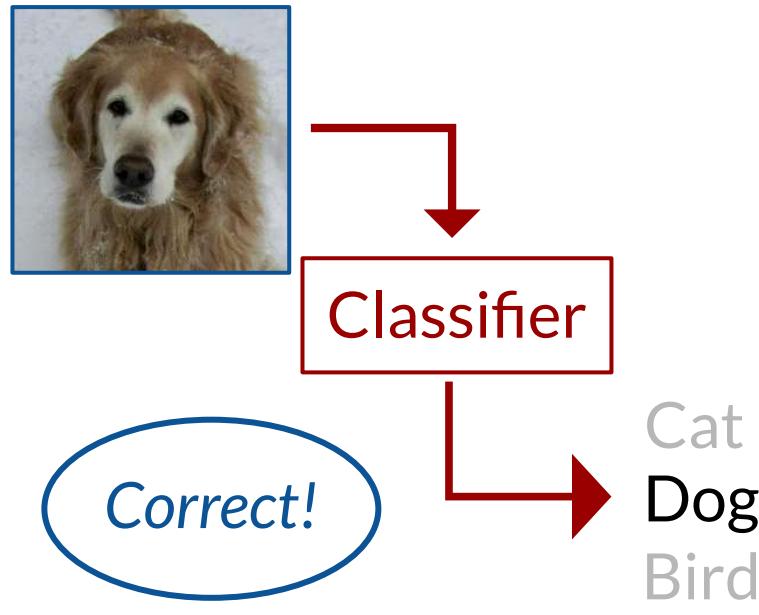
Why is evaluating GANs hard?



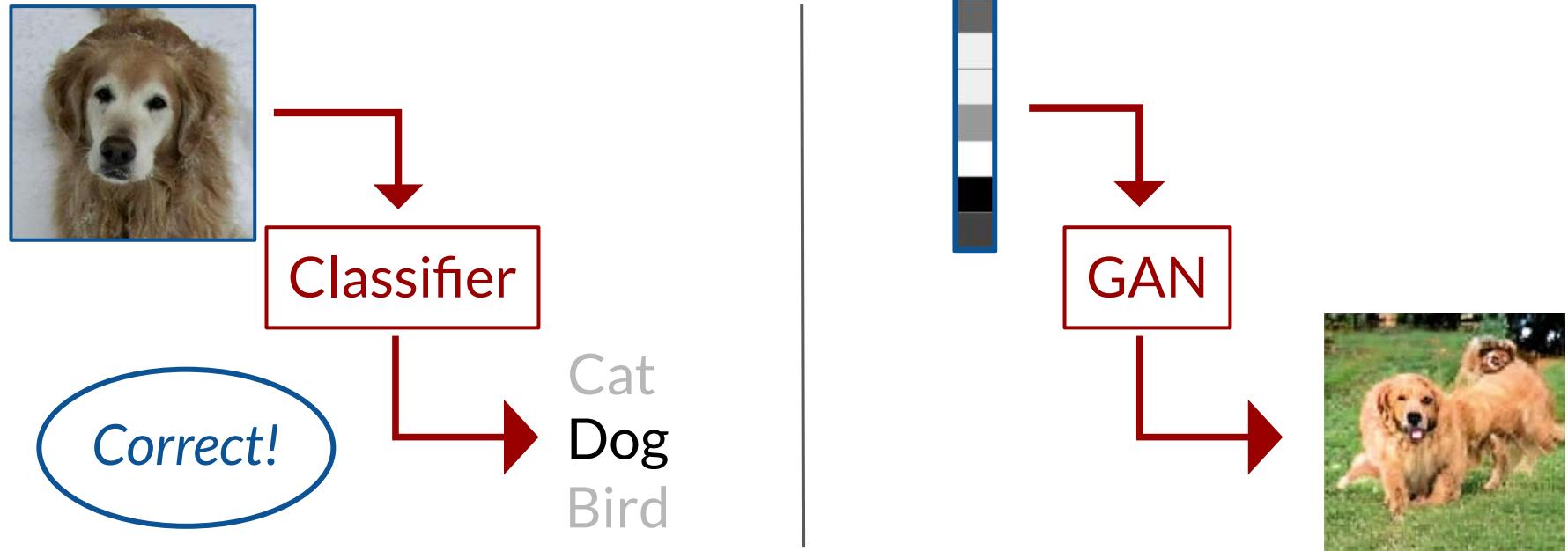
Why is evaluating GANs hard?



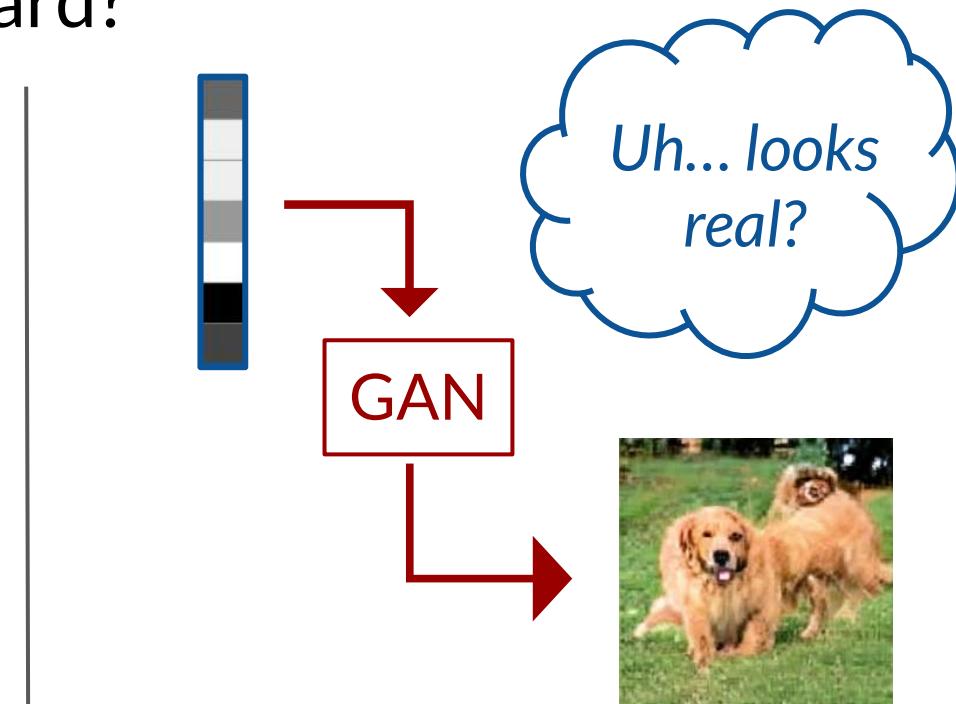
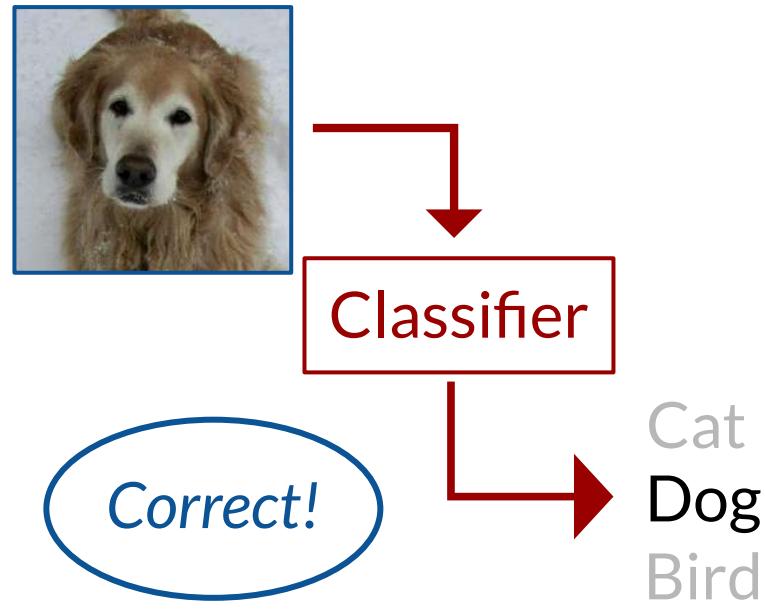
Why is evaluating GANs hard?



Why is evaluating GANs hard?



Why is evaluating GANs hard?



Two Important Properties

Fidelity:
quality of images



(Left) Available at: <https://github.com/NVlabs/stylegan>

Two Important Properties

Fidelity:
quality of images



Diversity:
variety of images



(Left) Available at: <https://github.com/NVlabs/stylegan>

Fidelity

Fidelity

7 777

Fake

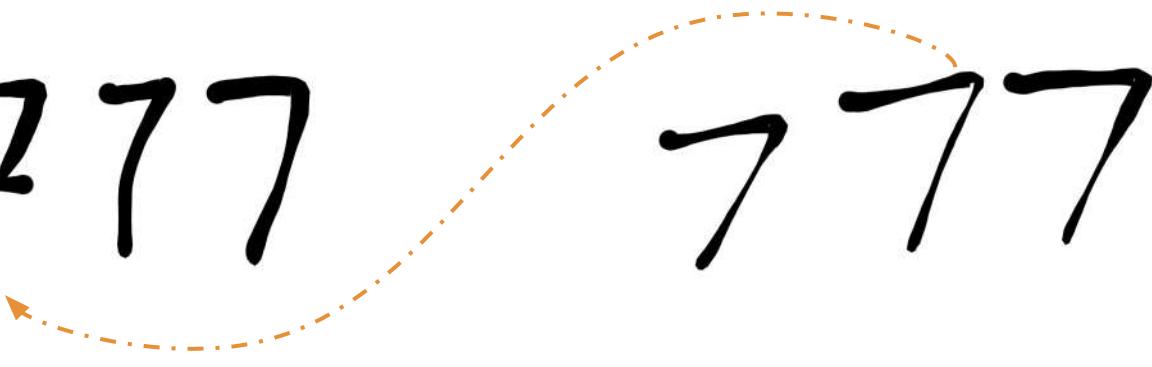
Fidelity

7 7 7 7

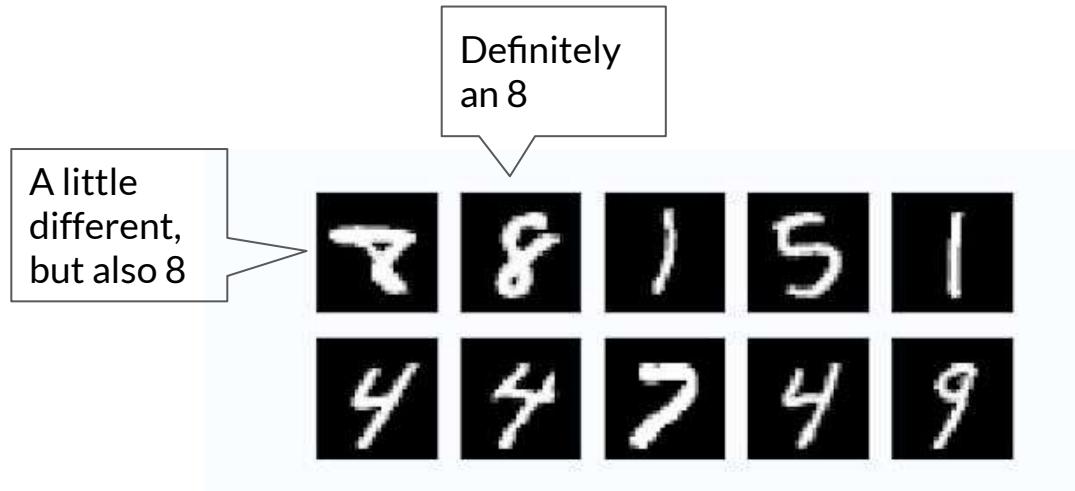
7 7 7 7

Real

Fake

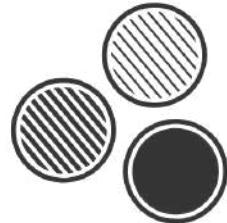


Diversity



Summary

- No ground-truth = challenging to evaluate
- Fidelity measures image quality and diversity measures variety
- Evaluation metrics try to quantify fidelity & diversity





deeplearning.ai

Comparing Images

Outline

- Pixel distance
- Feature distance



Pixel Distance

200	50	200
200	50	200
200	50	200

Real

200	50	200
200	50	200
200	50	200

Fake

0	0	0
0	0	0
0	0	0

Absolute
difference

Pixel Distance

200	50	200
200	50	200
200	50	200

Real

50	200	200
50	200	200
50	200	200

Fake

0	0	0
0	0	0
0	0	0

Absolute
difference

Feature Distance

Feature Distance

Real



Fake



Feature Distance

Real



2 eyes,
2 droopy ears,
1 nose, ...

Fake



2 eyes,
1 droopy ear,
5 legs,
1 nose, ...

Feature Distance

Real

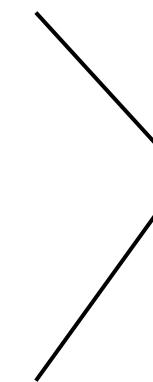


2 eyes,
2 droopy ears,
1 nose, ...

Fake



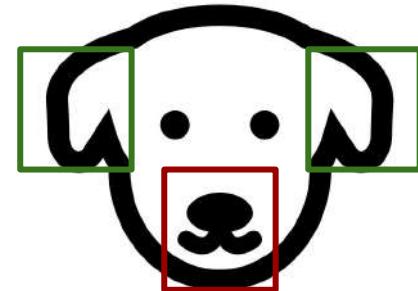
2 eyes,
1 droopy ear,
5 legs,
1 nose, ...



Compare
features!

Summary

- Pixel distance is simple but unreliable
- Feature distance uses the higher level features of an image, making it more reliable



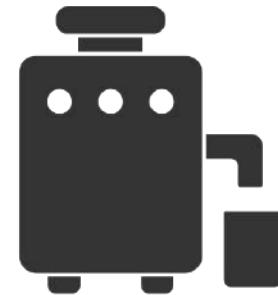


deeplearning.ai

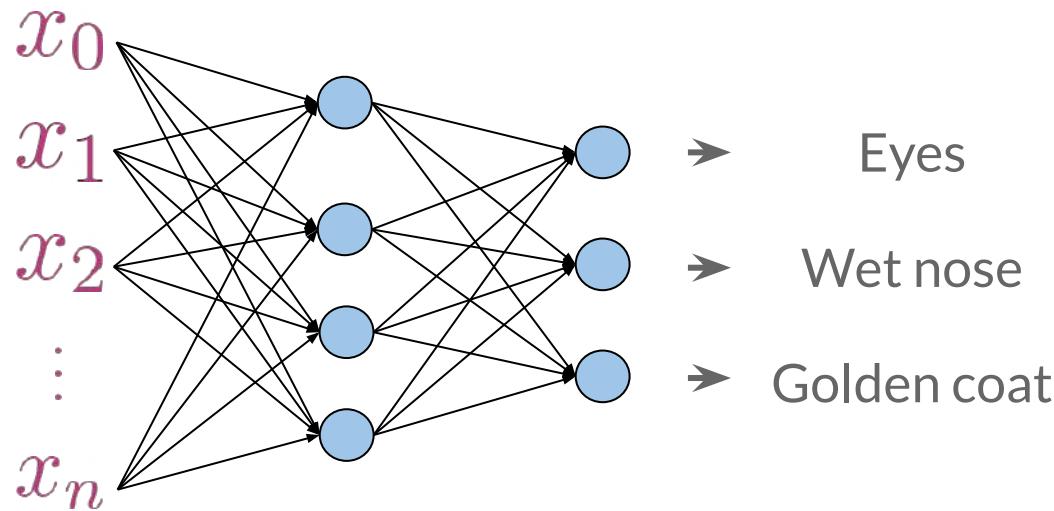
Feature Extraction

Outline

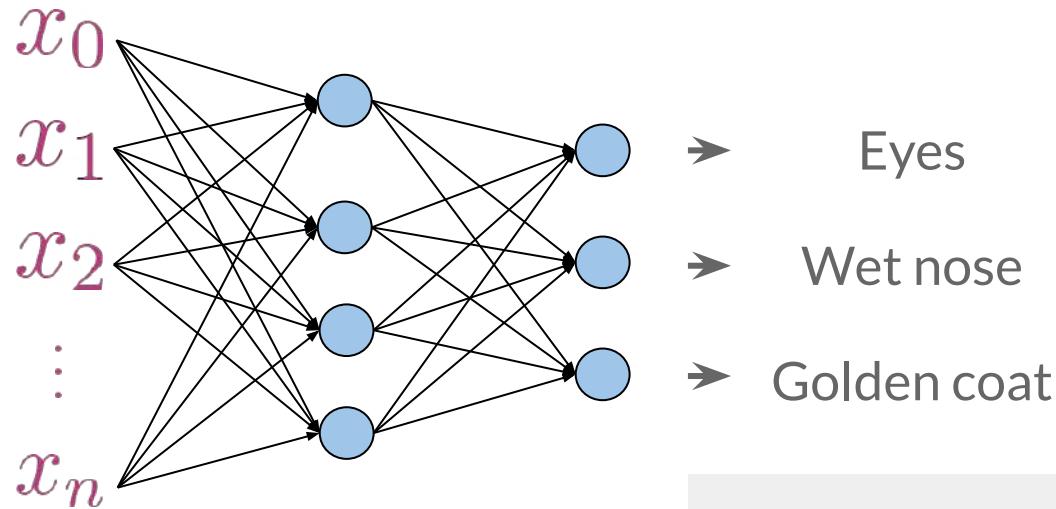
- Feature extraction using pre-trained classifiers
- ImageNet dataset



Classifier → Feature Extractor

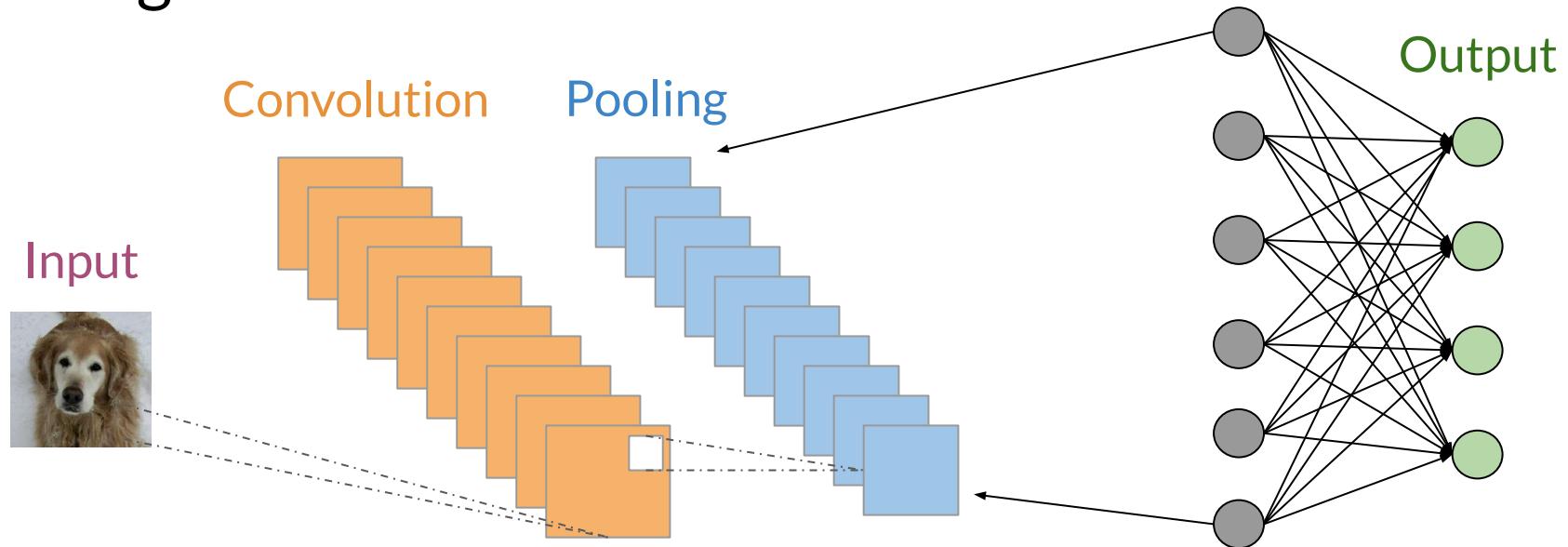


Classifier → Feature Extractor

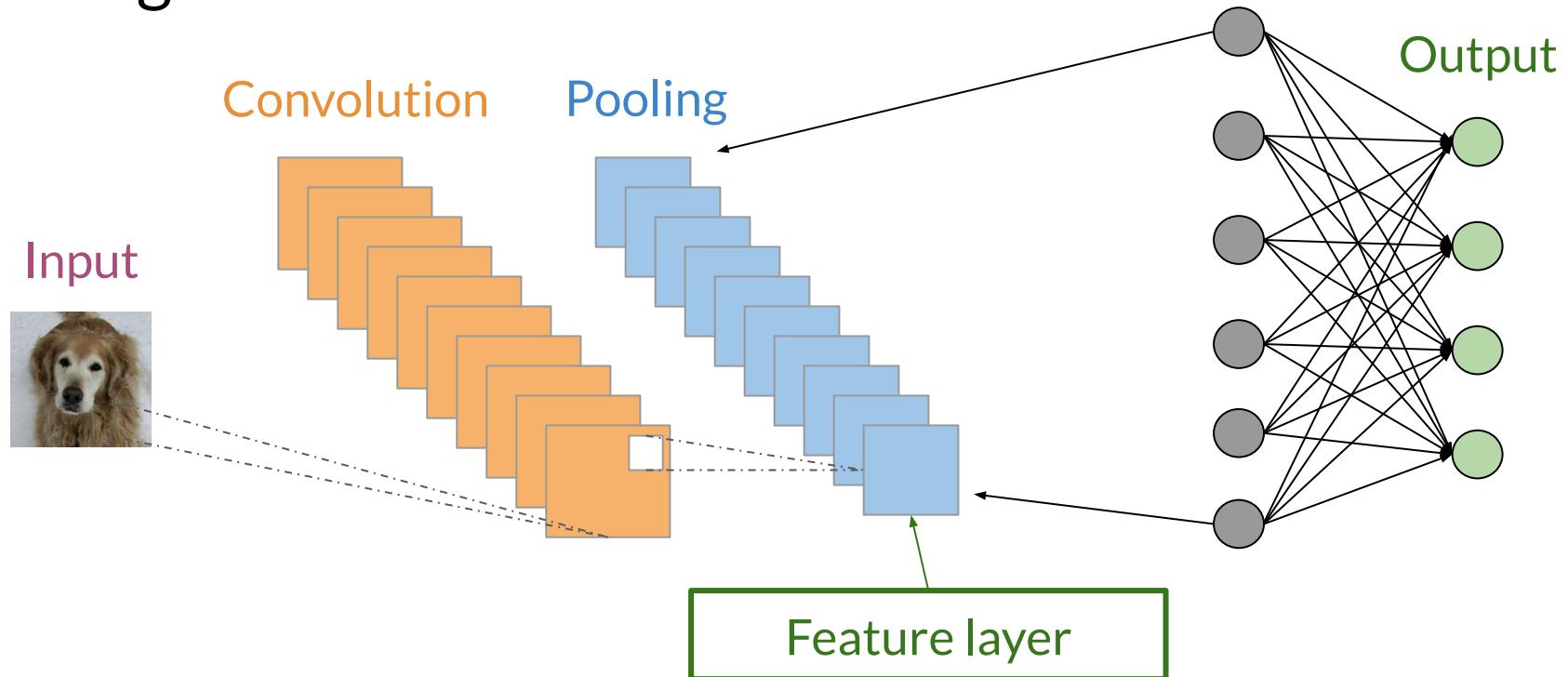


Extensively pre-trained
classifiers available to use

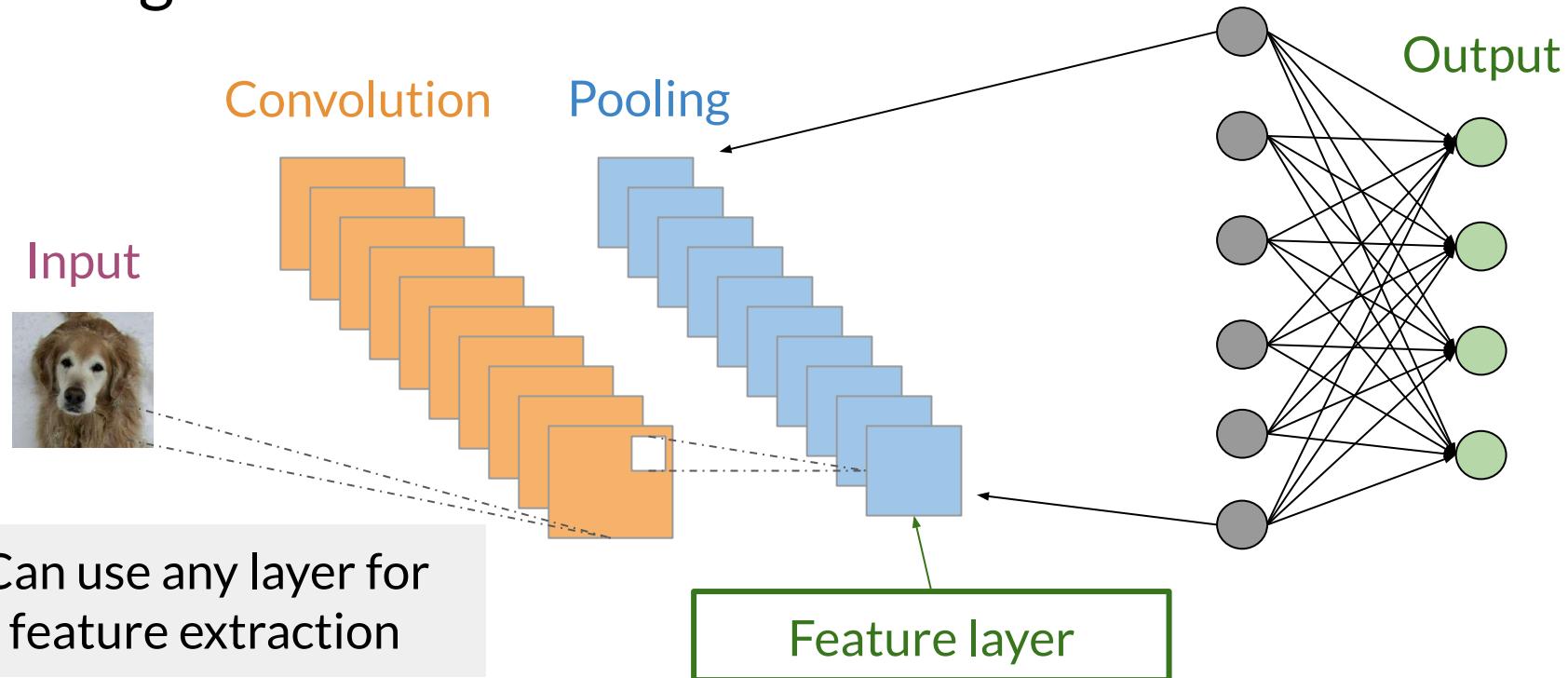
Using a Pre-trained Classifier



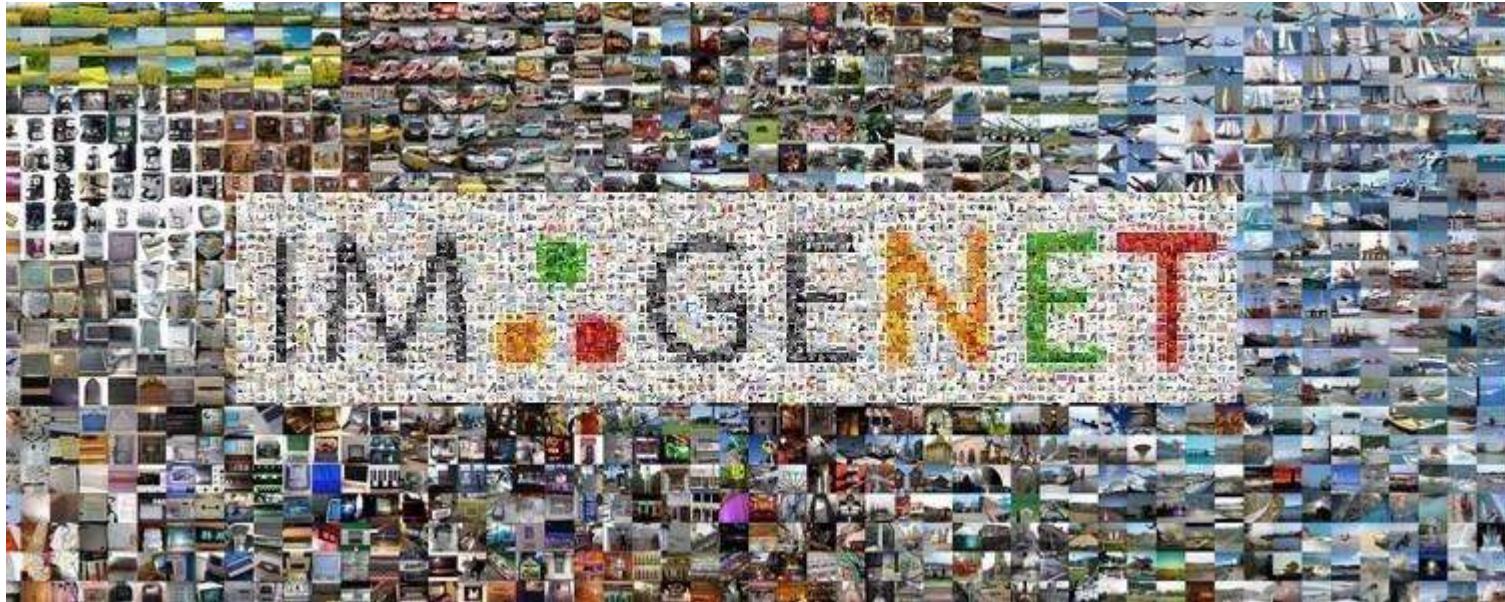
Using a Pre-trained Classifier



Using a Pre-trained Classifier



ImageNet



© 2016
Stanford
Vision Lab

ImageNet Attributes

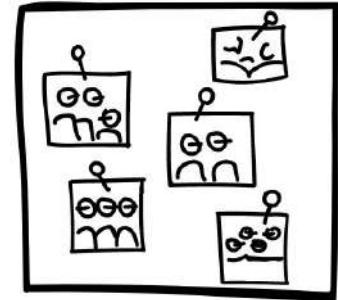
- > 14 million images
- > 20,000 categories



© 2016
Stanford
Vision Lab

Summary

- Classifiers can be used as feature extractors by cutting the network at earlier layers
- The last pooling layer is most commonly used for feature extraction
- Best to use classifiers that have been trained on large datasets—ImageNet





deeplearning.ai

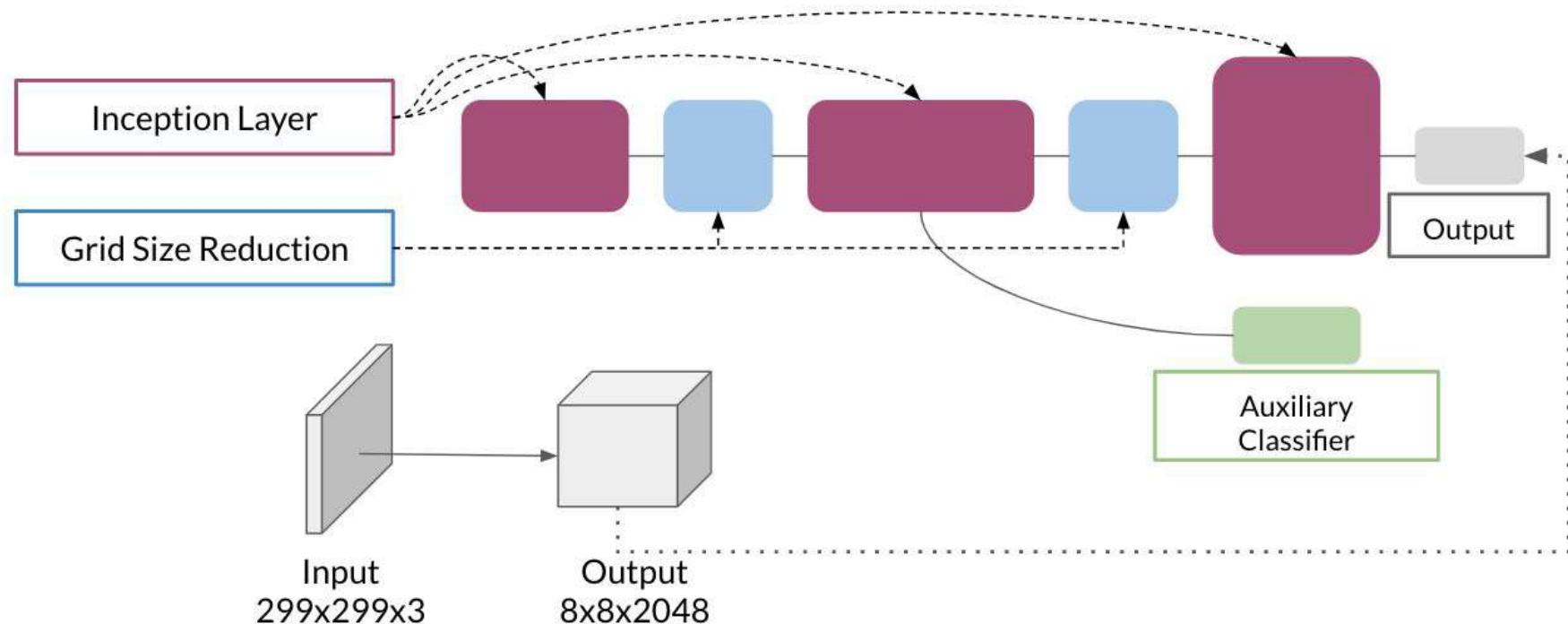
Inception-v3 and Embeddings

Outline

- Inception-v3 architecture
- Comparing extracted feature embeddings



Inception-v3 Architecture



Based on: <https://medium.com/@sh.tsang/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c>

Embeddings



2 eyes,
2 droopy ears,
1 nose, ...

Embeddings



Inception-v3

2 eyes,
2 droopy ears,
1 nose, ...

= $\phi(x)$

Embedding of x

Comparing Embeddings



Fake

Comparing Embeddings



Fake



Real

Summary

- Commonly used feature extractor: Inception-v3 classifier, which is pre-trained on ImageNet, with the output layer cut off
- These features are called embeddings
- Compare embeddings to get the feature distance





deeplearning.ai

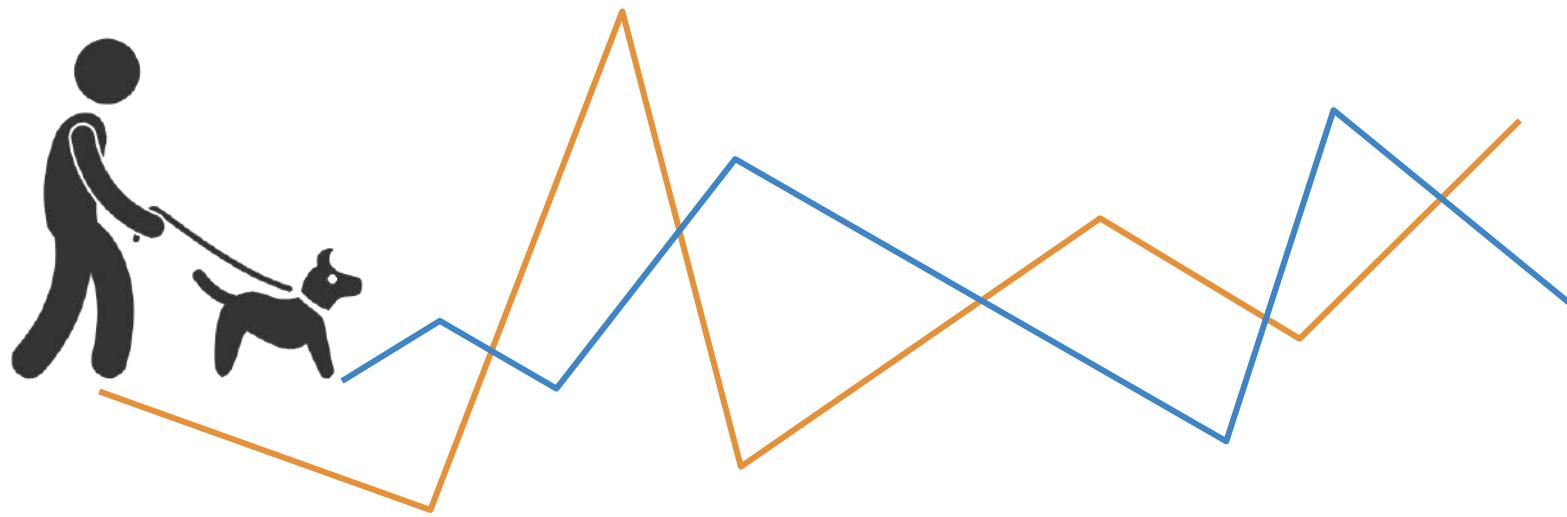
Fréchet Inception Distance (FID)

Outline

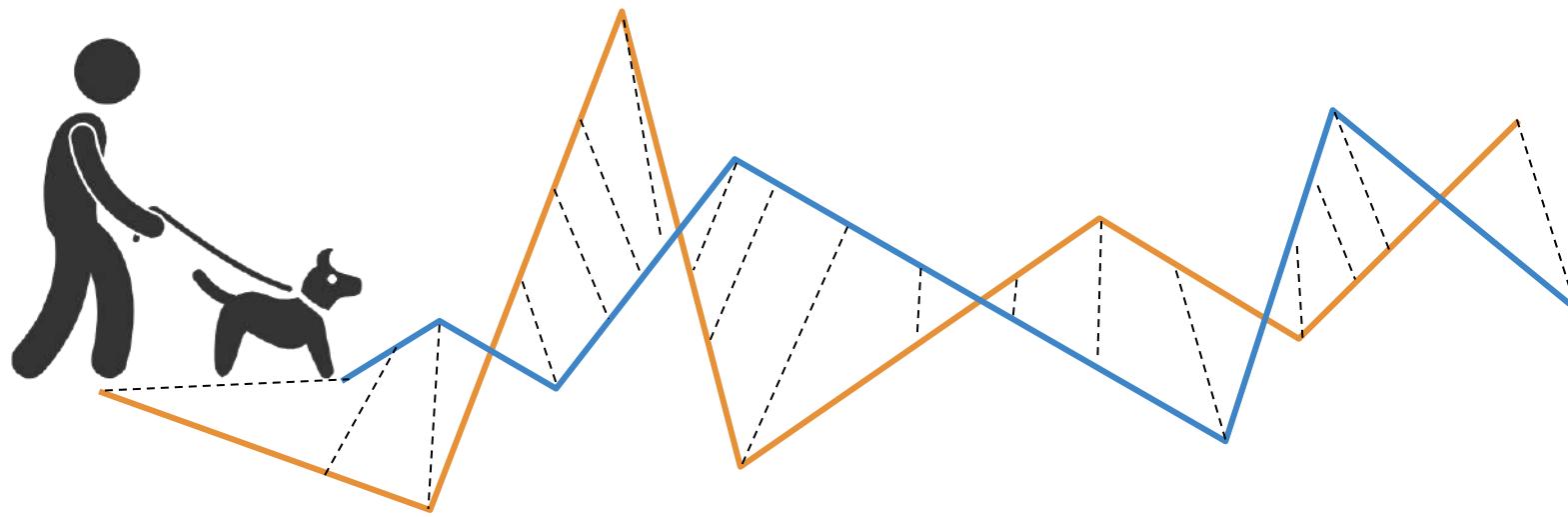
- Fréchet distance
- Evaluation method: Fréchet Inception Distance (FID)
- FID shortcomings



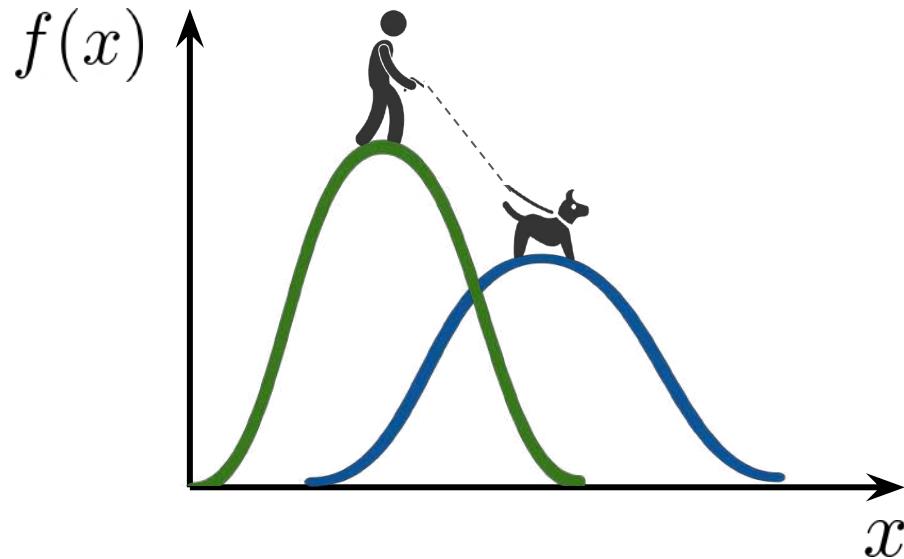
Fréchet Distance



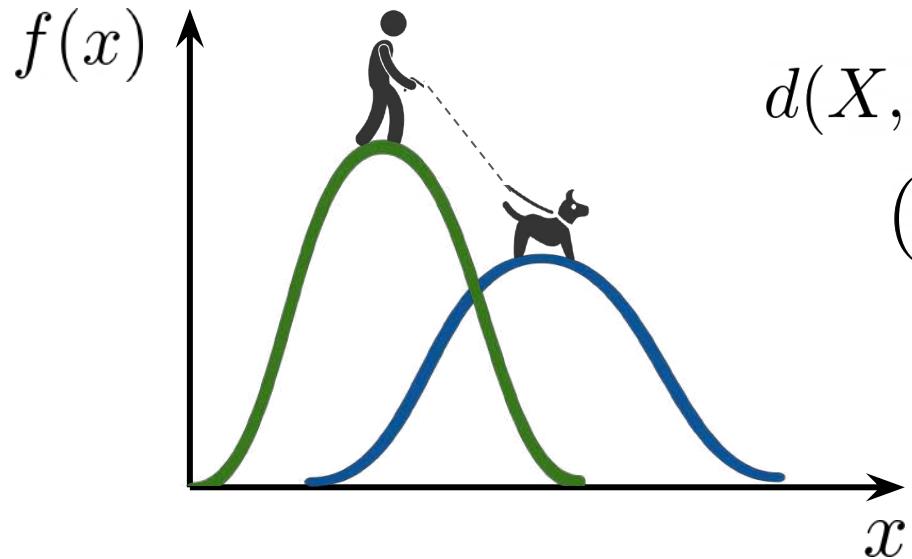
Fréchet Distance



Fréchet Distance Between Normal Distributions



Fréchet Distance Between Normal Distributions



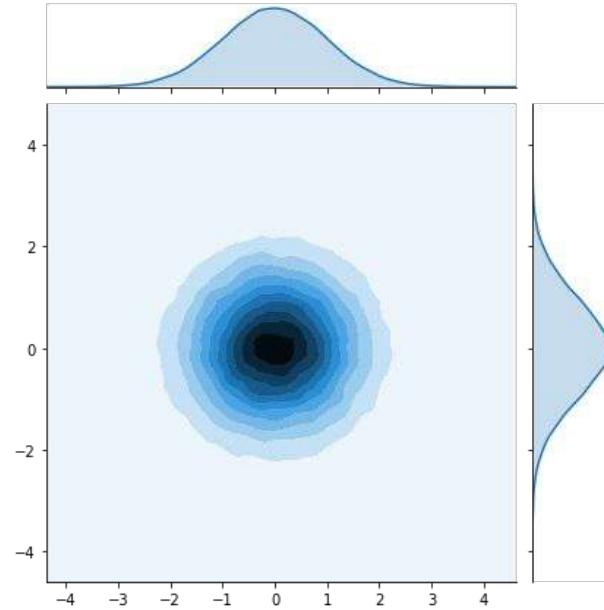
$$d(X, Y) =$$

$$(\mu_X - \mu_Y)^2 + (\sigma_X - \sigma_Y)^2$$

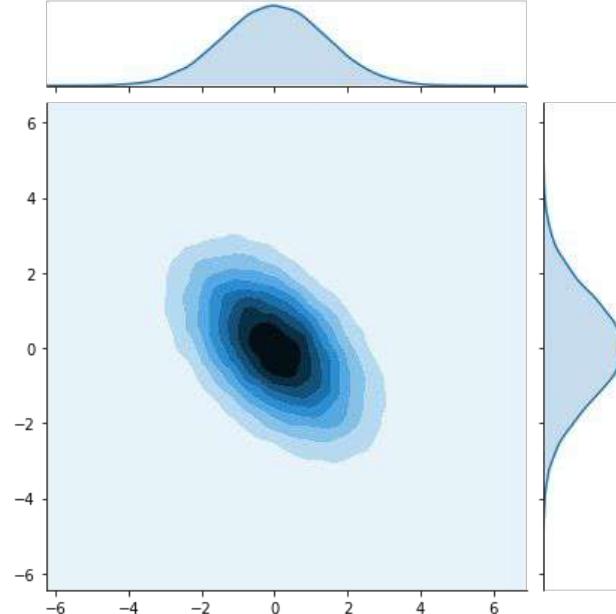
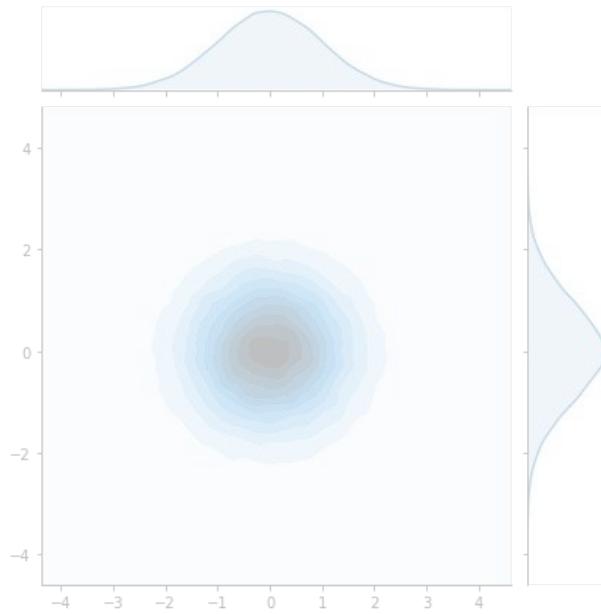
Mean

Standard deviation

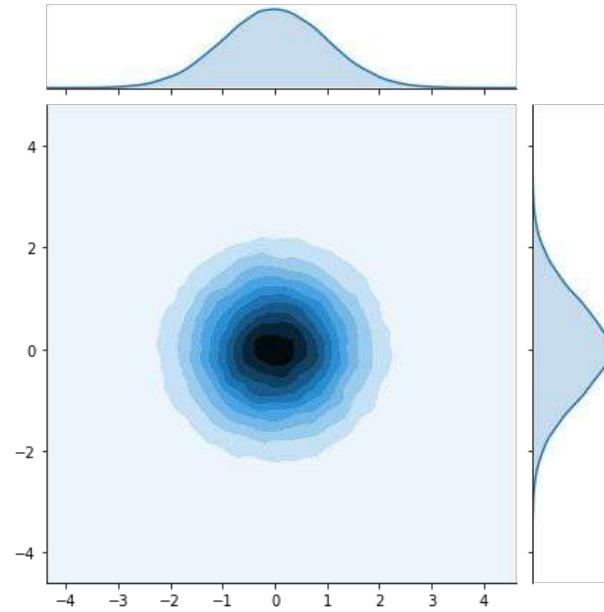
Multivariate Normal Distributions



Multivariate Normal Distributions



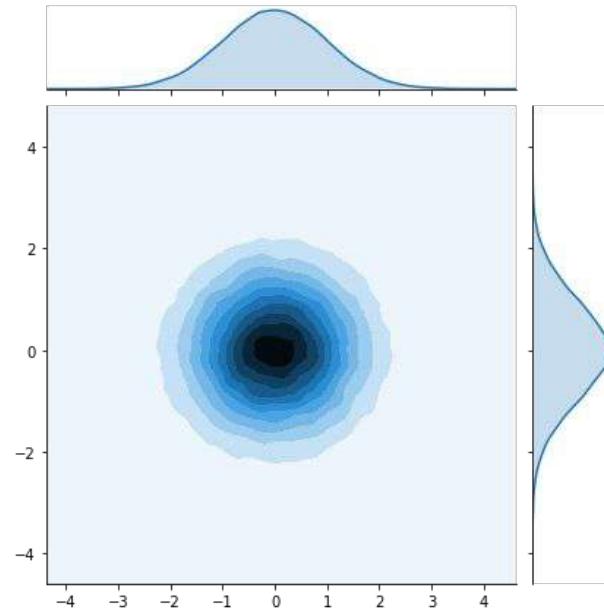
Multivariate Normal Distributions



$$\Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Covariance
matrix

Multivariate Normal Distributions



0's everywhere but the diagonal =
all dimensions are *independent*

$$\Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

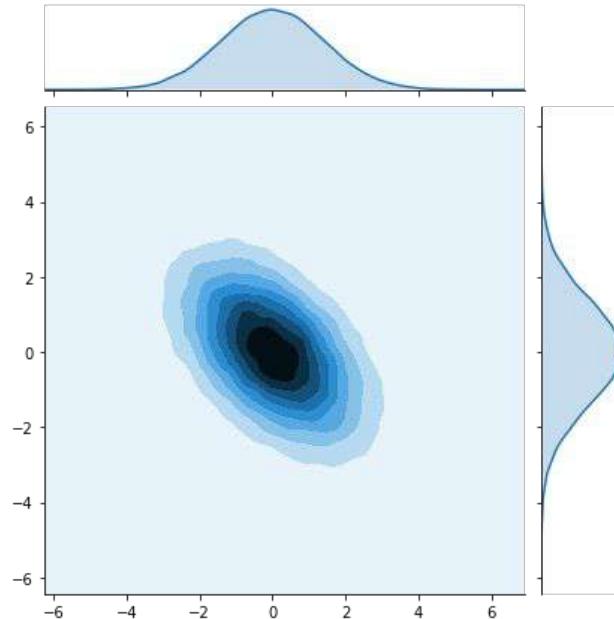
Covariance
matrix

Multivariate Normal Distributions

Non-0's not on the diagonal =
dimensions **covary**

$$\Sigma = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}$$

Covariance
matrix



Multivariate Normal Fréchet Distance

Univariate Normal Fréchet Distance =

$$(\mu_X - \mu_Y)^2 + (\sigma_X - \sigma_Y)^2$$

Multivariate Normal Fréchet Distance =

$$\|\mu_X - \mu_Y\|^2 + \text{Tr} \left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y} \right)$$

Multivariate Normal Fréchet Distance

Univariate Normal Fréchet Distance =

$$(\mu_X - \mu_Y)^2 + (\sigma_X - \sigma_Y)^2$$

Multivariate Normal Fréchet Distance =

$$\|\mu_X - \mu_Y\|^2 + \text{Tr} \left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y} \right)$$

Multivariate Normal Fréchet Distance

Univariate Normal Fréchet Distance =

$$(\mu_X - \mu_Y)^2 + (\sigma_X^2 + \sigma_Y^2 - 2\sigma_X\sigma_Y)$$

Multivariate Normal Fréchet Distance =

$$\|\mu_X - \mu_Y\|^2 + \text{Tr} \left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y} \right)$$

Multivariate Normal Fréchet Distance

Univariate Normal Fréchet Distance =

$$(\mu_X - \mu_Y)^2 + (\sigma_X^2 + \sigma_Y^2 - 2\sigma_X\sigma_Y)$$

Multivariate Normal Fréchet Distance =

$$\|\mu_X - \mu_Y\|^2 + \text{Tr} \left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y} \right)$$

Fréchet Inception Distance (FID)

FID =

$$\|\mu_X - \mu_Y\|^2 + \text{Tr} \left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y} \right)$$

Real and fake embeddings are two
multivariate normal distributions

Fréchet Inception Distance (FID)

FID =

$$\|\mu_X - \mu_Y\|^2 + \text{Tr} \left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y} \right)$$

Lower FID = closer
distributions

Real and fake embeddings are two
multivariate normal distributions

Fréchet Inception Distance (FID)

FID =

$$\|\mu_X - \mu_Y\|^2 + \text{Tr} \left(\Sigma_X + \Sigma_Y - 2\sqrt{\Sigma_X \Sigma_Y} \right)$$

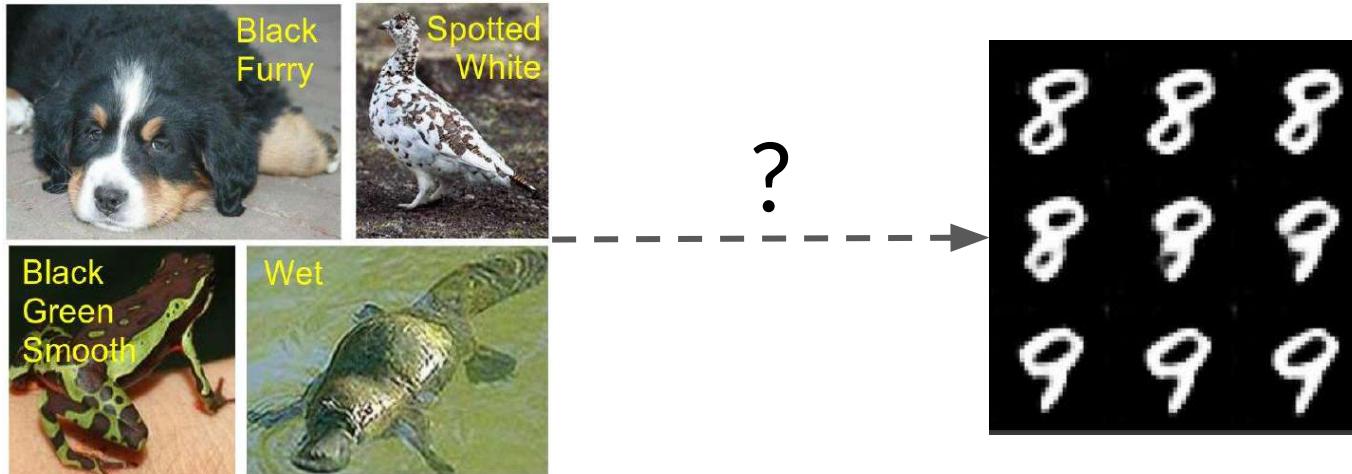
Lower FID = closer
distributions

Real and fake embeddings are two
multivariate normal distributions

Use *large sample size* to
reduce noise

Shortcomings of FID

- Uses pre-trained Inception model, which may not capture all features



© 2016
Stanford
Vision Lab

Shortcomings of FID

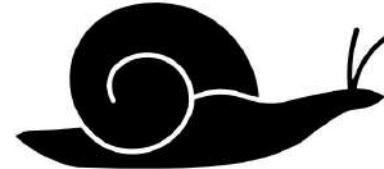
- Uses pre-trained Inception model, which may not capture all features
- Needs a large sample size



© 2016
Stanford
Vision Lab

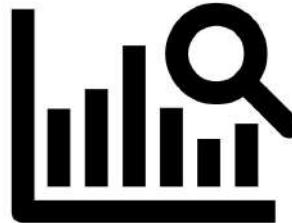
Shortcomings of FID

- Uses pre-trained Inception model, which may not capture all features
- Needs a large sample size
- Slow to run



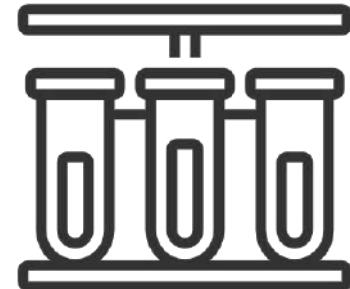
Shortcomings of FID

- Uses pre-trained Inception model, which may not capture all features
- Needs a large sample size
- Slow to run
- Limited statistics used: only mean and covariance



Summary

- FID calculates the difference between reals and fakes
- FID uses the Inception model and multivariate normal Fréchet distance
- Sample size needs to be large for FID to work well





deeplearning.ai

Inception Score

Outline

- Another evaluation metric: Inception Score (IS)
 - Intuition, notation, shortcomings



Inception Model Classification



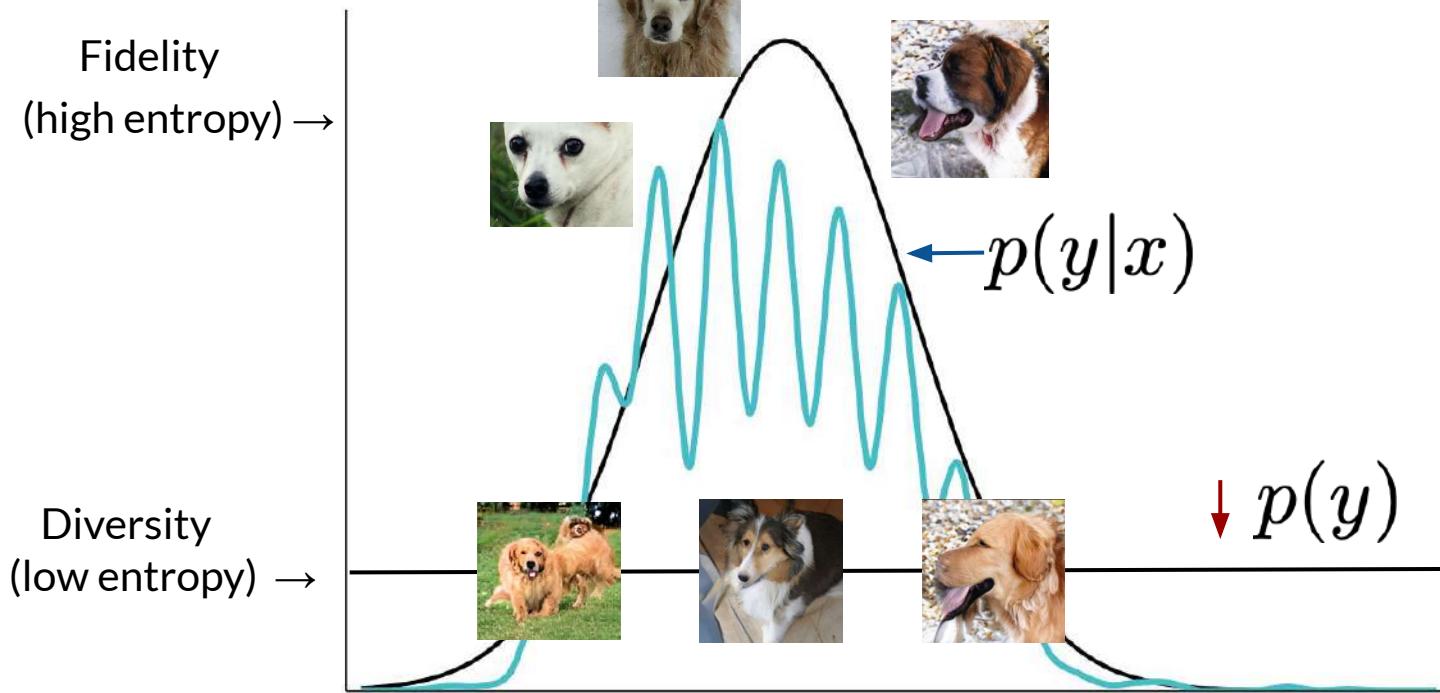
0.30 Cat

0.60 Dog

0.10 Bird

Fake

Entropy



KL Divergence

$$D_{KL}(p(y|x) \| p(y)) =$$

$$p(y|x) \log \left(\frac{p(y|x)}{p(y)} \right)$$

Conditional distribution
(fidelity)

Marginal distribution
(diversity)

Inception Score (IS)

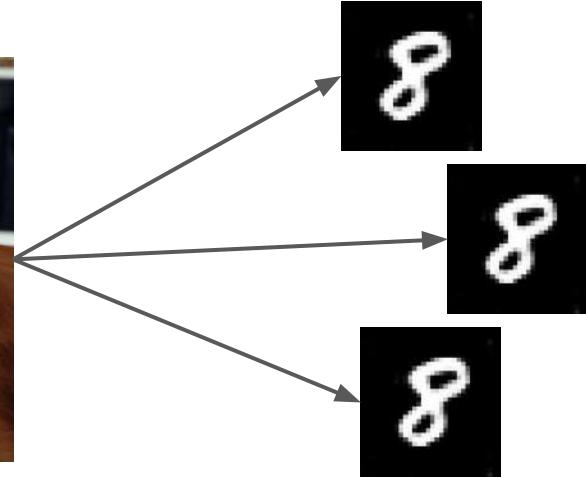
$$\text{IS} = \exp(\mathbb{E}_{x \sim p_\varepsilon} D_{KL}(p(y | x) || p(y)))$$

KL Divergence

Shortcomings of IS

- Can be exploited or gamed
 - Generate one realistic image of each class

*This is the
only 8 I make*



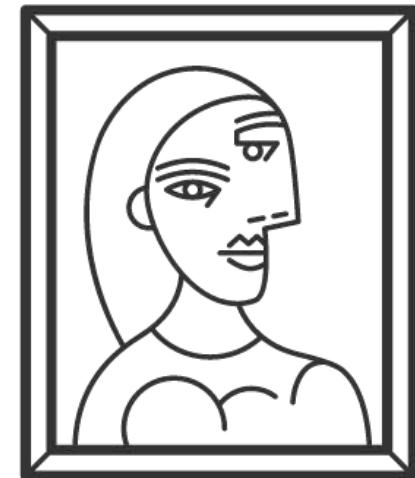
Shortcomings of IS

- Can be exploited or gamed
 - Generate one realistic image of each class
- Only looks at fake images
 - No comparison to real images

$$p(y|x) \quad p(y)$$

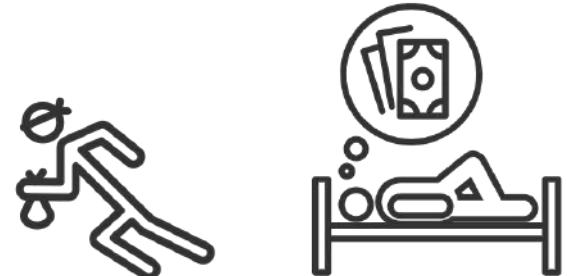
Shortcomings of IS

- Can be exploited or gamed
 - Generate one realistic image of each class
- Only looks at fake images
 - No comparison to real images
- Can miss useful features
 - ImageNet isn't everything



Summary

- Inception Score tries to capture fidelity & diversity
- Inception Score has many shortcomings
 - Can be gamed too easily
 - Only looks at fake images, not reals
 - ImageNet doesn't teach a model all features
- Worse than Fréchet Inception Distance





deeplearning.ai

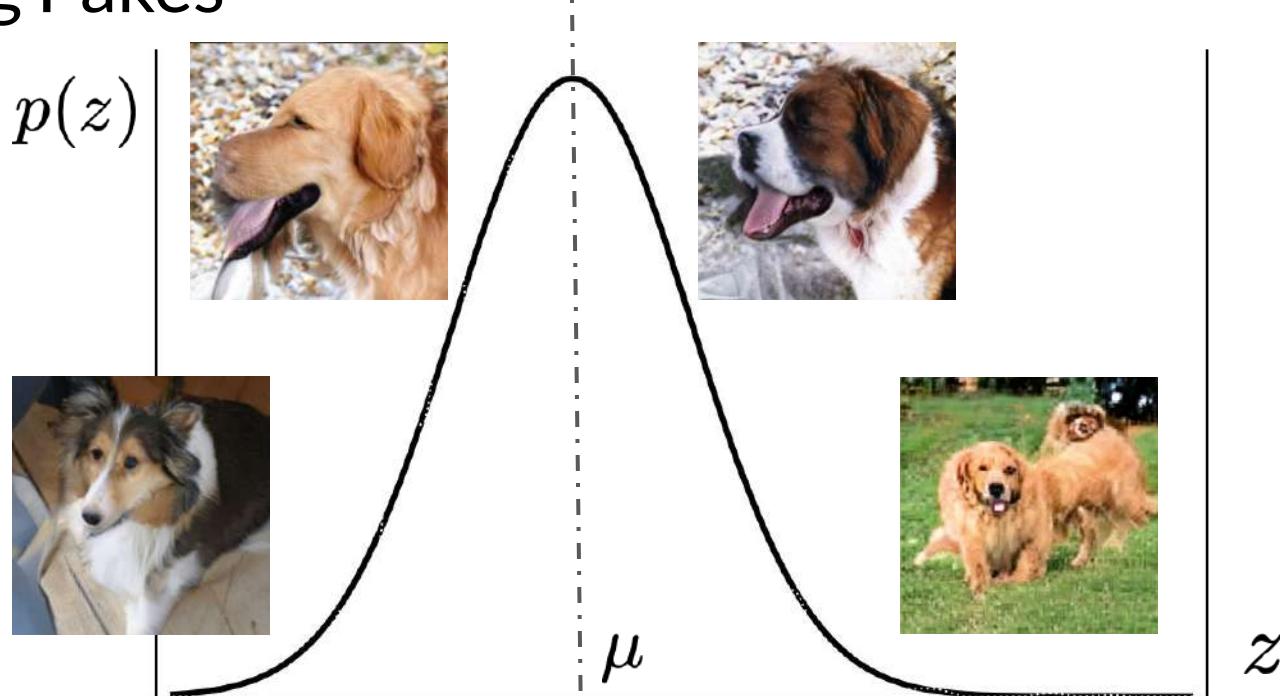
Sampling and Truncation

Outline

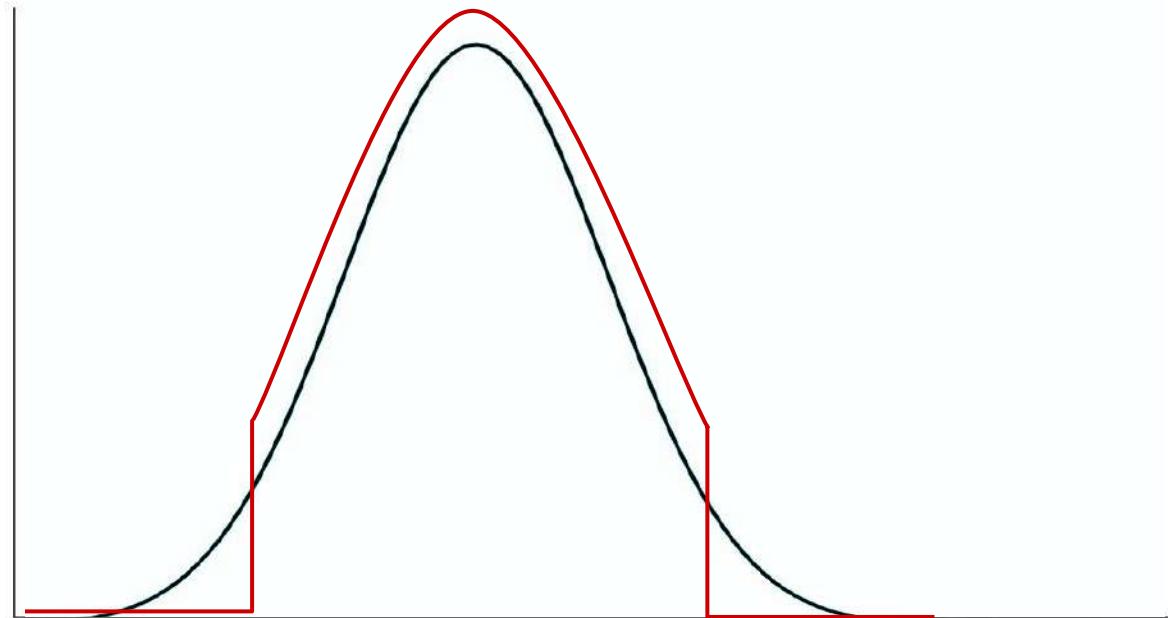
- Sampling reals vs. fakes
- The truncation trick
- HYPE!



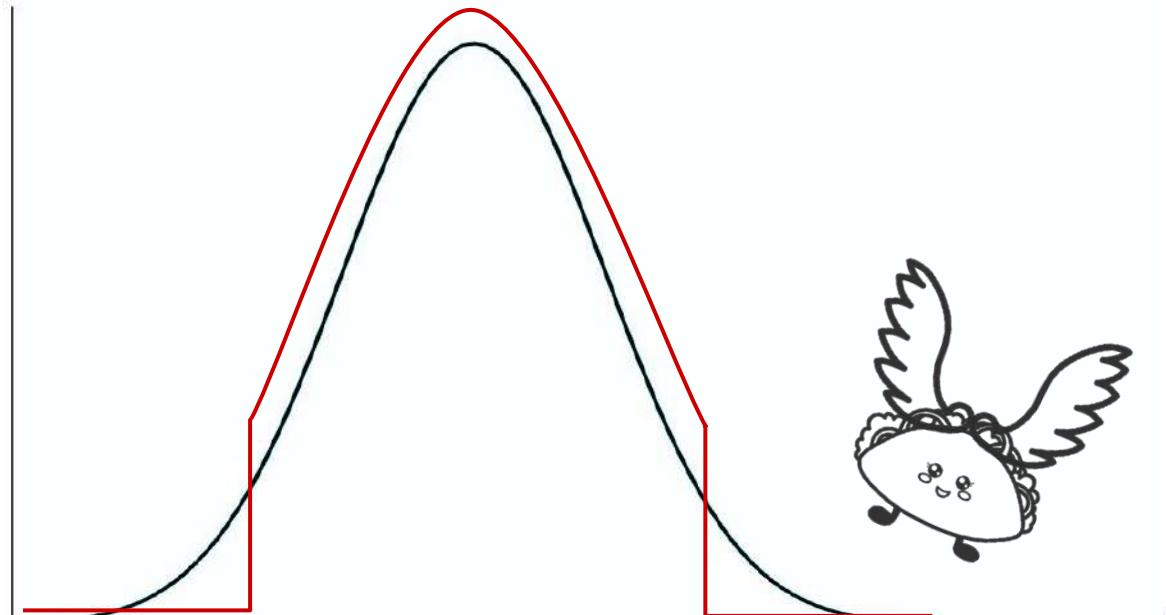
Sampling Fakes



Truncation Trick



Truncation Trick



HYPE and Human Evaluation

- Crowdsourced evaluation from Amazon Mechanical Turk
- $\text{HYPE}_{\text{time}}$ measures time-limited perceptual thresholds
- HYPE_{∞} measures error rate on a percentage of images
- Ultimately, evaluation depends on the type of downstream task



Available from: <https://arxiv.org/abs/1904.01121>

Summary

- Fakes are sampled using the training or prior distribution of z
- Truncate more for higher fidelity, lower diversity
- Human evaluation is still necessary for sampling



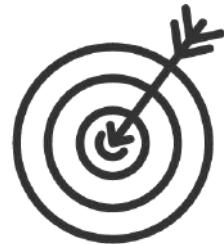


deeplearning.ai

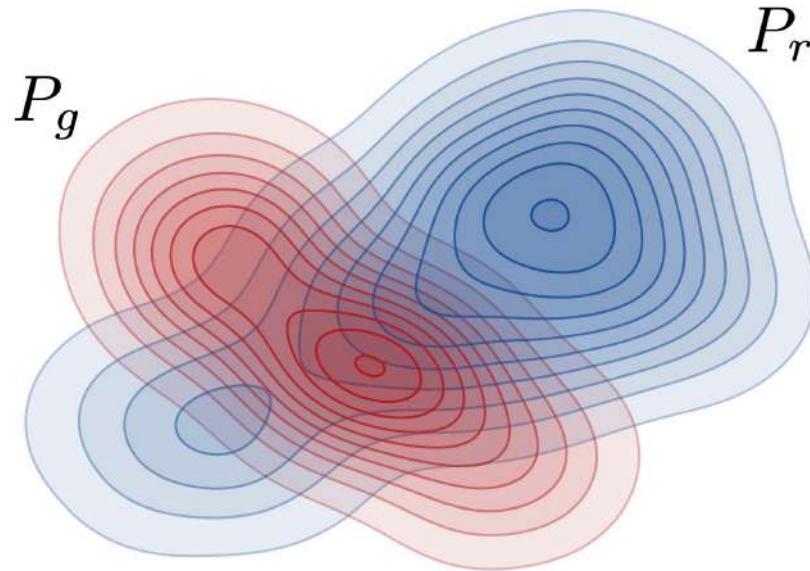
Precision and Recall

Outline

- Precision and recall in GANs evaluation
- Relating precision and recall to fidelity and diversity

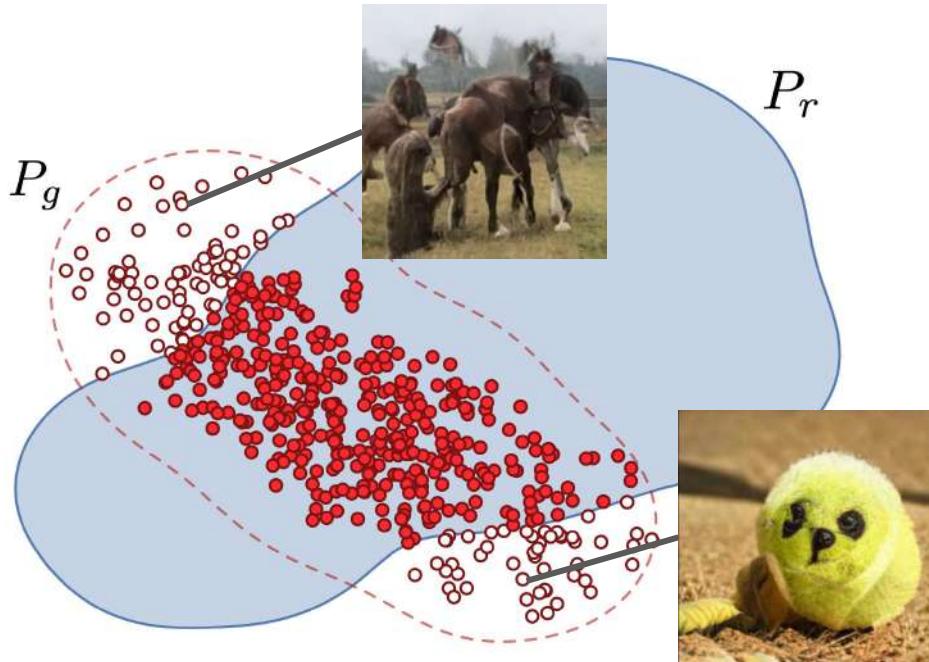


Precision and Recall



Available at: <https://arxiv.org/abs/1904.06991>

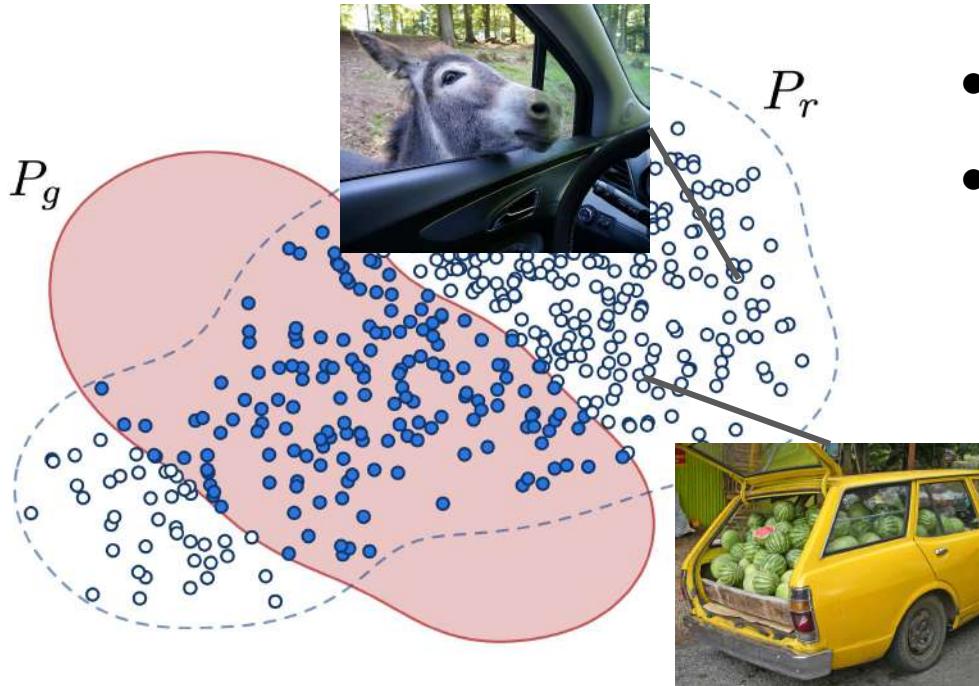
Precision



- Relates to fidelity
- Looks at overlap between reals and fakes, over how much extra gunk the generator produces (non-overlap red)

Diagram available at: <https://arxiv.org/abs/1904.06991>; Tennis dog available at: <https://arxiv.org/abs/1809.11096>

Recall

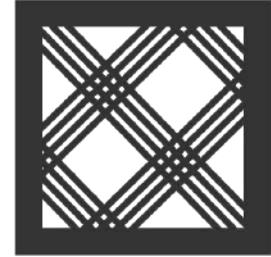


- Relates to diversity
- Looks at overlap between reals and fakes, over all the reals that the generator cannot model (non-overlap blue)

Diagram available at: <https://arxiv.org/abs/1904.06991>

Summary

- Precision is to fidelity as to recall is to diversity
- Models tend to be better at recall
- Use truncation trick to improve precision



Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>

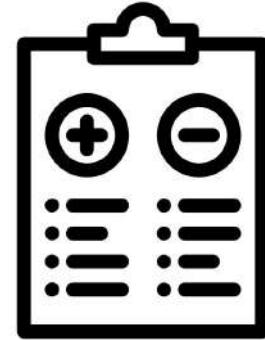


deeplearning.ai

Disadvantages of GANs

Outline

- Advantages of GANs
- Disadvantages of GANs



Advantages of GANs

- Amazing empirical results - especially with fidelity



Advantages of GANs

- Amazing empirical results - especially with fidelity
- Fast inference (image generation during testing)



Disadvantages of GANs

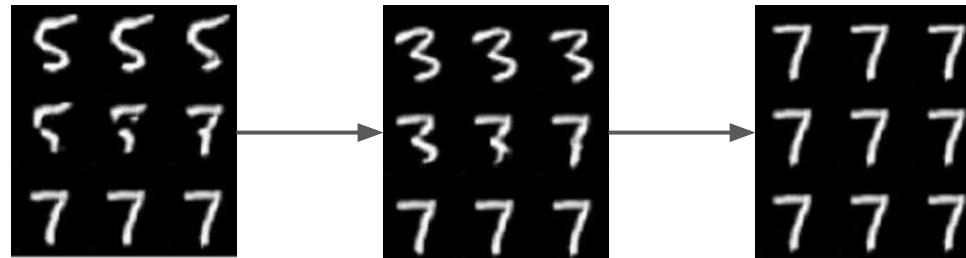
- Lack of intrinsic evaluation metrics



Looks pretty
real...?

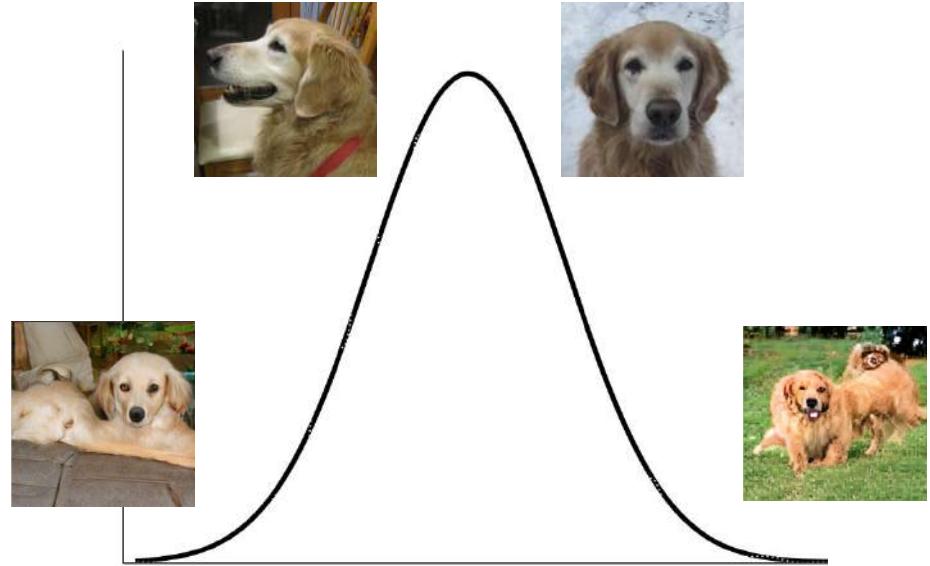
Disadvantages of GANs

- Lack of intrinsic evaluation metrics
- Unstable training



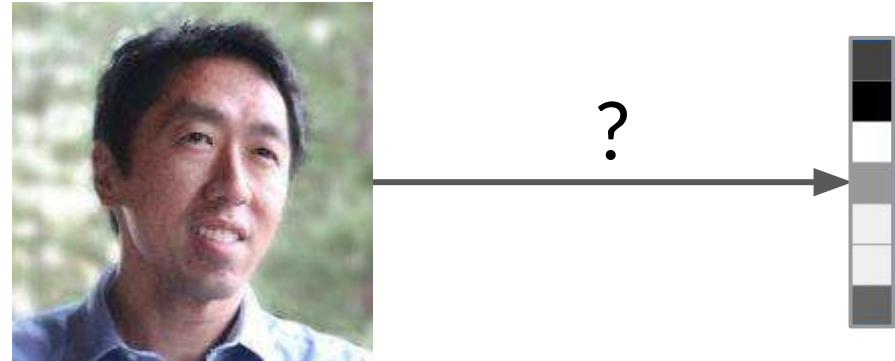
Disadvantages of GANs

- Lack of intrinsic evaluation metrics
- Unstable training
- No density estimation



Disadvantages of GANs

- Lack of intrinsic evaluation metrics
- Unstable training
- No density estimation
- Inverting is not straightforward



Summary

Advantages

- Amazing empirical results
- Fast inference

Disadvantages

- Lack of intrinsic evaluation metrics
- Unstable training
- No density estimation
- Inverting is not straightforward

Summary

Advantages

- Amazing empirical results
- Fast inference

GANs have **amazing results**,
but shortcomings as well.

Disadvantages

- Lack of intrinsic evaluation metrics
- Unstable training
- No density estimation
- Inverting is not straightforward

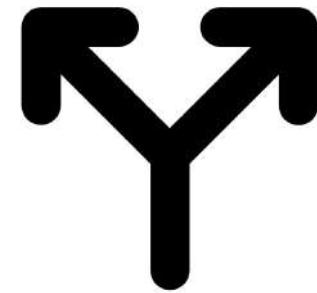


deeplearning.ai

Alternatives to GANs

Outline

- Overview of generative models
- VAEs and other alternatives



Generative Models

Noise Class Features

$$\xi, Y \rightarrow X$$

$$P(X|Y)$$

Generative Models

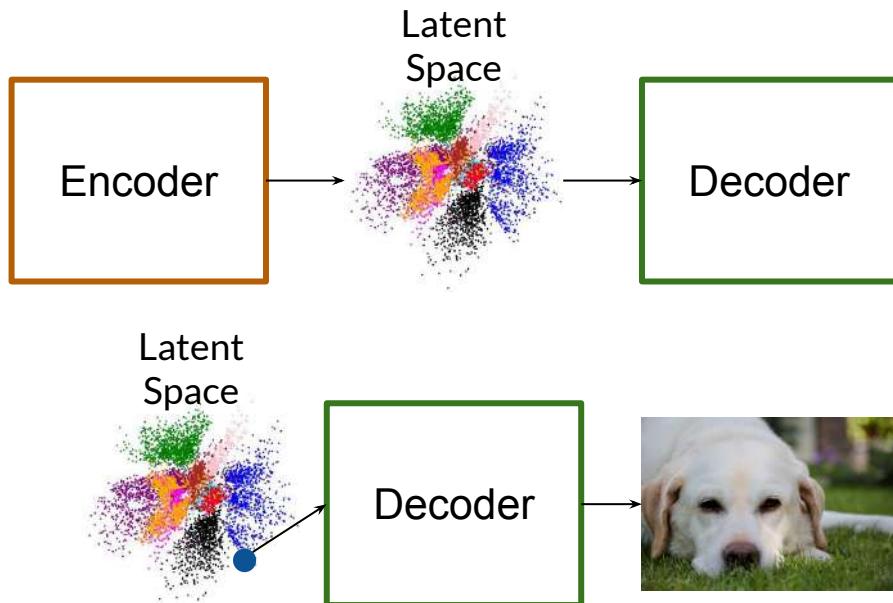
There are other generative models than just GANs!

Noise Class Features

$$\xi, Y \rightarrow X$$

$$P(X|Y)$$

Variational Autoencoders (VAEs)



Available from: <https://arxiv.org/abs/1804.00891>

Variational Autoencoders (VAEs)

Advantages

- Has density estimation
- Invertible
- Stable training

Disadvantages

- Lower quality results

Variational Autoencoders (VAEs)



VQ-VAE (Proposed)

BigGAN deep

Available from: <https://arxiv.org/abs/1906.00446>

Autoregressive Models



Left: source image **Right:** new portraits generated from high-level latent representation

Relies on previous pixels to
generate next pixel

Available from: <https://arxiv.org/abs/1606.05328>

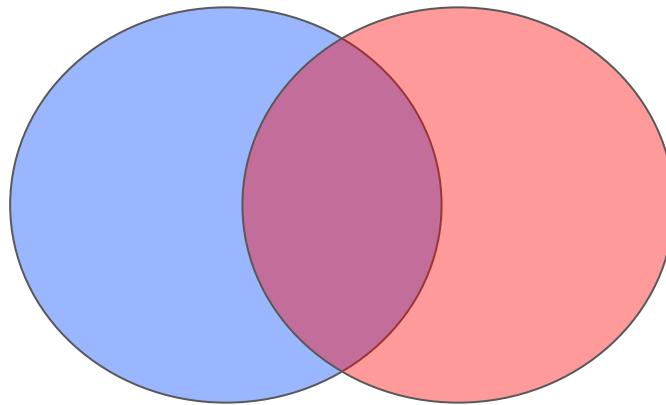
Flow Models



Uses invertible
mappings

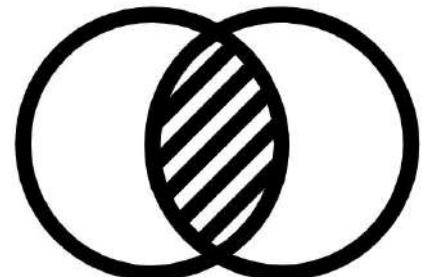
Available from: <https://openai.com/blog/glow/>

Hybrid Models



Summary

- VAEs have the opposite pros/cons as GANs
 - Often lower fidelity results
 - Density estimation, inversion, stable training
- Other alternative generative models:
 - Autoregressive models
 - Flow models
 - Hybrid models





deeplearning.ai

Intro to Machine Bias

Outline

- *Machine Bias* (ProPublica)
- Racial disparity in AI for risk assessments
- Impacts of biased AI



Machine Bias



Bernard Parker, left, was rated high risk; Dylan Fugett was rated low risk. (Josh Ritchie for ProPublica)

Machine Bias

There's software used across the country to predict future criminals. And it's biased against blacks.

by Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, ProPublica
May 23, 2016

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

Machine Bias

Risk assessment
= likelihood of
committing a
crime in the
future



Bernard Parker, left, was rated high risk; Dylan Fugett was rated low risk. (Josh Ritchie for ProPublica)

Machine Bias

There's software used across the country to predict future criminals. And it's biased against blacks.

by Julia Angwin, Jeff Larson, Surya Mattu and Lauren Kirchner, ProPublica

May 23, 2016

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

COMPAS Algorithm

- One of two leading commercial tools used by the legal system

COMPAS Algorithm

- One of two leading commercial tools used by the legal system
- Used in pretrial hearings and criminal sentencing to assess risk of re-offense (recidivism)

COMPAS Algorithm

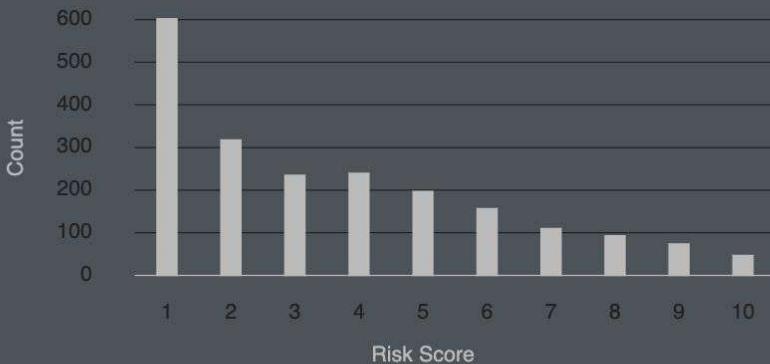
- One of two leading commercial tools used by the legal system
- Used in pretrial hearings and criminal sentencing to assess risk of re-offense (recidivism)
- Score based on proprietary calculations
 - Not available to the public
 - Unvalidated

COMPAS Algorithm

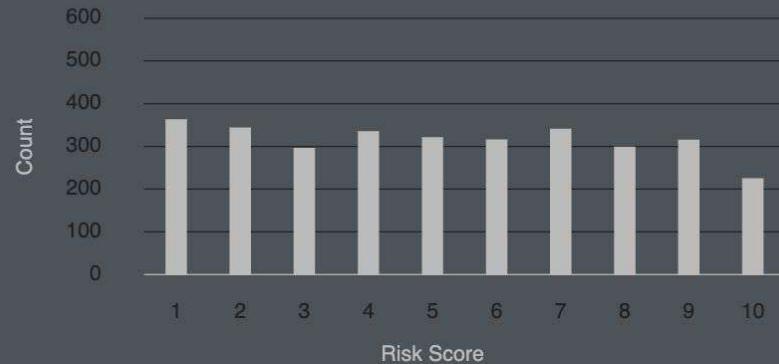
- One of two leading commercial tools used by the legal system
- Used in pretrial hearings and criminal sentencing to assess risk of re-offense (recidivism)
- Score based on proprietary calculations
 - Not available to the public
 - Unvalidated
- Predicts recurrence of violent crime correctly only 20% of the time

Biased Risk Assessment

White Defendants' Risk Scores



Black Defendants' Risk Scores

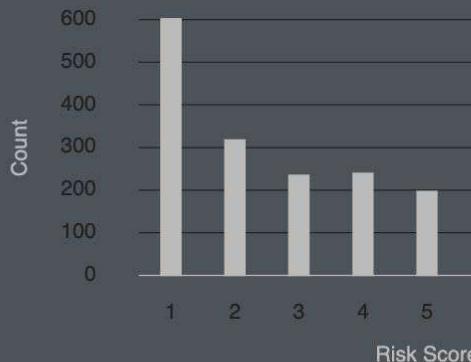


These charts show that scores for white defendants were skewed toward lower-risk categories. Scores for black defendants were not. (Source: ProPublica analysis of data from Broward County, Fla.)

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

Biased Risk Assessment

White Defendants' Risk Scores



Two DUI Arrests

GREGORY LUGO

Prior Offenses

3 DUIs, 1 battery

Subsequent Offenses

1 domestic violence
battery

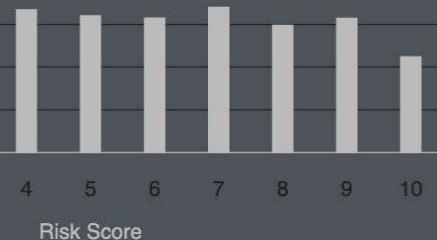
MALLORY WILLIAMS

Prior Offenses

2 misdemeanors

Subsequent Offenses

None



These charts show that scores were not. (Source: ProPublica)

LOW RISK

1

MEDIUM RISK

6

Scores for black defendants

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

Consequences of a Higher Score

Paul
Zilly



Plea deal overturned and sentenced to two years in state prison.

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

Consequences of a Higher Score

Paul
Zilly



Plea deal overturned and sentenced to two years in state prison.

“Had I not had the COMPAS, I believe it would likely be that ***I would have given one year, six months***”

- Appeals judge

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

Consequences of a Higher Score

Sade
Jones



Bond was raised from the recommended \$0 to \$1000

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

Consequences of a Higher Score

Sade
Jones



Bond was raised from the recommended \$0 to \$1000

“I went to McDonald’s and a dollar store, and they all said no ***because of my background***”

- Jones

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

Prediction Failure

Prediction Fails Differently for Black Defendants

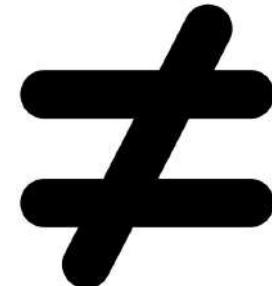
	WHITE	AFRICAN AMERICAN
Labeled Higher Risk, But Didn't Re-Offend	23.5%	44.9%
Labeled Lower Risk, Yet Did Re-Offend	47.7%	28.0%

Overall, Northpointe's assessment tool correctly predicts recidivism 61 percent of the time. But blacks are almost twice as likely as whites to be labeled a higher risk but not actually re-offend. It makes the opposite mistake among whites: They are much more likely than blacks to be labeled lower risk but go on to commit other crimes. (Source: ProPublica analysis of data from Broward County, Fla.)

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

Summary

- Machine learning bias has a disproportionately negative effect on historically underserved populations
- Proprietary risk assessment software:
 - Difficult to validate
 - Misses important considerations about people



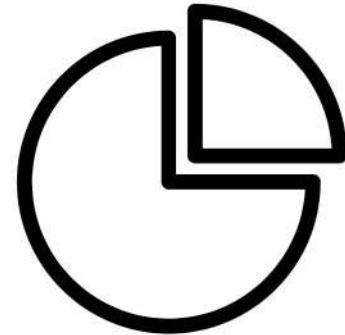


deeplearning.ai

Defining Fairness

Outline

- What is fairness?
- Complexity of defining fairness



Fairness in Machine Learning

Reading 1: Fairness Definitions
Explained

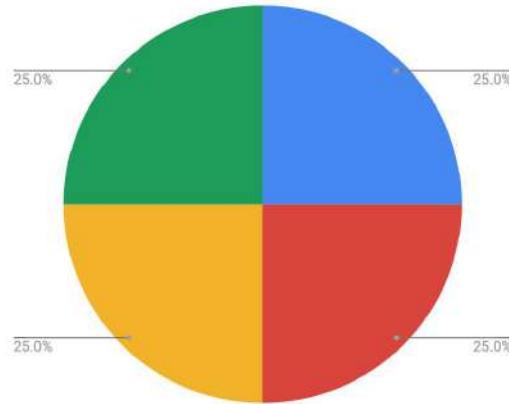
Reading 2: A Survey on Bias and
Fairness in Machine Learning

	Definition	Paper	Citation #	Result
3.1.1	Group fairness or statistical parity	[12]	208	✗
3.1.2	Conditional statistical parity	[11]	29	✓
3.2.1	Predictive parity	[10]	57	✓
3.2.2	False positive error rate balance	[10]	57	✗
3.2.3	False negative error rate balance	[10]	57	✓
3.2.4	Equalised odds	[14]	106	✗
3.2.5	Conditional use accuracy equality	[8]	18	✗
3.2.6	Overall accuracy equality	[8]	18	✓
3.2.7	Treatment equality	[8]	18	✗
3.3.1	Test-fairness or calibration	[10]	57	✓
3.3.2	Well calibration	[16]	81	✓
3.3.3	Balance for positive class	[16]	81	✓
3.3.4	Balance for negative class	[16]	81	✗
4.1	Causal discrimination	[13]	1	✗
4.2	Fairness through unawareness	[17]	14	✓
4.3	Fairness through awareness	[12]	208	✗
5.1	Counterfactual fairness	[17]	14	-
5.2	No unresolved discrimination	[15]	14	-
5.3	No proxy discrimination	[15]	14	-
5.4	Fair inference	[19]	6	-

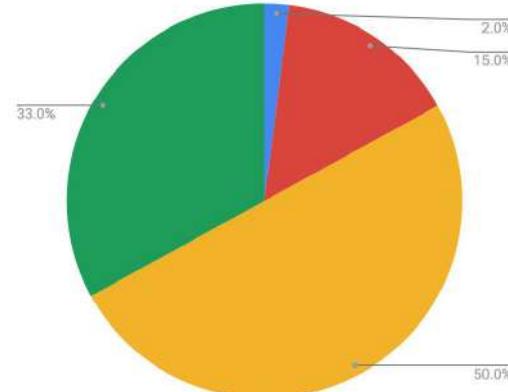
Table 1: Considered Definitions of Fairness

Available from: <https://fairware.cs.umass.edu/papers/Verma.pdf>

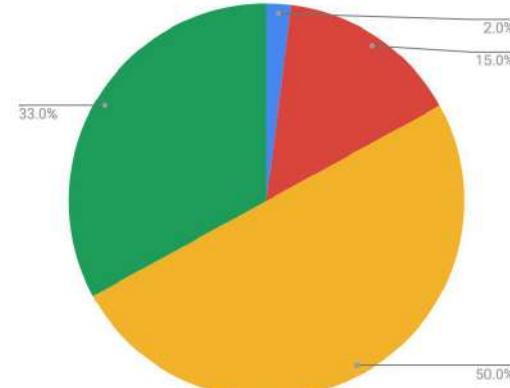
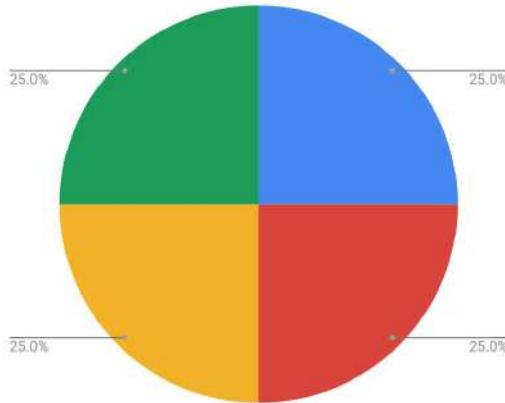
Defining Fairness



Defining Fairness



Defining Fairness

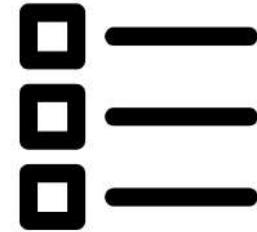


	WHITE	AFRICAN AMERICAN
Labeled Higher Risk, But Didn't Re-Offend	23.5%	44.9%
Labeled Lower Risk, Yet Did Re-Offend	47.7%	28.0%

Available from: <https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing>

Summary

- Fairness is difficult to define
- There is no single definition of fairness
- Important to explore these before releasing a system into production



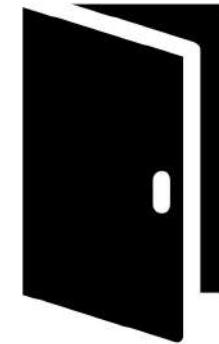


deeplearning.ai

Ways Bias is Introduced

Outline

- A few ways bias can enter a model
- PULSE: A case study with a biased GAN



Training Bias

Training data

- **No variation** in who or what is represented



Training Bias

Training data

- **No variation** in who or what is represented
- Bias in **collection methods**



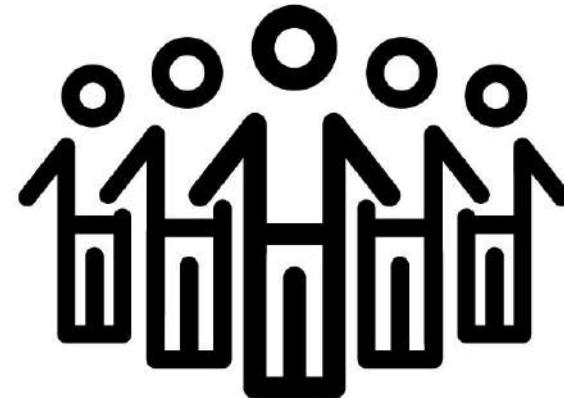
Training Bias

Training data

- **No variation** in who or what is represented
- Bias in **collection methods**

Data labelling

- **Diversity** of the labellers



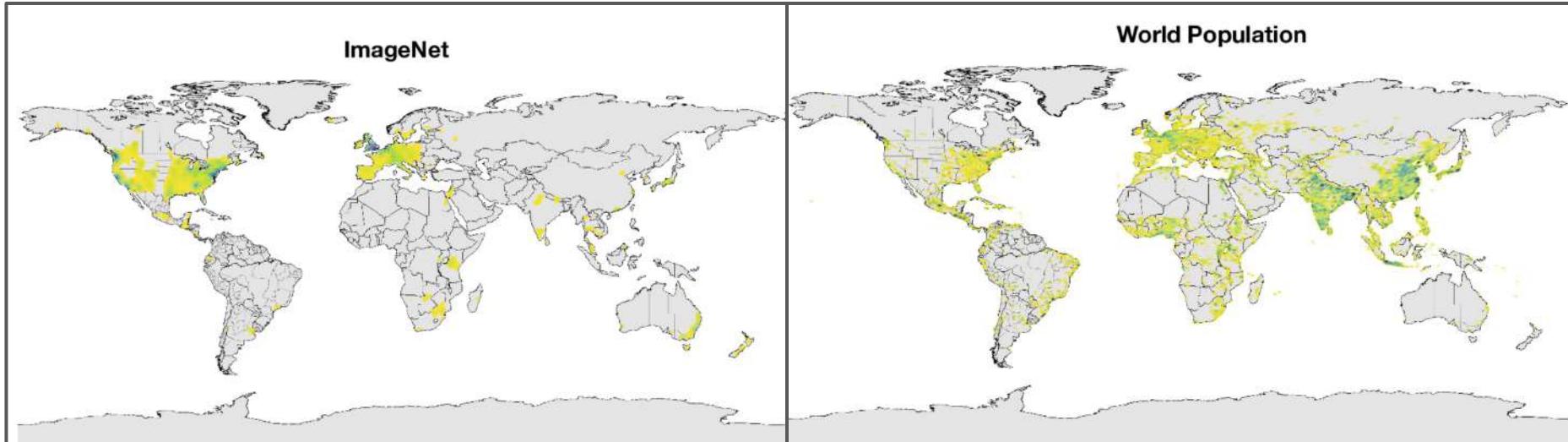
Evaluation Bias

- Images can be biased to reflect “correctness” in the dominant culture



Evaluation Bias

- Images can be biased to reflect “correctness” in the dominant culture



Available from: <https://arxiv.org/abs/1906.02659>

Evaluation Bias

- Images can be biased to reflect “correctness” in the dominant culture



Available from: <https://arxiv.org/abs/1906.02659>

Evaluation Bias

- Images can be biased to reflect “correctness” in the dominant culture

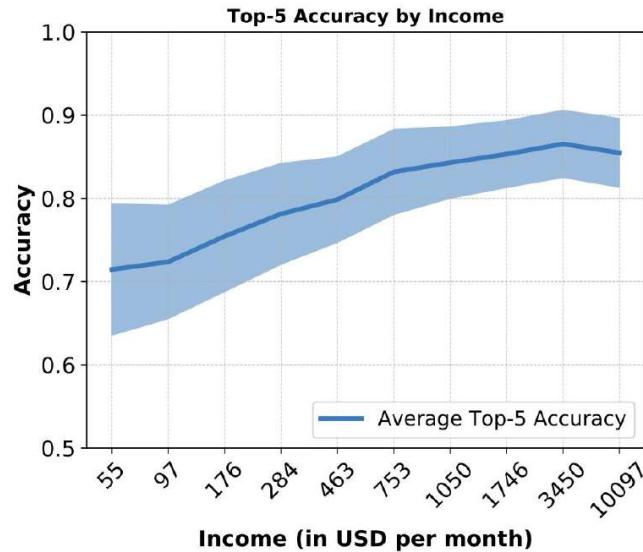
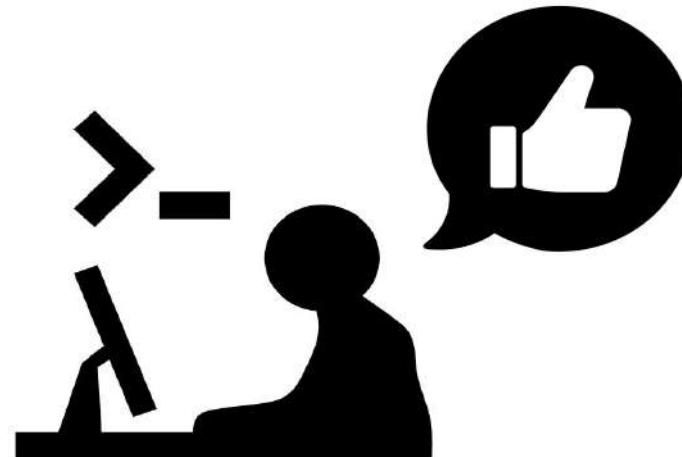


Figure 3: Average accuracy (and standard deviation) of six object-recognition systems as a function of the normalized consumption income of the household in which the image was collected (in US\$ per month).

Available from: <https://arxiv.org/abs/1906.02659>

Model Architecture Bias

- Can be influenced by the coders who designed the architecture or optimized the code



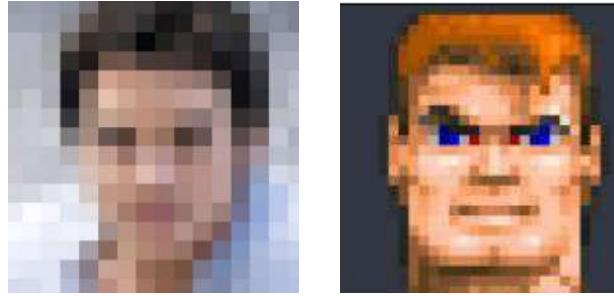
Other Avenues for Bias Introduction

Bias can appear at any step:

- Research
- Design
- Engineering
- Anywhere a person was involved

PULSE

Pixelated



“Upsampled”



(Left) Available from: <https://arxiv.org/abs/2003.03808>

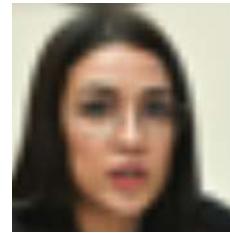
(Right) Available from: <https://www.theverge.com/21298762/face-depixelizer-ai-machine-learning-tool-pulse-stylegan-obama-bias>

PULSE

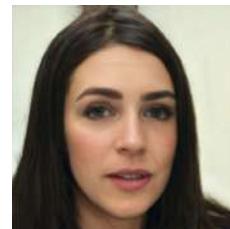
Ground
Truth



Pixelated



“Upsampled”



Available from: <https://www.theverge.com/21298762/face-depixelizer-ai-machine-learning-tool-pulse-stylegan-obama-bias>

Summary

- Bias can be introduced into a model at each step of the process
- Awareness and mitigation of bias is vital to responsible use of AI and, especially, state-of-the-art GANs



Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>



deeplearning.ai

GAN Improvements

Outline

- How GANs have improved
- State of the art methods for improving GANs performance



GANs Over Time



Ian Goodfellow
@goodfellow_ian

▼

4.5 years of GAN progress on face generation.

arxiv.org/abs/1406.2661 arxiv.org/abs/1511.06434

arxiv.org/abs/1606.07536 arxiv.org/abs/1710.10196

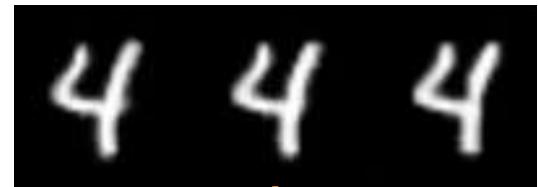
arxiv.org/abs/1812.04948



Main Improvements: (1) Stability



High standard deviation



Low standard deviation

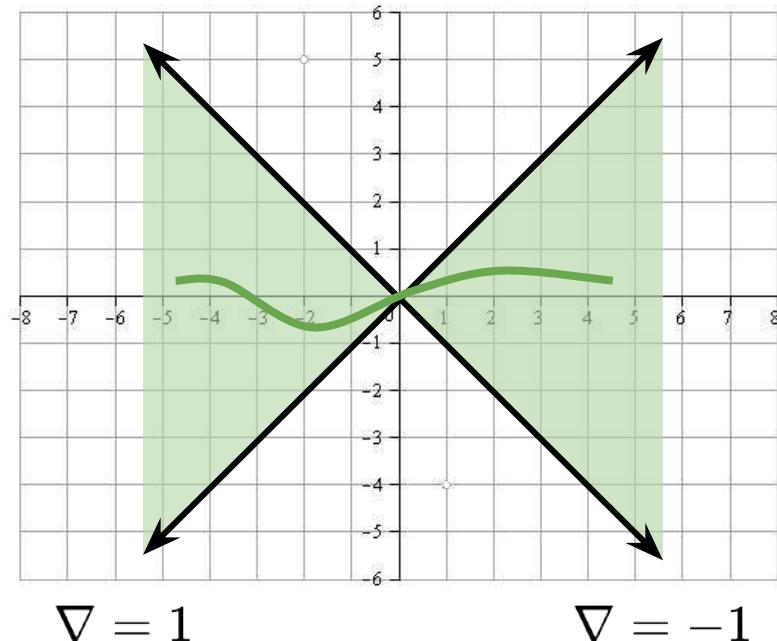
Use batch standard deviation to encourage diversity

Main Improvements: (1) Stability

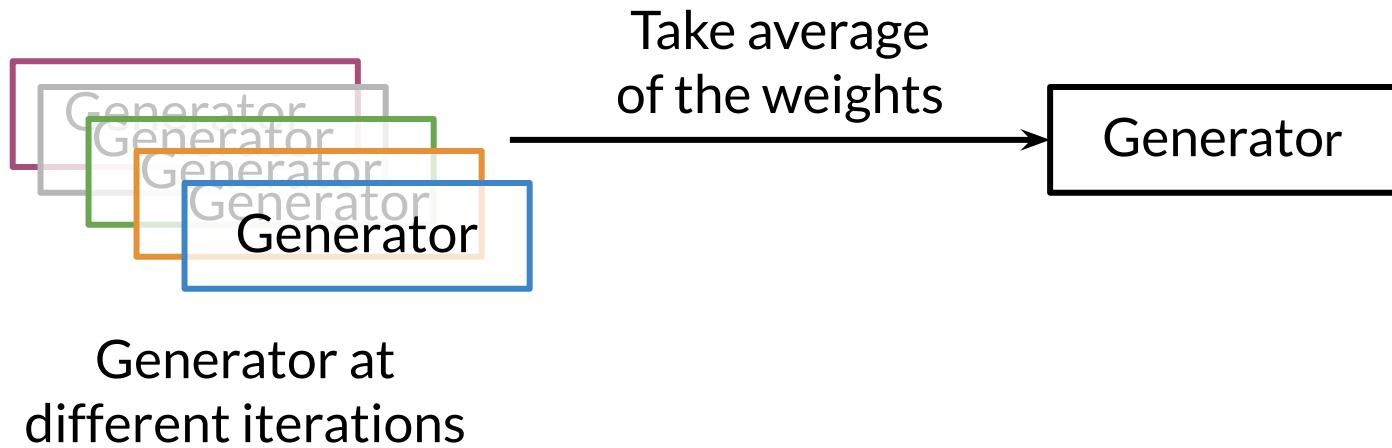
Improve stability by enforcing
1-Lipschitz continuity

E.g. **WGAN-GP** and **Spectral
Normalization**

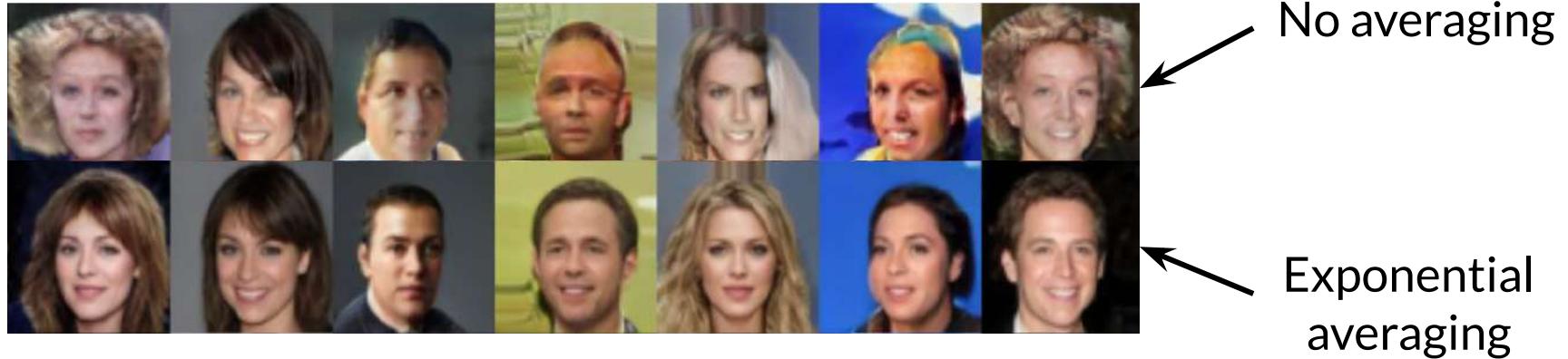
∇ : gradient



Main Improvements: (1) Stability



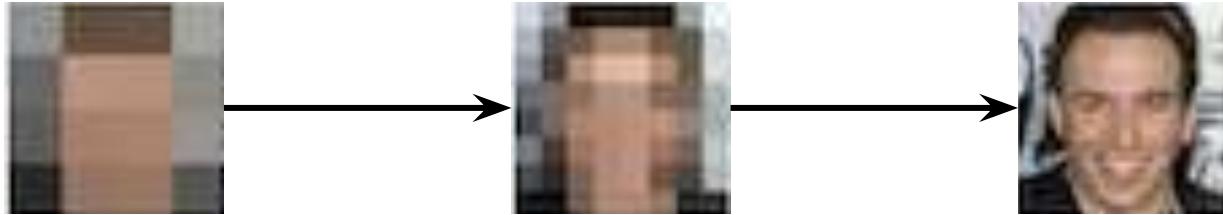
Main Improvements: (1) Stability



Use moving average for
smoother results

Available from: <https://arxiv.org/abs/1806.04498v2>

Main Improvements: (1) Stability



Progressive growing gradually trains
GAN at increasing resolutions

Available from: <https://arxiv.org/abs/1710.10196>

Main Improvements: (2) Capacity



Larger models can use
higher resolution images

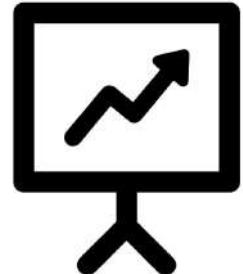
Main Improvements: (3) Diversity



Available from: <https://github.com/NVlabs/stylegan>

Summary

- GANs have improved because of:
 - Stability - longer training and better images
 - Capacity - larger models and higher resolution images
 - Diversity - increasing variety in generated images





deeplearning.ai

StyleGAN Overview

Outline

- StyleGAN achievements
- What styles are
- Introduction to StyleGAN architecture and components



StyleGAN Goals

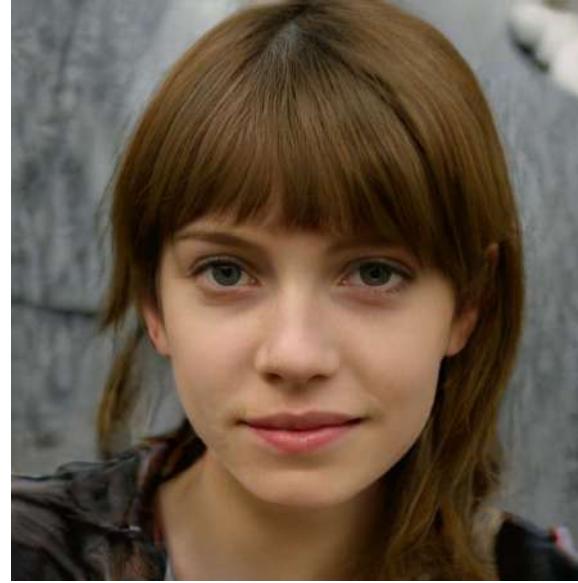
1. Greater **fidelity** on high-resolution images
2. Increased *diversity* of outputs
3. More control over image features



Greater Fidelity



Not fooling anyone



I'm shook

(Left) Available from: <https://arxiv.org/abs/1406.2661>

(Right) Available from: <https://github.com/NVlabs/stylegan>

Increased Diversity



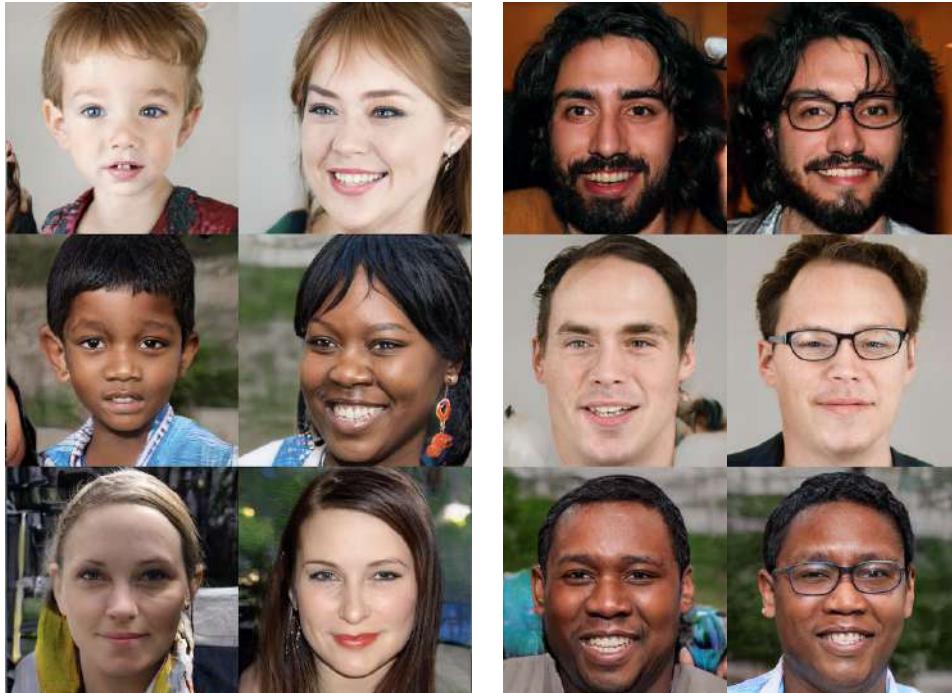
Available from: <https://arxiv.org/abs/1812.04948>

Increased Diversity



More Feature Control

Hair color/style →



← Glasses

Available from: <https://arxiv.org/abs/1812.04948>

Style in GANs

Style = variation in an image

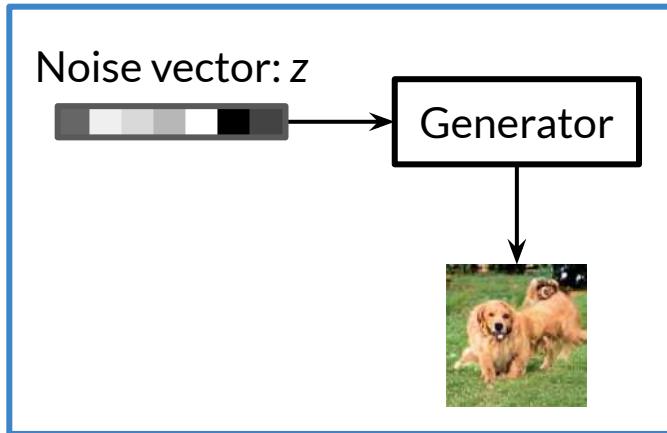
Early styles are **coarser** like face shape

Later styles are **finer** like hair wisps



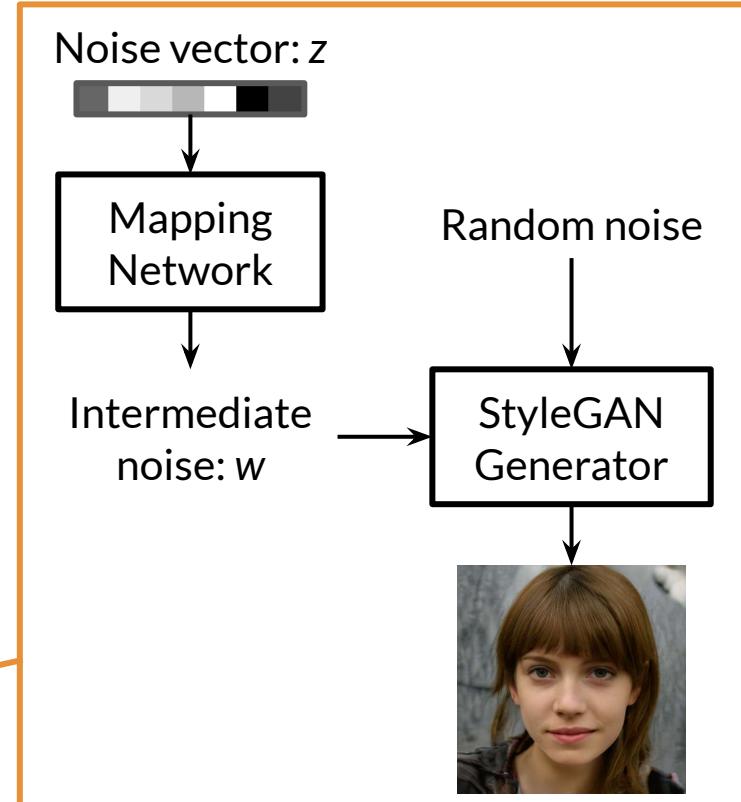
Available from: <https://arxiv.org/abs/1812.04948>

The Style-Based Generator

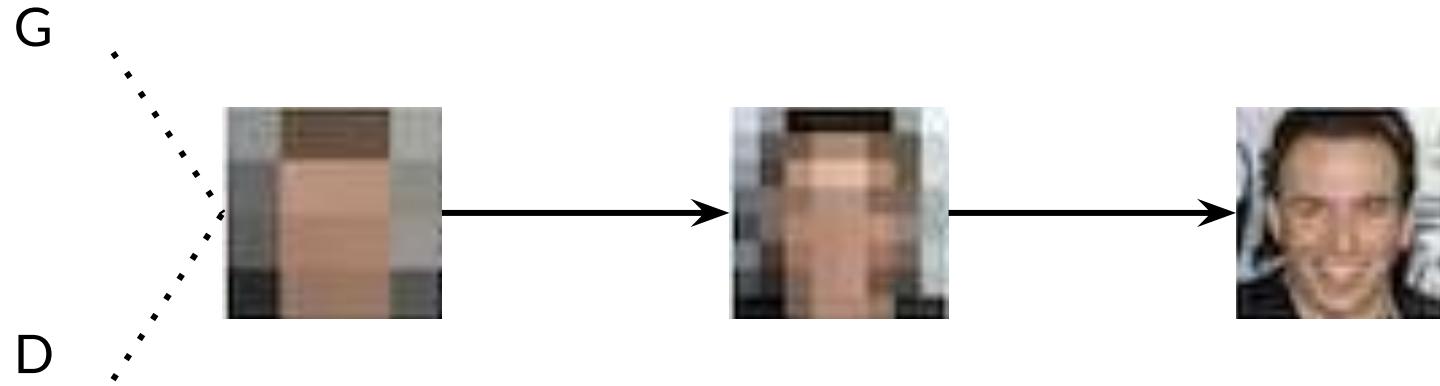


Traditional architecture

StyleGAN architecture



Progressive Growing



Available from: <https://arxiv.org/abs/1710.10196>

Summary

- StyleGAN's **goals**:
 - Greater fidelity, increased diversity, improved control over features
- Style is any variation in the image
- **Main components** of StyleGAN:
 - Progressive growing
 - Noise mapping network
 - Adaptive instance normalization (AdaIN)



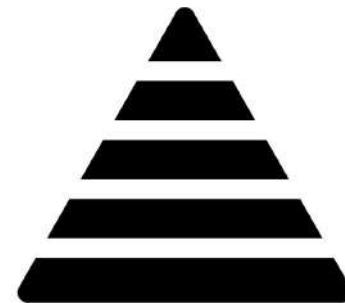


deeplearning.ai

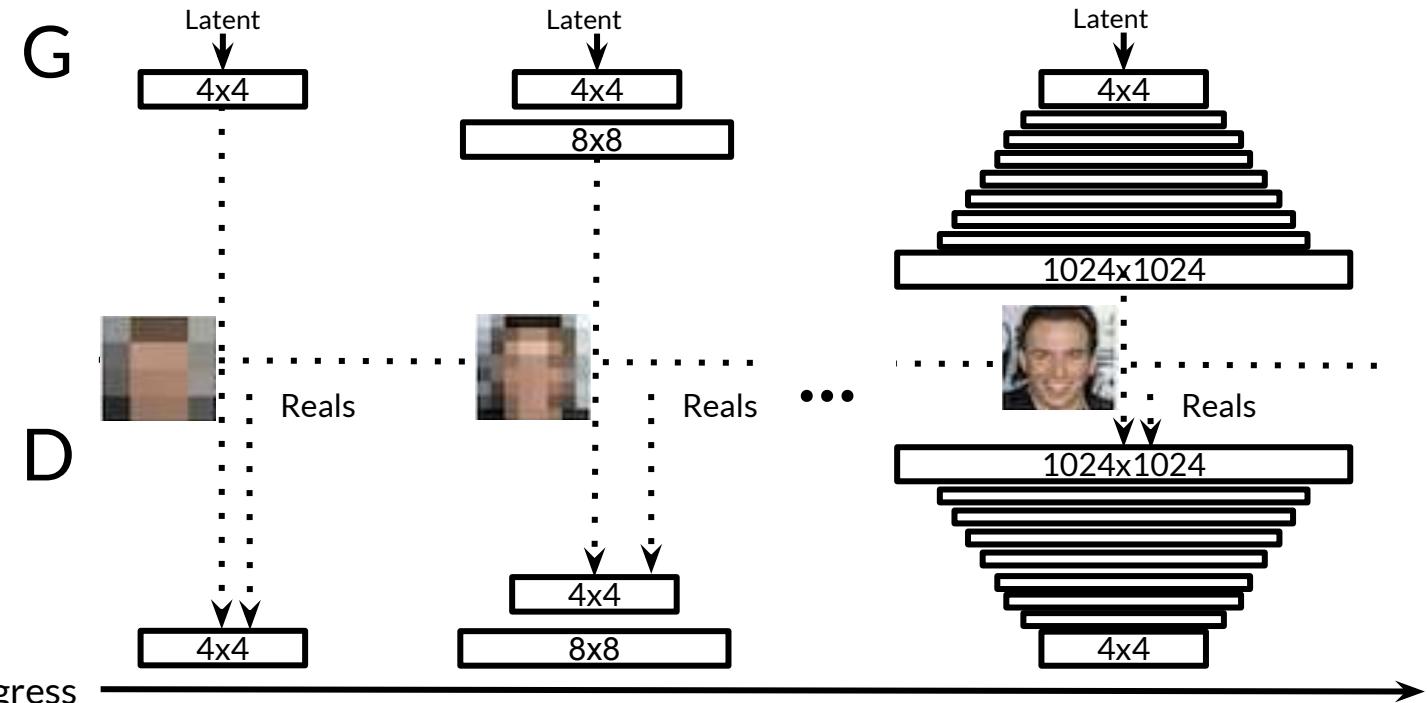
Progressive Growing

Outline

- Progressive growing intuition and motivation
- How to implement it



Progressive Growing



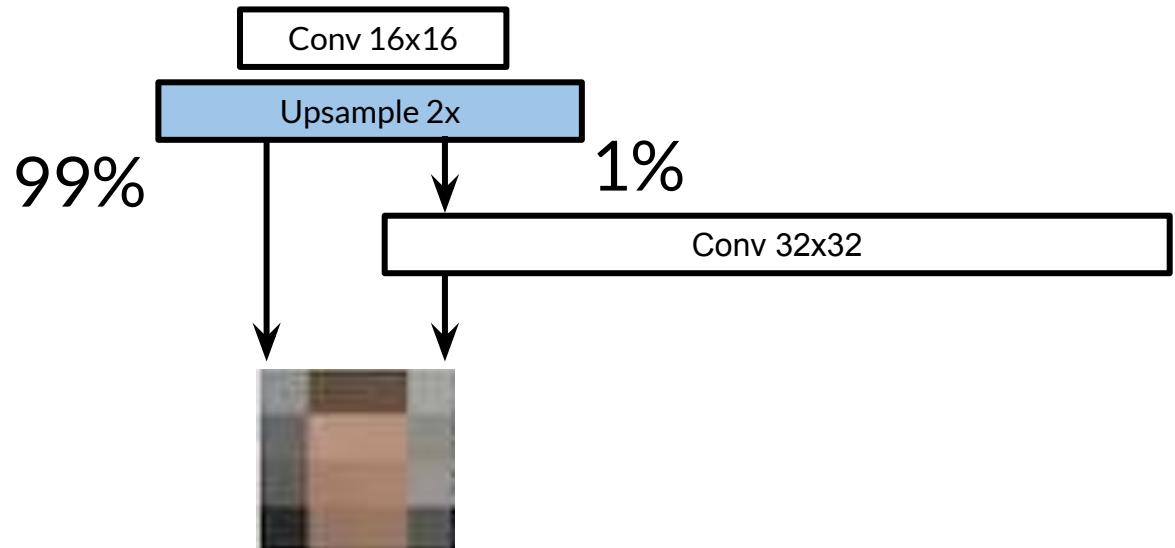
Based on: <https://arxiv.org/abs/1710.10196>

Progressive Growing in Action



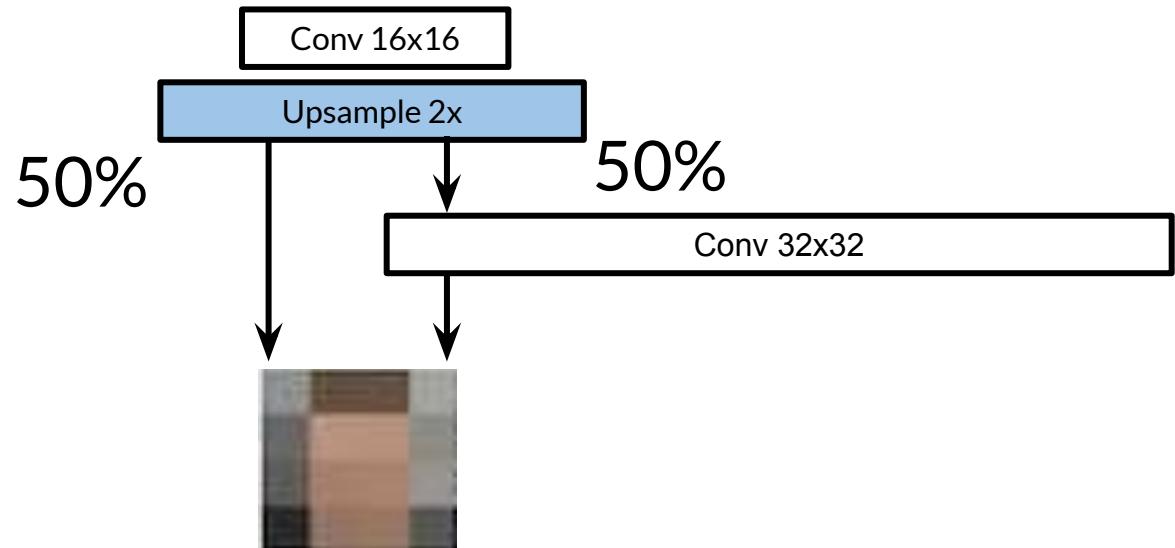
Available from: <https://www.gwern.net/images/gan/2019-03-16-stylegan-facestraining.mp4>

Progressive Growing: Generator



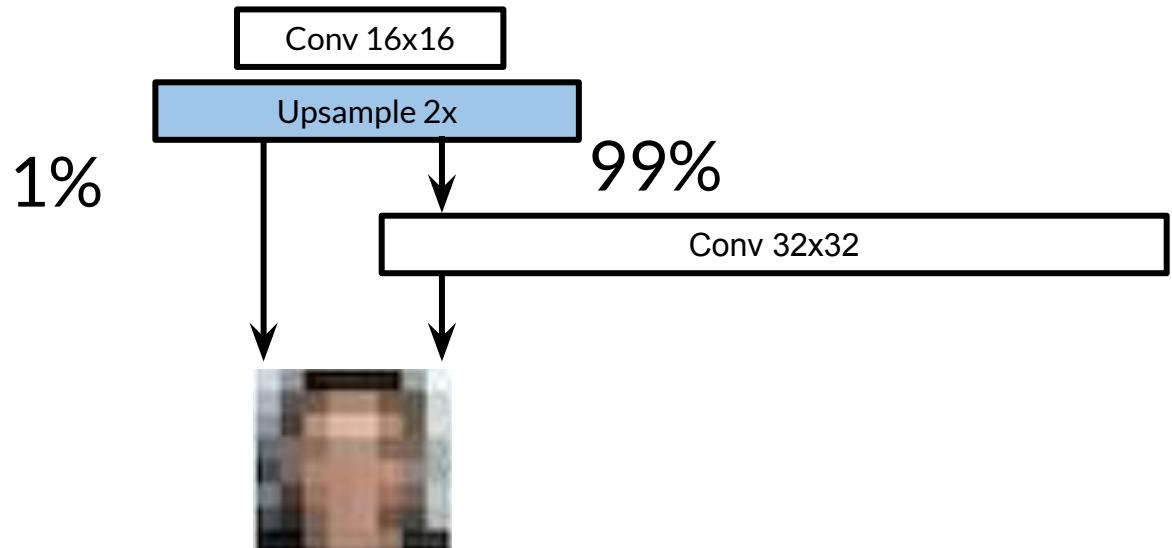
Based on: <https://arxiv.org/abs/1710.10196>

Progressive Growing: Generator



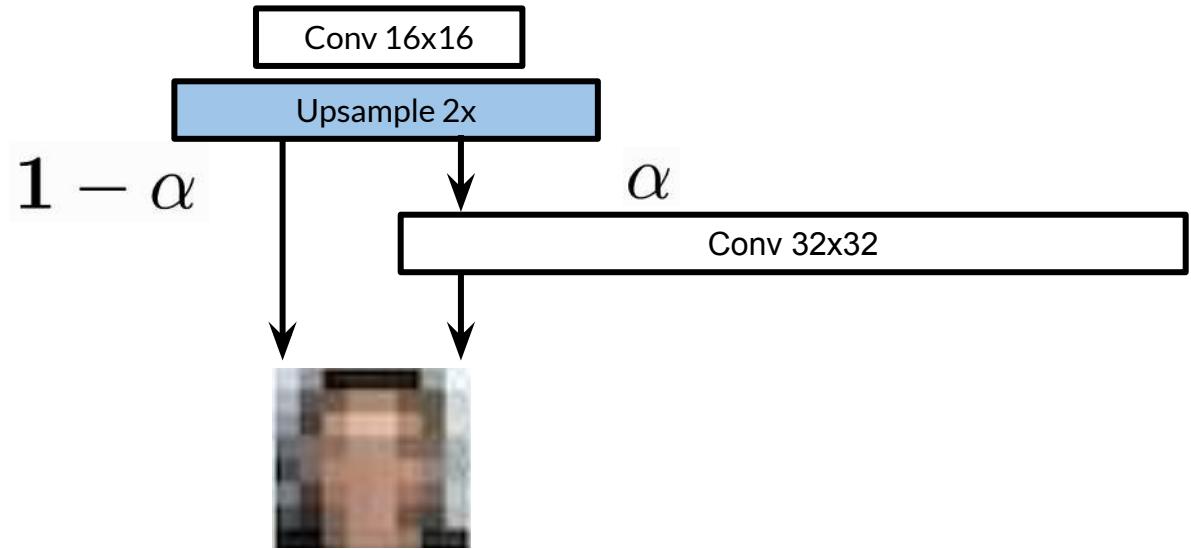
Based on: <https://arxiv.org/abs/1710.10196>

Progressive Growing: Generator



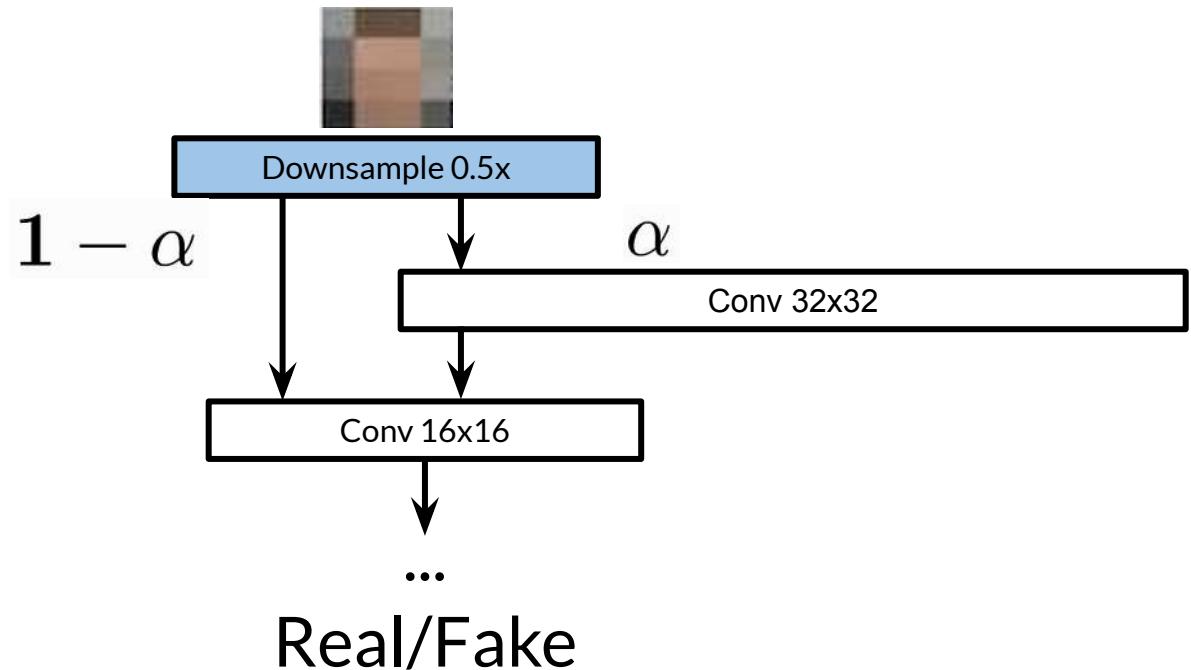
Based on: <https://arxiv.org/abs/1710.10196>

Progressive Growing: Generator



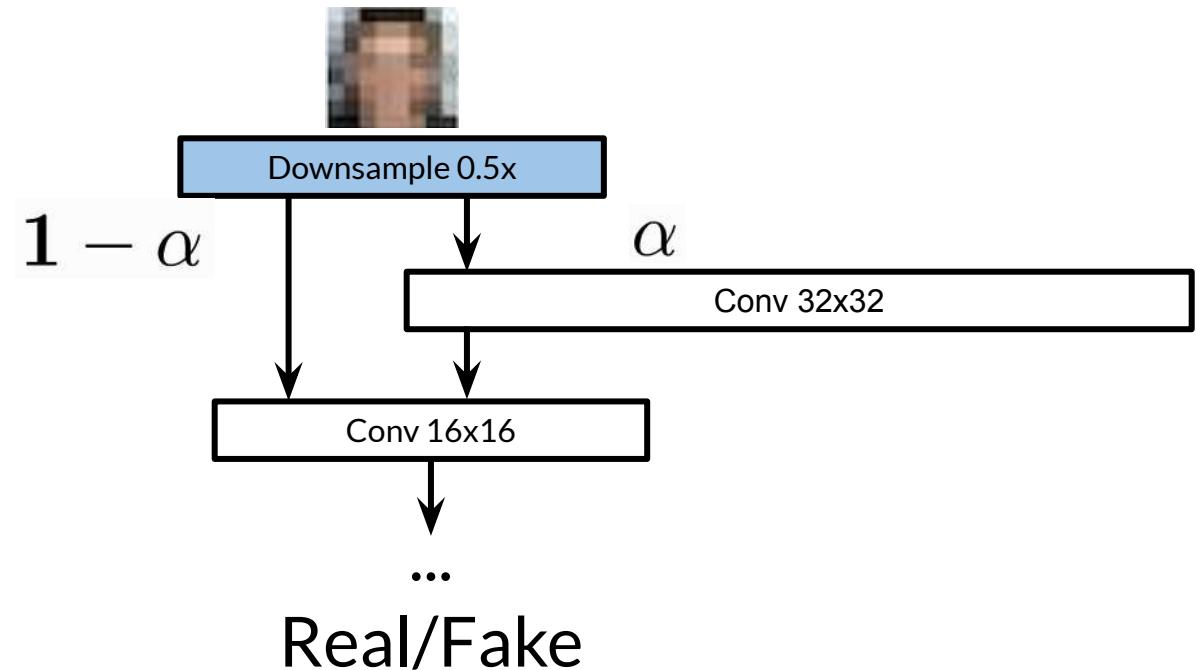
Based on: <https://arxiv.org/abs/1710.10196>

Progressive Growing: Discriminator



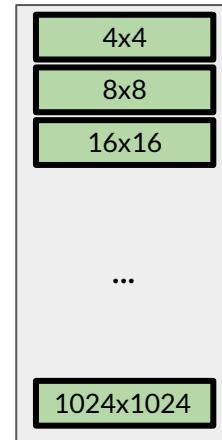
Based on: <https://arxiv.org/abs/1710.10196>

Progressive Growing: Discriminator



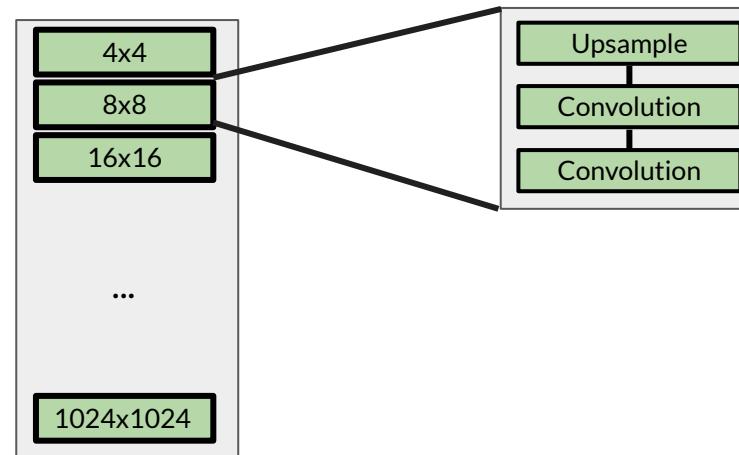
Based on: <https://arxiv.org/abs/1710.10196>

Progressive Growing in Context



Based on: <https://arxiv.org/abs/1812.04948>

Progressive Growing in Context



Based on: <https://arxiv.org/abs/1812.04948>

Summary

- Progressive growing gradually doubles image resolution
- Helps with faster, more stable training for higher resolutions





deeplearning.ai

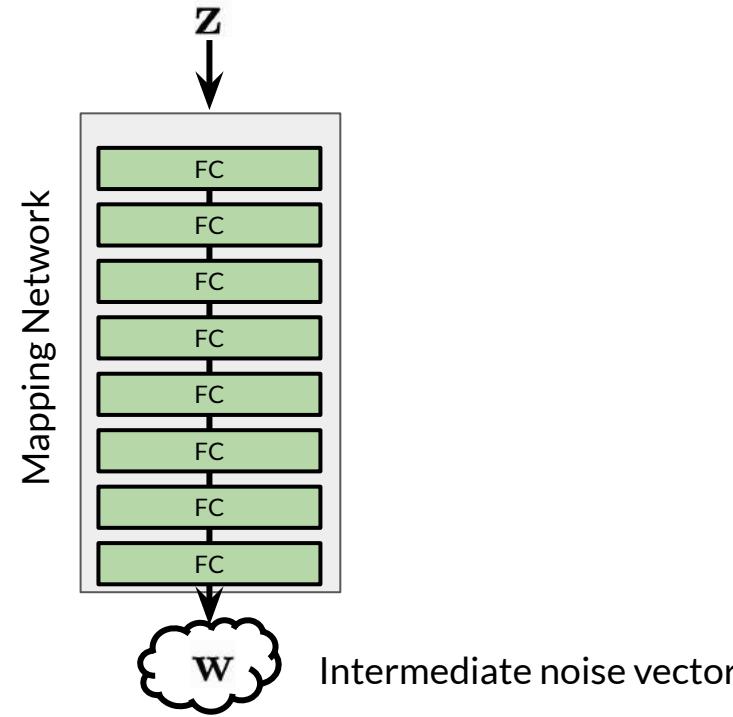
Noise Mapping Network

Outline

- Noise mapping network structure
- Motivation behind the noise mapping network
- Where its output w goes

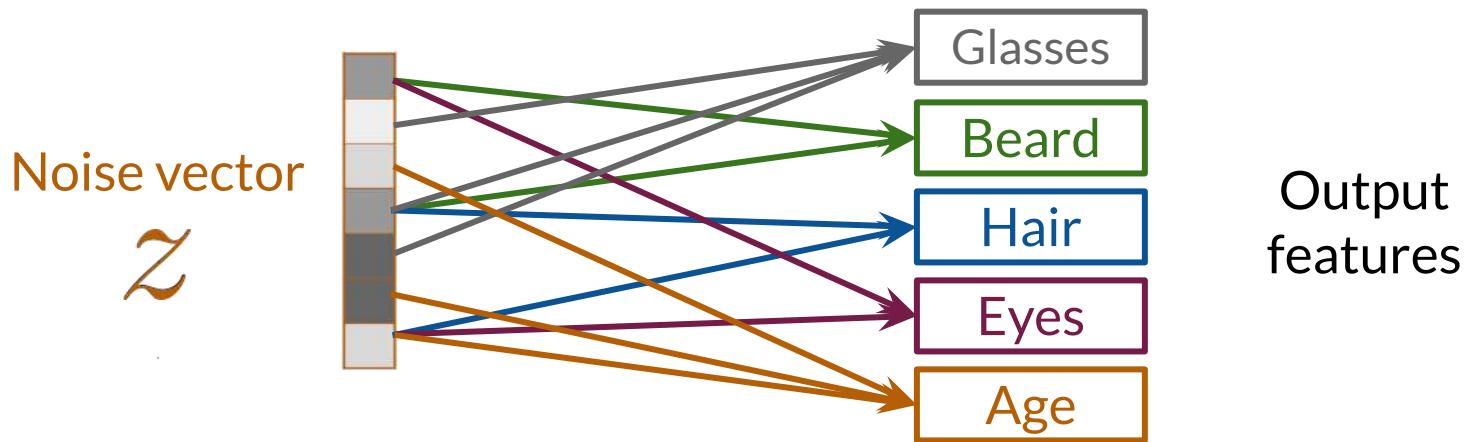


Noise Mapping Network



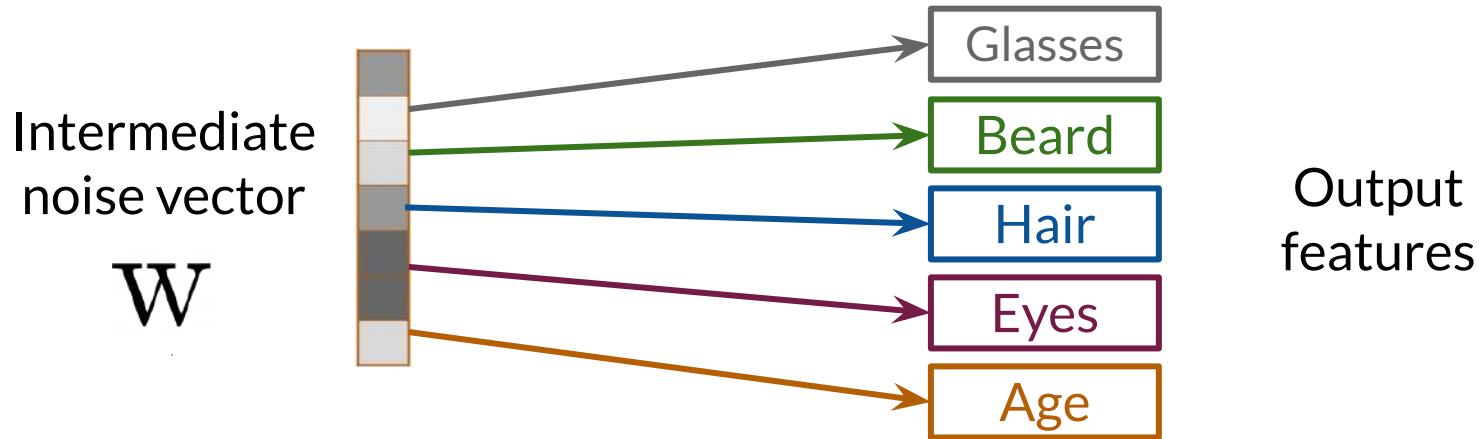
Based on: <https://arxiv.org/abs/1812.04948>

Remember: Z-Space Entanglement



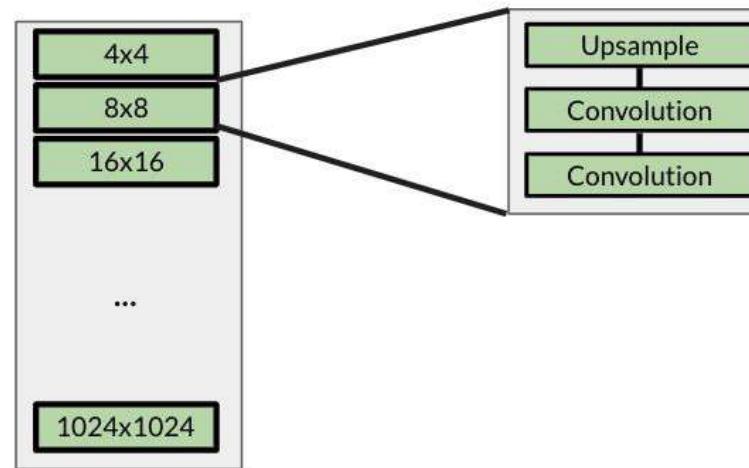
Not possible to control single output features

W-Space: Less Entangled



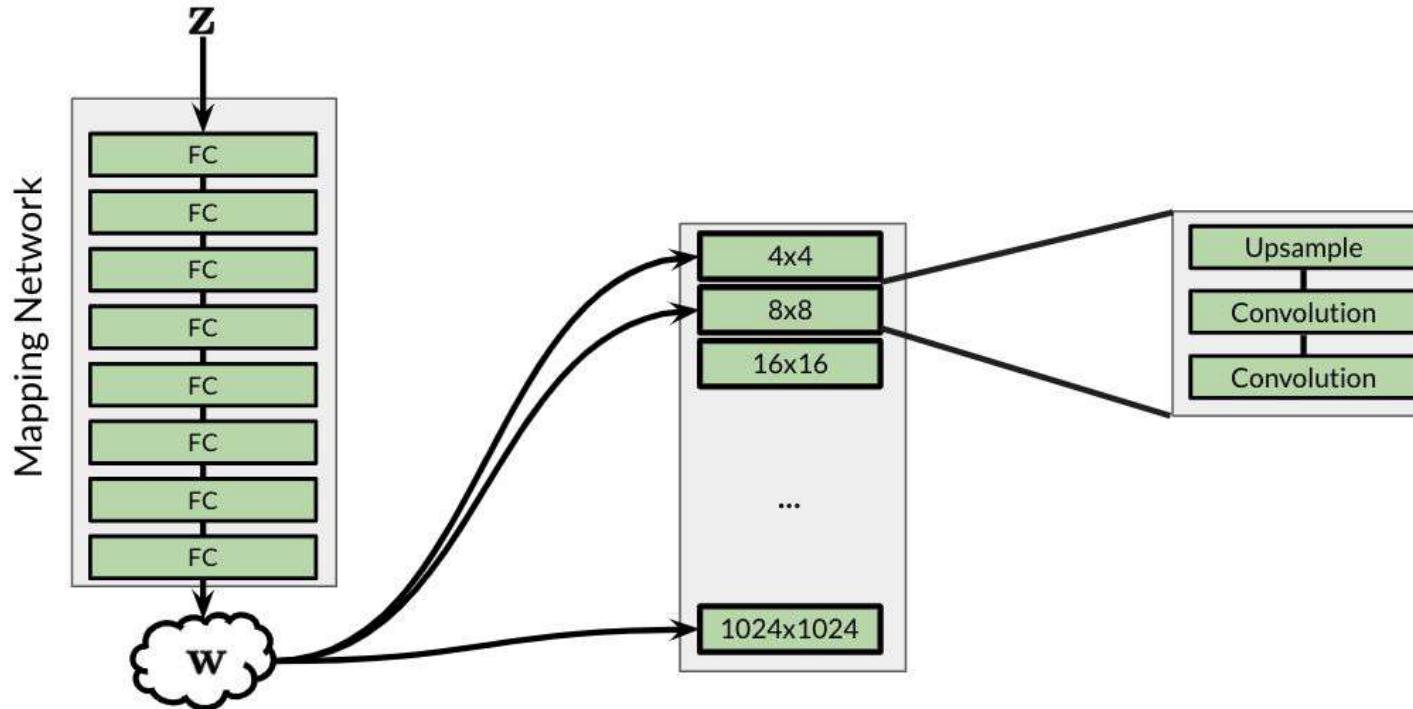
More possible to control single output features

Mapping Network in Context



Based on: <https://arxiv.org/abs/1812.04948>

Mapping Network in Context



Based on: <https://arxiv.org/abs/1812.04948>

Summary

- Noise mapping allows for a more disentangled noise space
- The intermediate noise vector \tilde{W} is used as input to the generator





deeplearning.ai

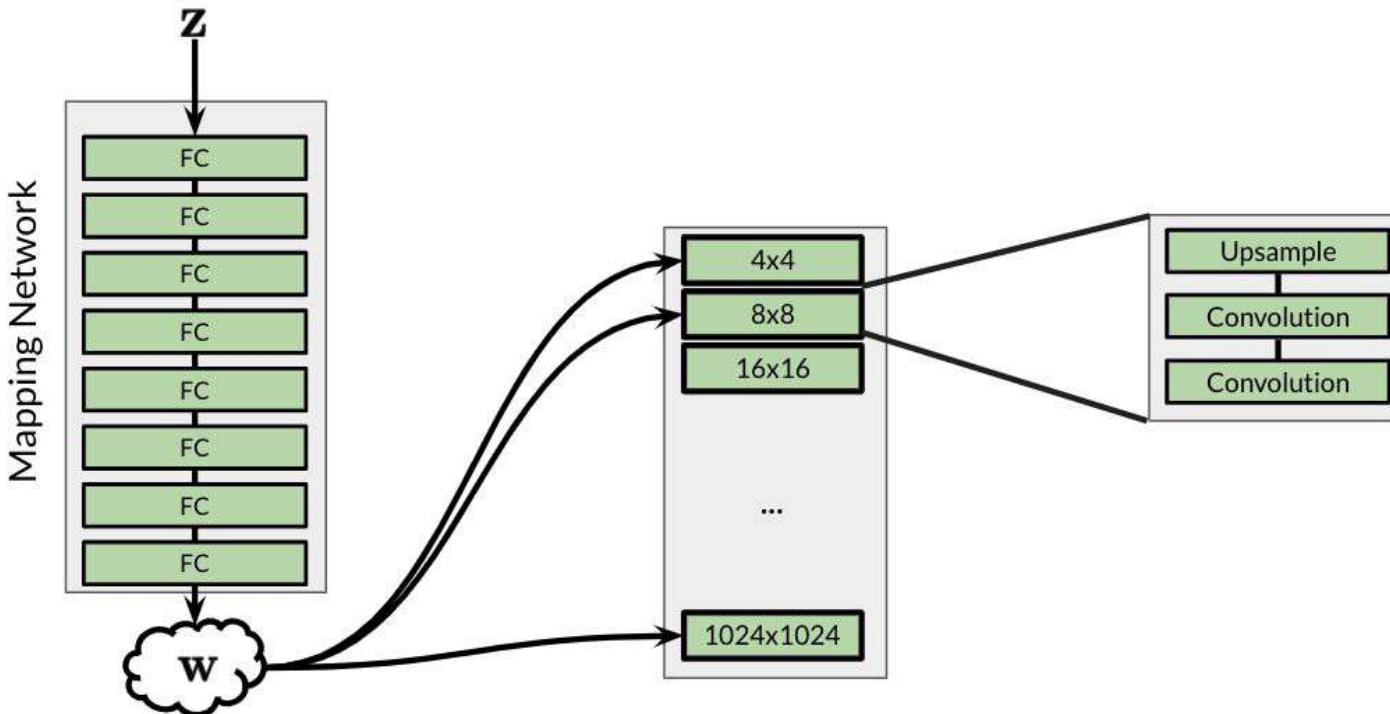
Adaptive Instance Normalization (AdaIN)

Outline

- Instance Normalization
- Adaptive Instance Normalization (AdaIN)
- Where and why AdaIN is used

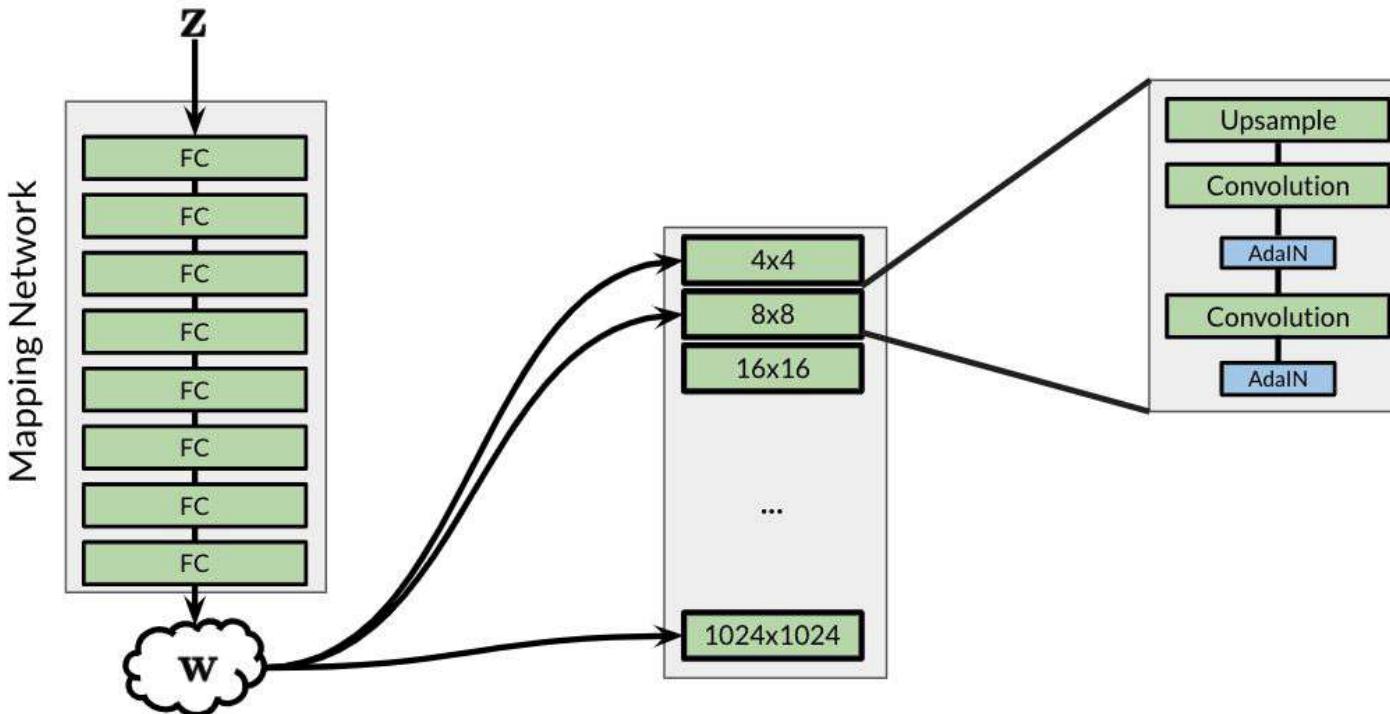


AdaIN in Context



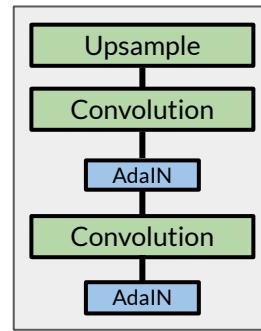
Based on: <https://arxiv.org/abs/1812.04948>

AdaIN in Context



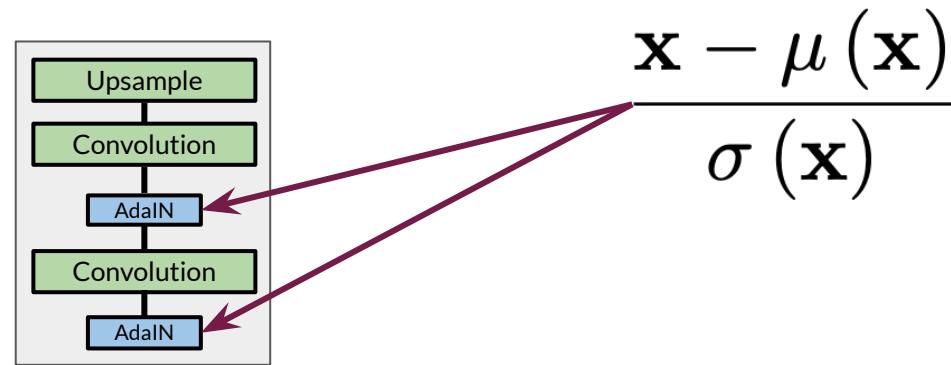
Based on: <https://arxiv.org/abs/1812.04948>

AdaIN



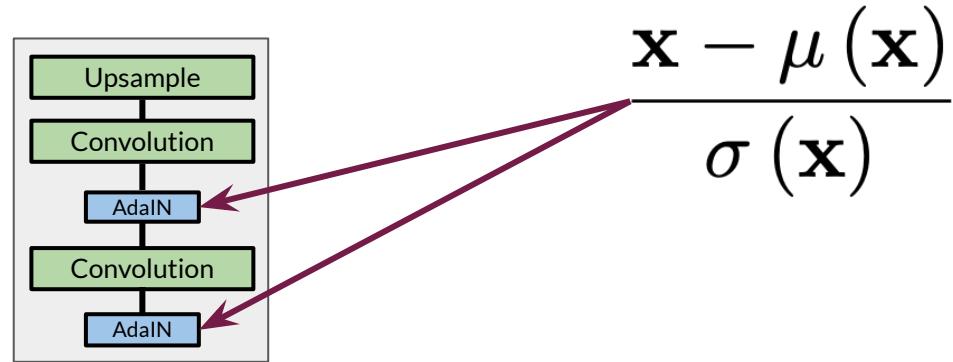
Based on: <https://arxiv.org/abs/1812.04948>

AdaIN



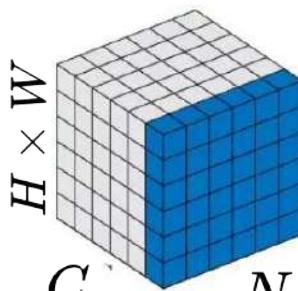
Step 1: Normalize convolution outputs

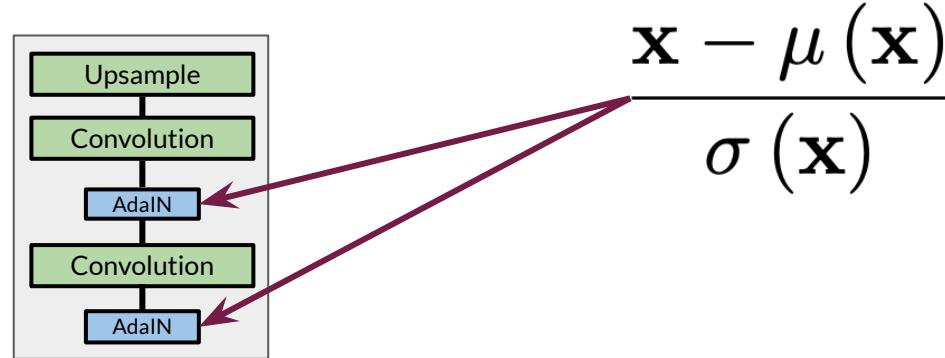
AdaIN



Step 1: Normalize convolution outputs using Instance Normalization

AdaIN

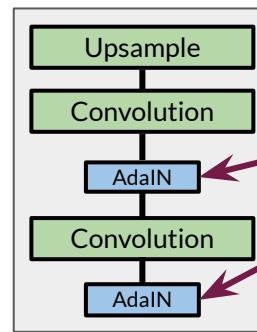
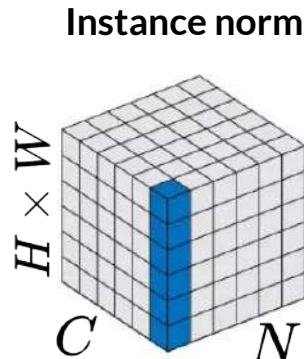
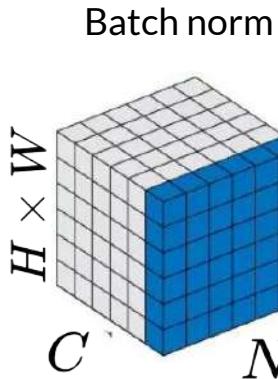
Batch norm

 $H \times W$
 C N



Step 1: Normalize convolution outputs using Instance Normalization

(Left) Available from: <https://medium.com/syncedreview/facebook-ai-proposes-group-normalization-alternative-to-batch-normalization-fb0699bffae7>
(Right) Based on: <https://arxiv.org/abs/1812.04948>

AdaIN

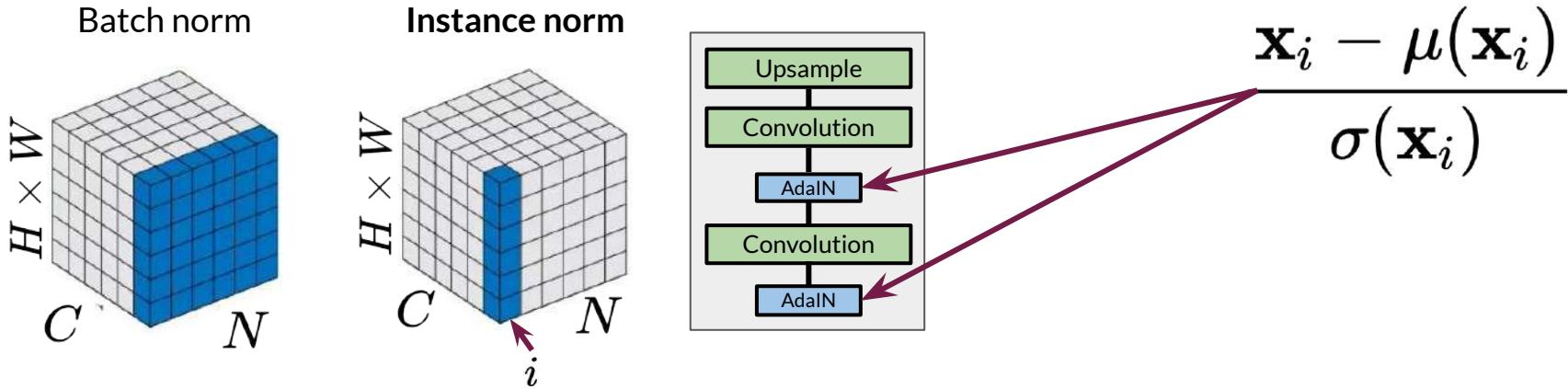


$$\frac{\mathbf{x} - \mu(\mathbf{x})}{\sigma(\mathbf{x})}$$

Step 1: Normalize convolution outputs using Instance Normalization

(Left) Available from: <https://medium.com/syncedreview/facebook-ai-proposes-group-normalization-alternative-to-batch-normalization-fb0699bfae7>
(Right) Based on: <https://arxiv.org/abs/1812.04948>

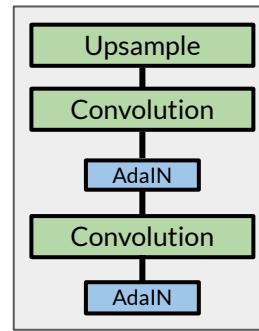
AdaIN



Step 1: Normalize convolution outputs using Instance Normalization

(Left) Available from: <https://medium.com/syncedreview/facebook-ai-proposes-group-normalization-alternative-to-batch-normalization-fb0699bffae7>
(Right) Based on: <https://arxiv.org/abs/1812.04948>

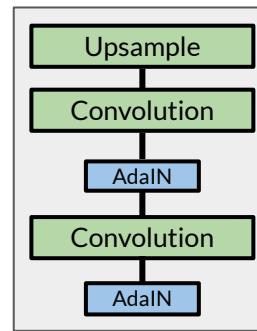
AdaIN



Step 2: Apply **adaptive styles**

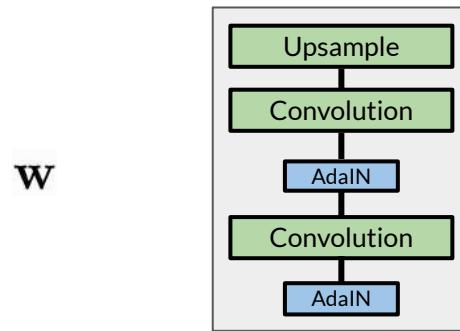
AdaIN

$z \longrightarrow$ Mapping Network $\longrightarrow w$



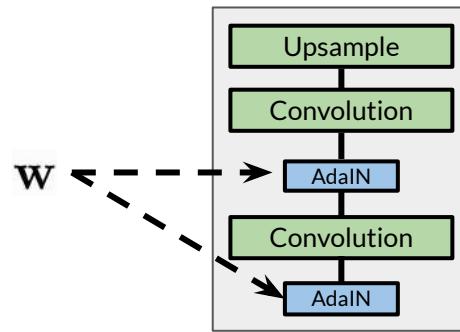
Step 2: Apply adaptive styles using the intermediate noise vector

AdaIN



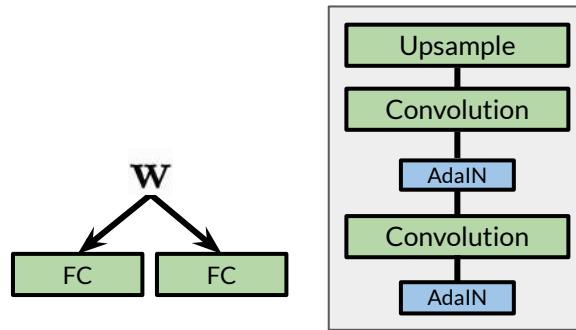
Step 2: Apply adaptive styles using the intermediate noise vector

AdaIN



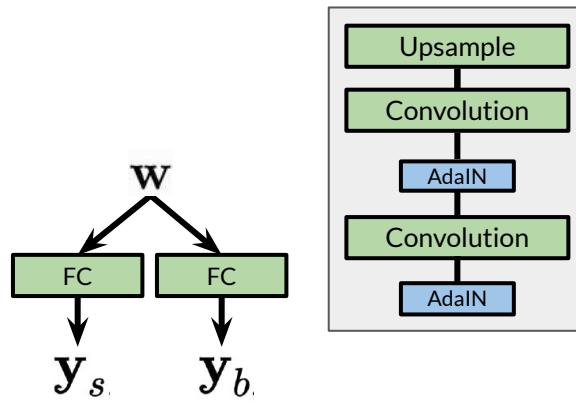
Step 2: Apply **adaptive styles** using the intermediate noise vector

AdaIN



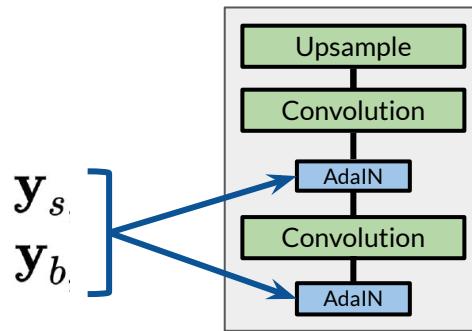
Step 2: Apply adaptive styles using the intermediate noise vector

AdaIN



Step 2: Apply **adaptive styles** using the intermediate noise vector

AdaIN



Step 2: Apply adaptive styles using the intermediate noise vector

AdaIN

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



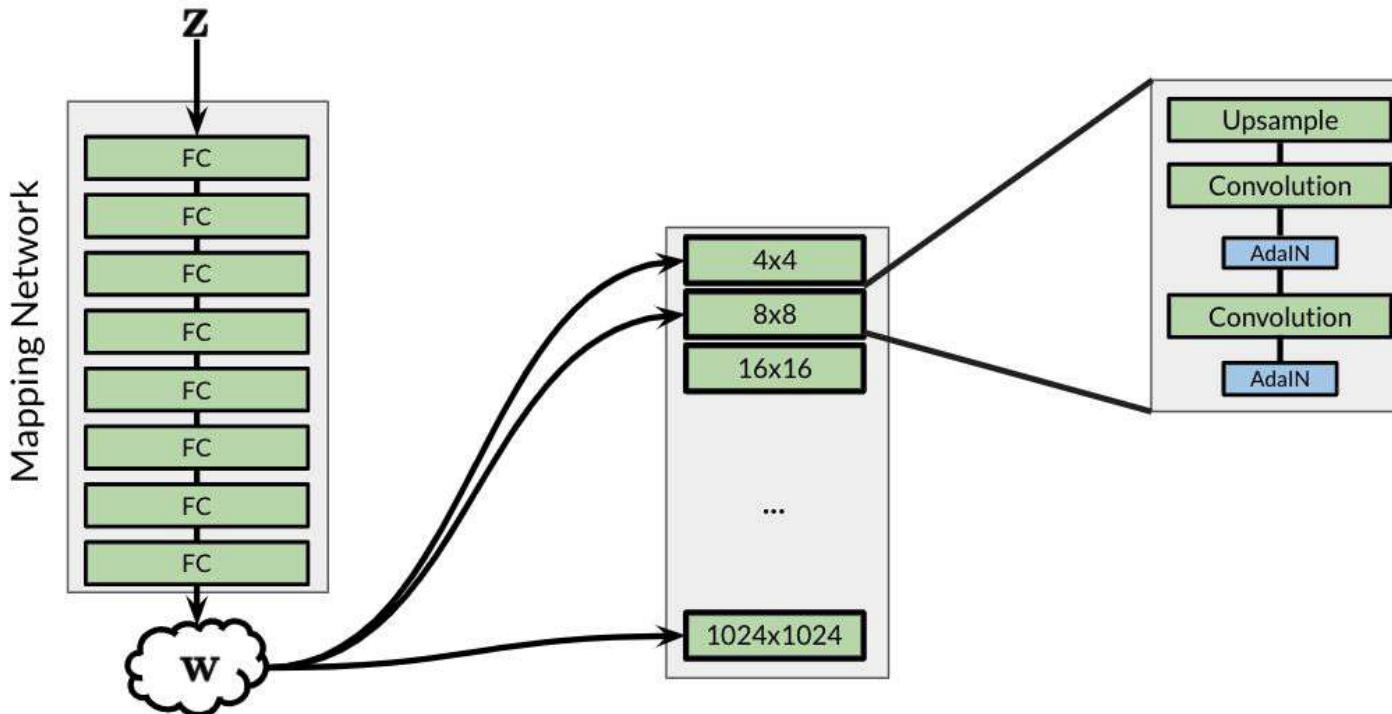
Step 1: Instance normalization

AdaIN

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

Step 2: Adaptive styles

AdaIN in Context



Based on: <https://arxiv.org/abs/1812.04948>

Summary

- AdaIN transfers style information onto the generated image from the intermediate noise vector W
- Instance Normalization is used to normalize individual examples before apply style statistics from W





deeplearning.ai

Style Mixing & Stochastic Noise

Outline

- Controlling coarse and fine styles with StyleGAN
- Style mixing for increased diversity during training/inference
- Stochastic noise for additional variation



Style Mixing

Tabby
Cat

Tuxedo
Cat



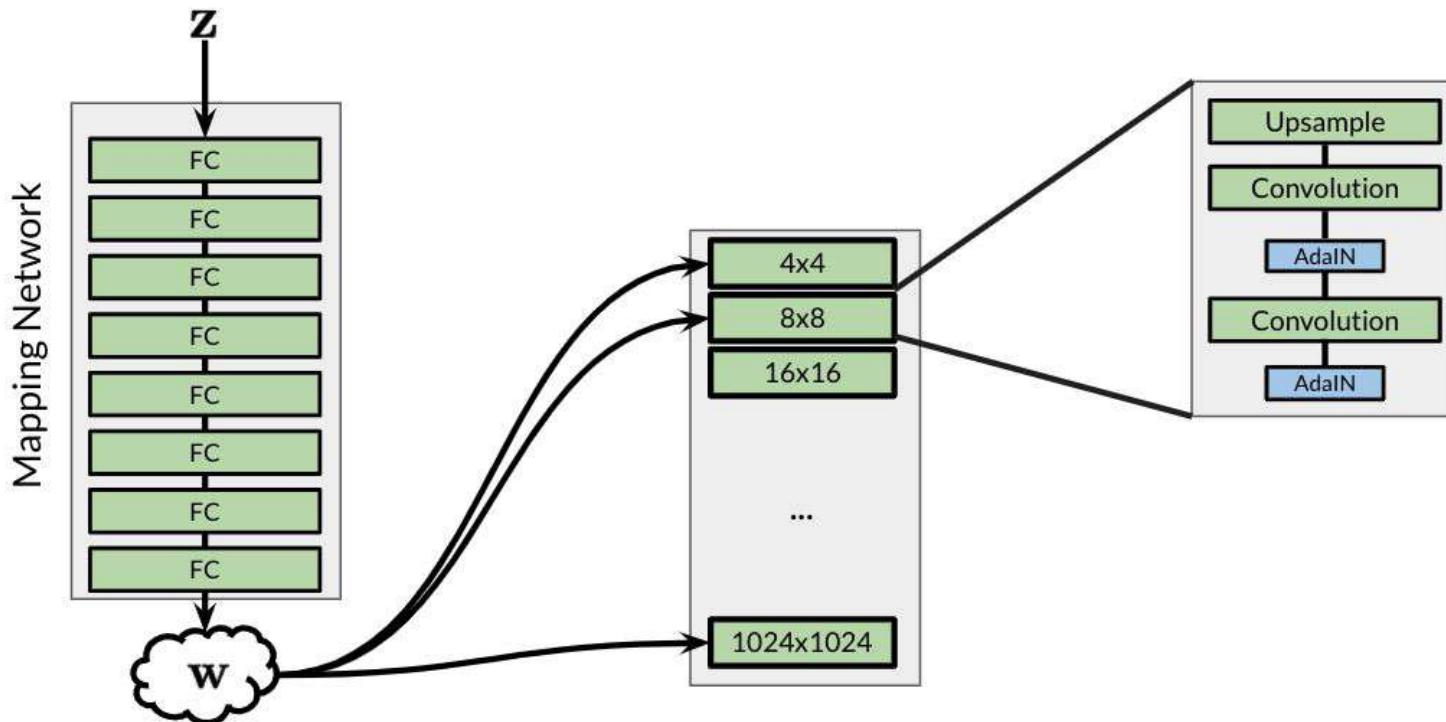
Style Mixing

Tabby Cat

Tuxedo Cat

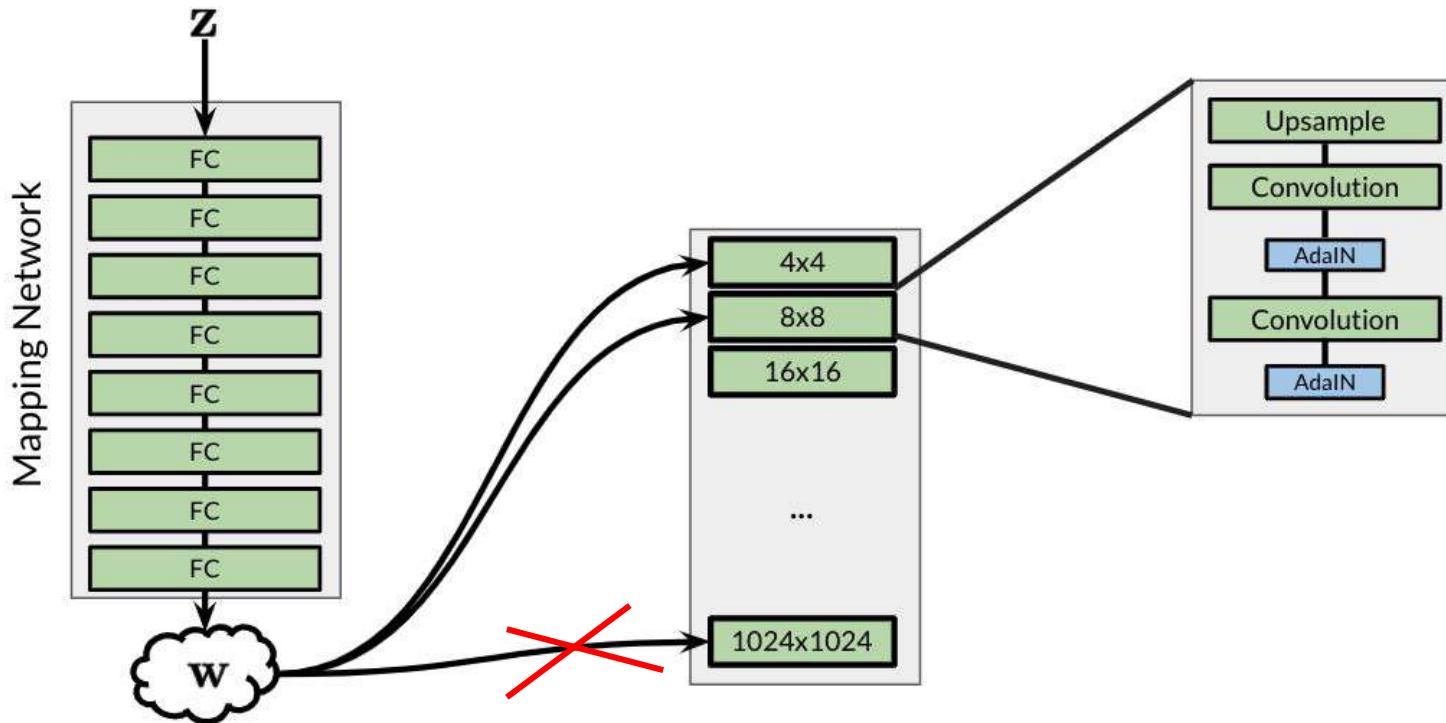


Style Mixing in Context



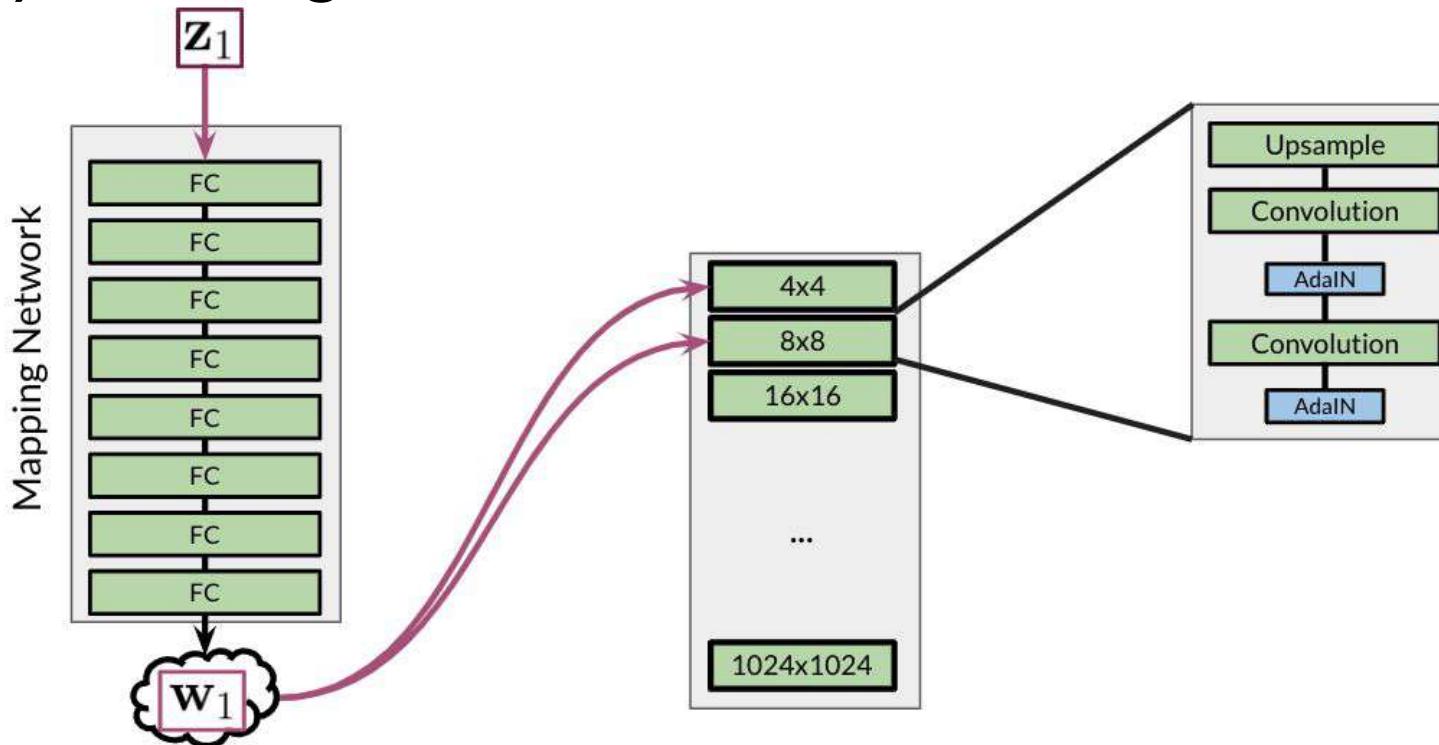
Based on: <https://arxiv.org/abs/1812.04948>

Style Mixing in Context



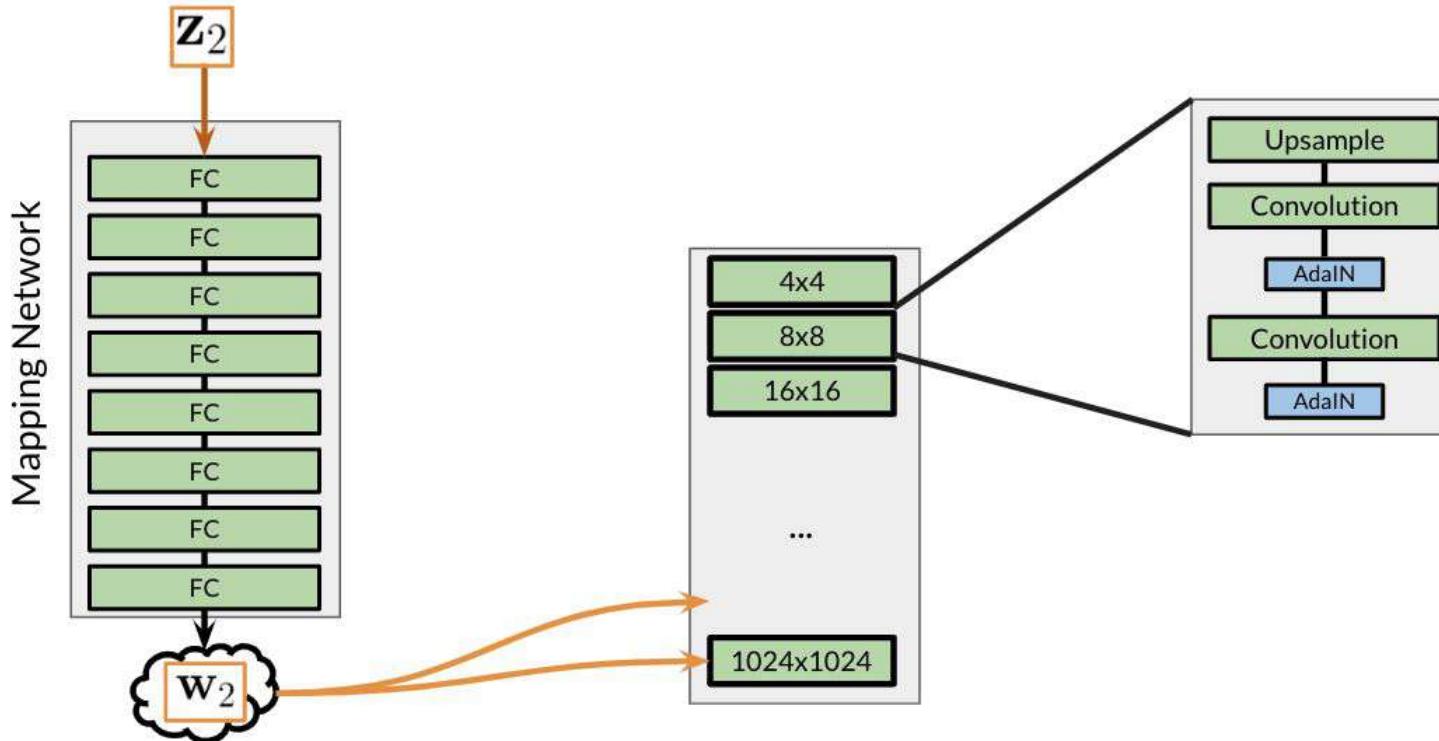
Based on: <https://arxiv.org/abs/1812.04948>

Style Mixing in Context



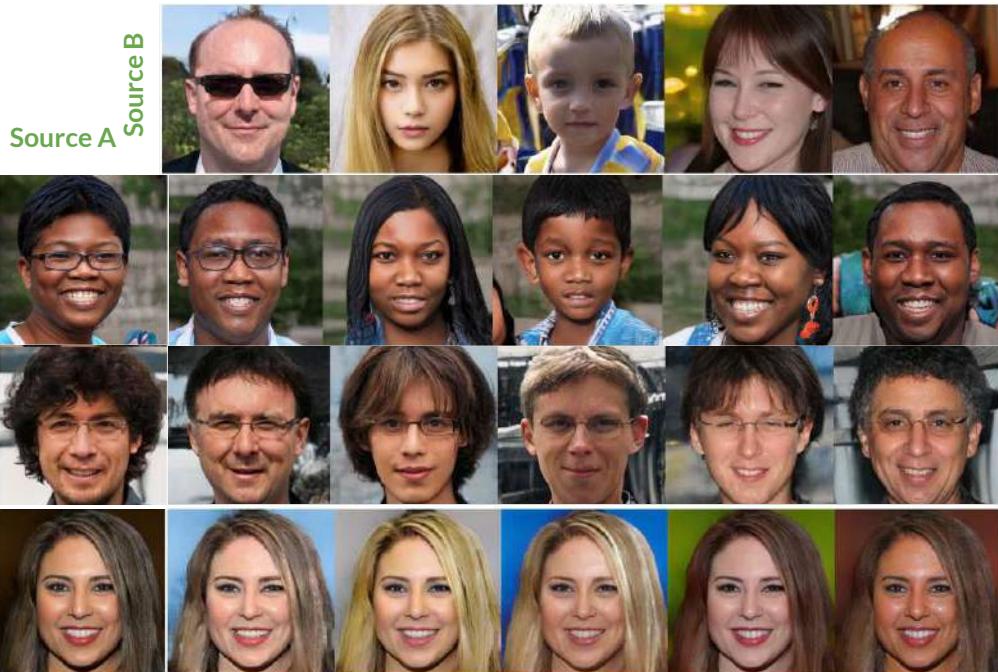
Based on: <https://arxiv.org/abs/1812.04948>

Style Mixing in Context



Based on: <https://arxiv.org/abs/1812.04948>

Style Mixing



Coarse styles from B

Middle styles from B

Fine styles from B

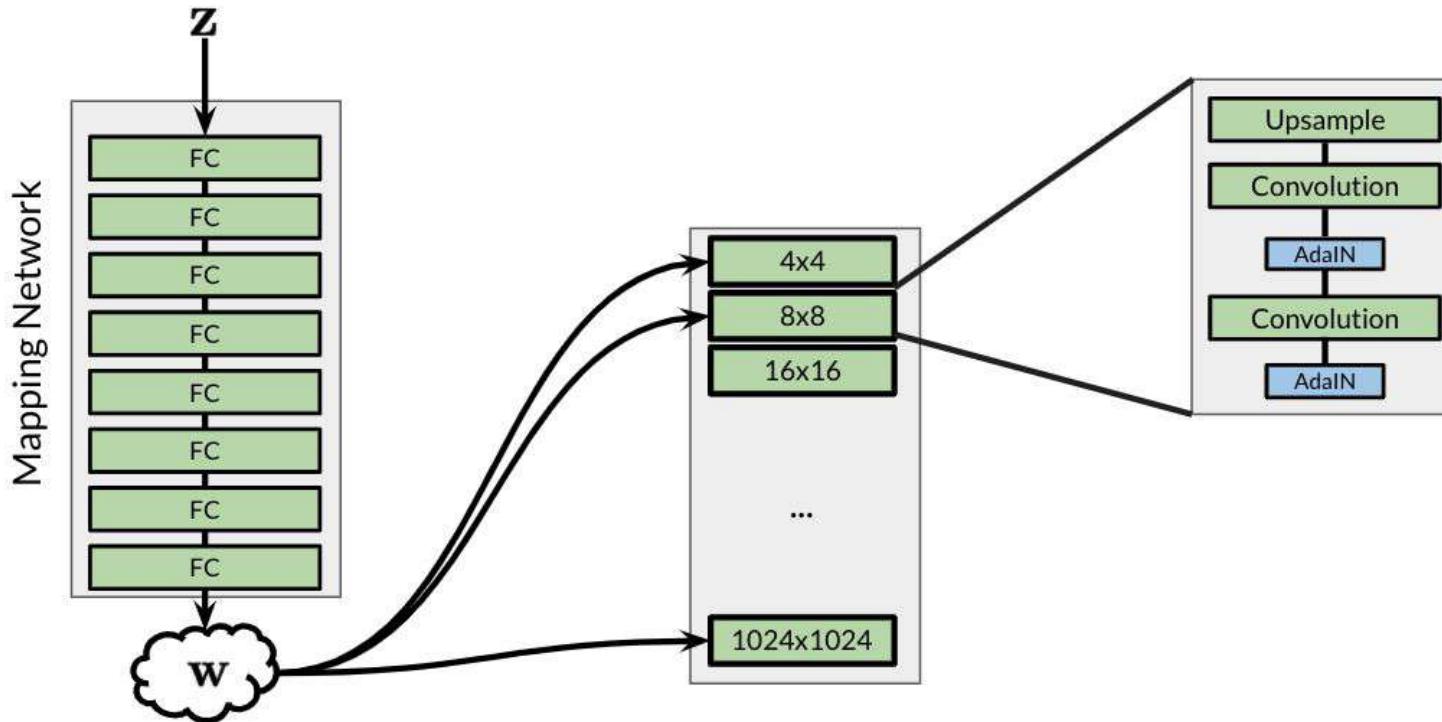
Stochastic Variation

Fine
layers



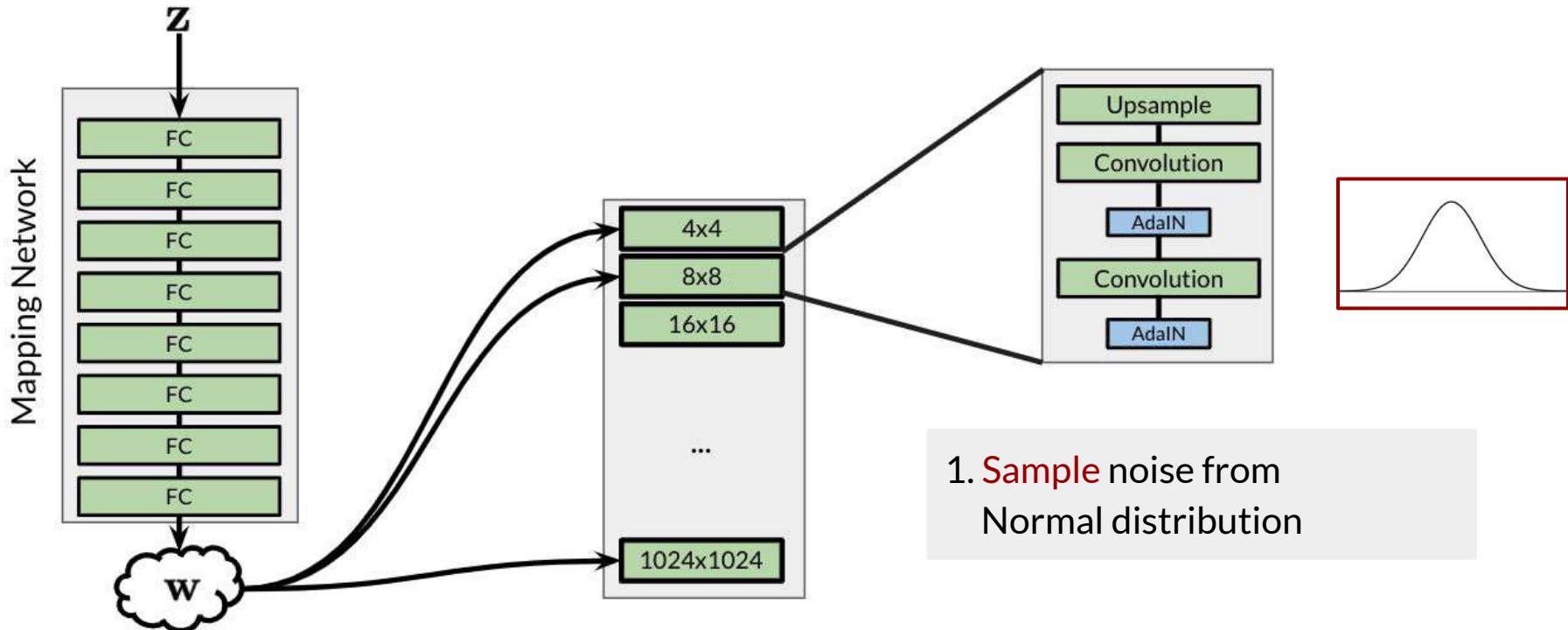
Coarse
layers

Stochastic Noise in Context



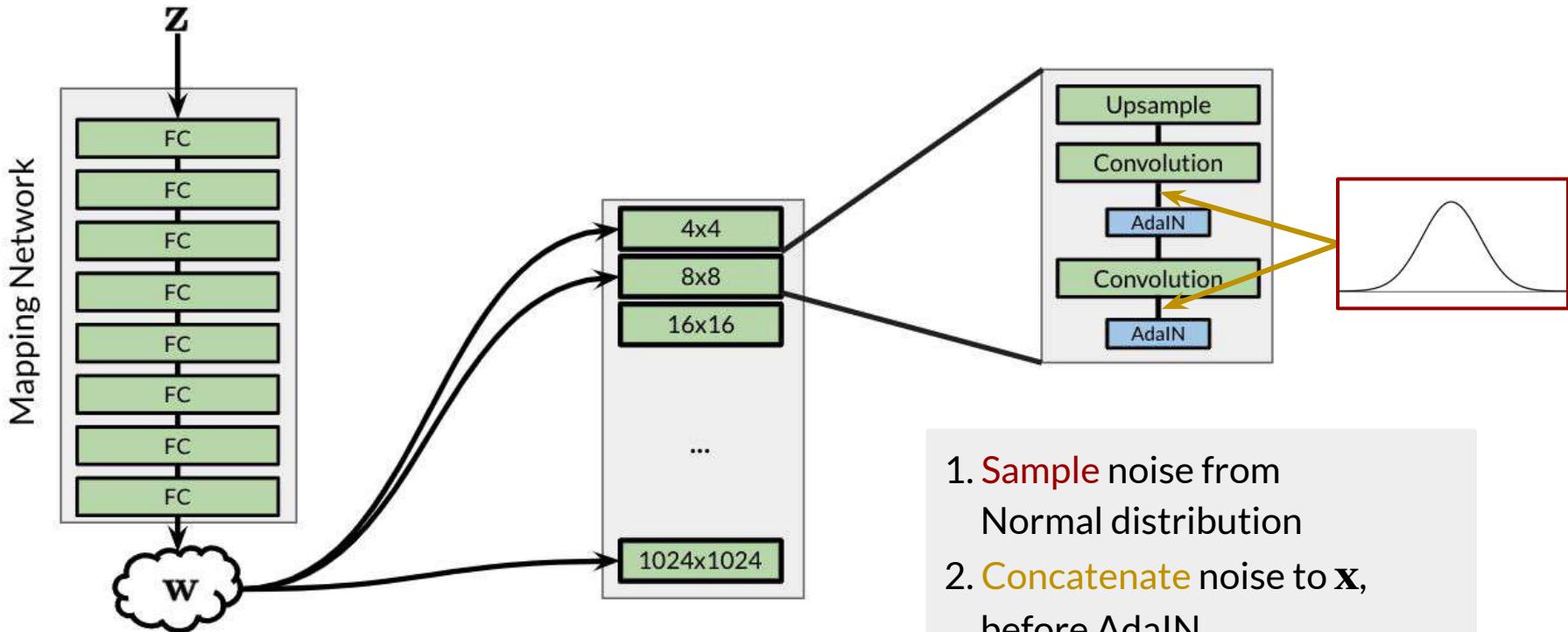
Based on: <https://arxiv.org/abs/1812.04948>

Stochastic Noise in Context



Based on: <https://arxiv.org/abs/1812.04948>

Stochastic Noise in Context



Based on: <https://arxiv.org/abs/1812.04948>

Stochastic Variation

Small details: hair strands,
wrinkles, etc.

Different extra noise values
create stochastic variation



Summary

- Style mixing increases diversity that the model sees during training
- Stochastic noise causes small variations to output
- Coarse or fineness depends where in the network style or noise is added
 - Earlier for coarser variation
 - Later for finer variation



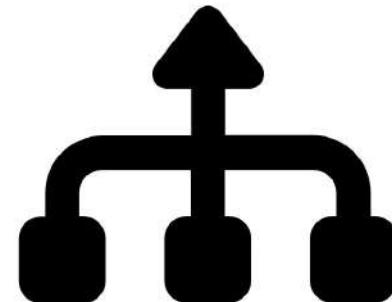


deeplearning.ai

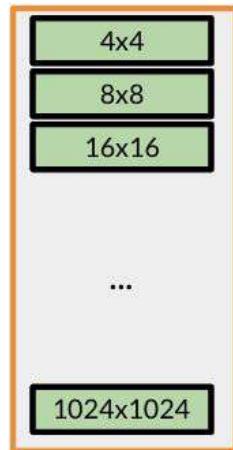
Putting It All Together

Outline

- Putting all the StyleGAN components together!

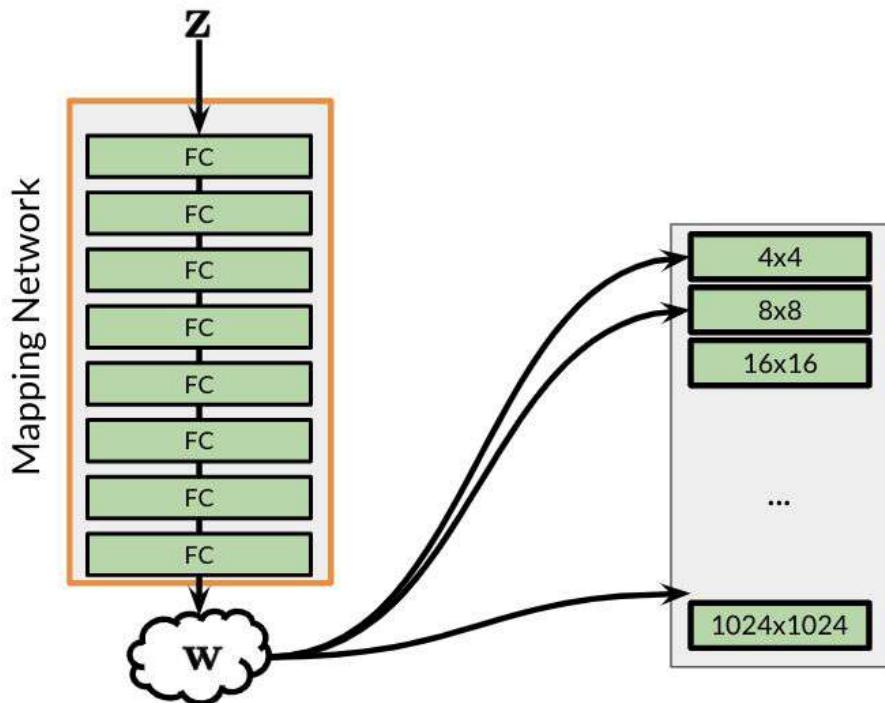


StyleGAN Architecture: Progressive Growing



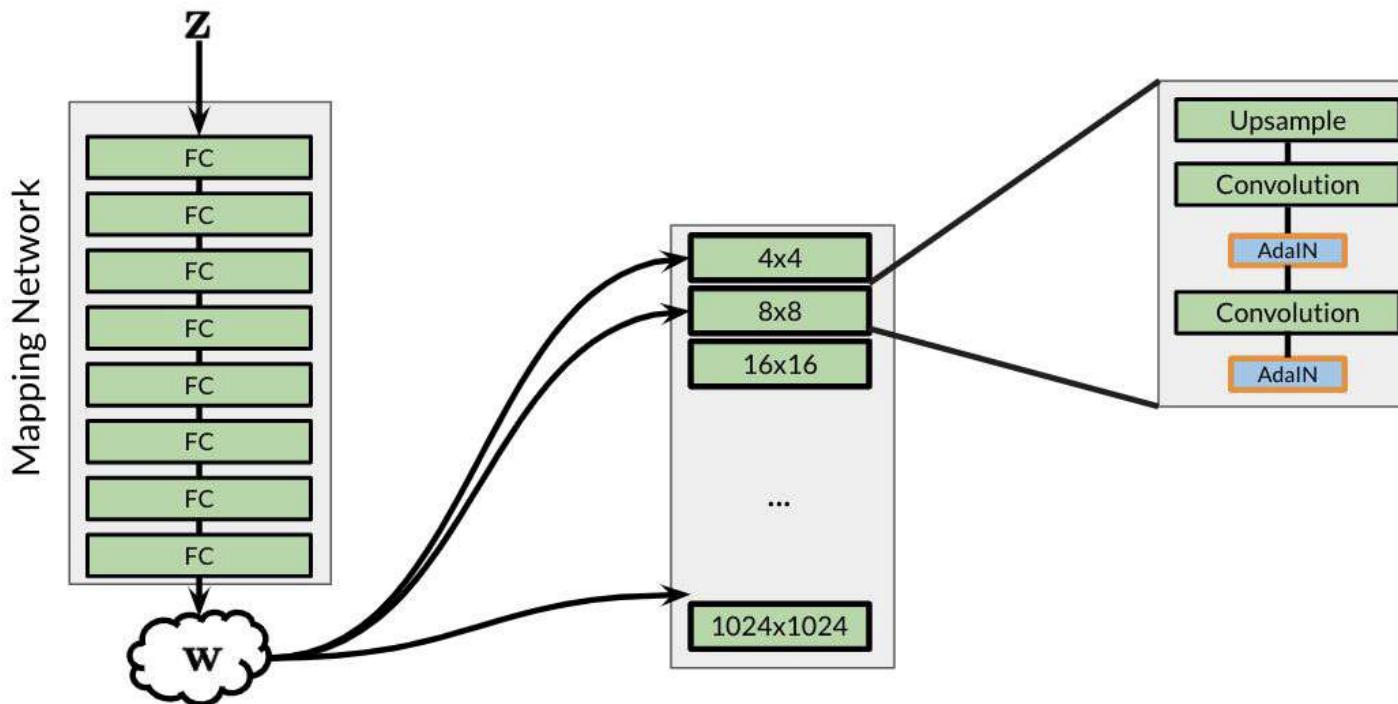
Based on: <https://arxiv.org/abs/1812.04948>

StyleGAN Architecture: Noise Mapping Network



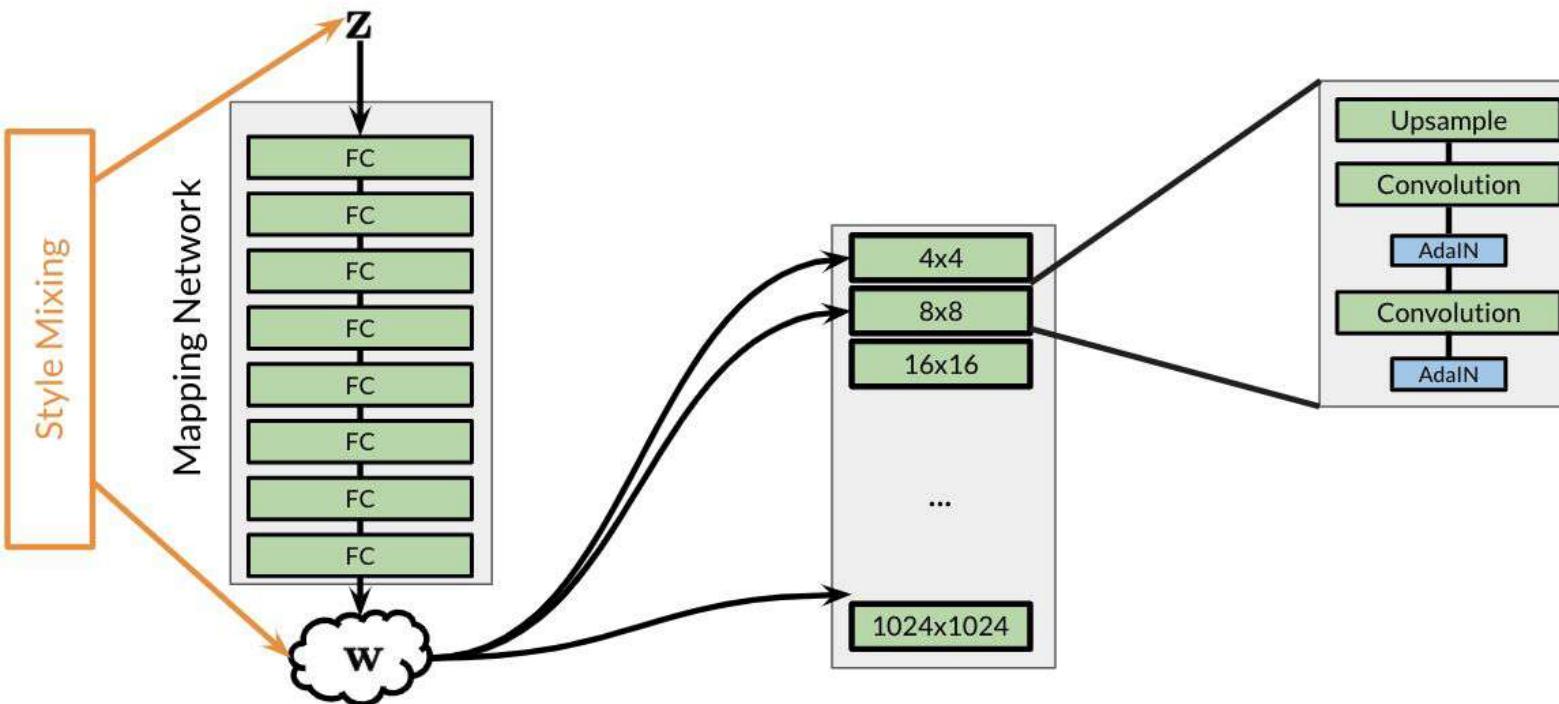
Based on: <https://arxiv.org/abs/1812.04948>

StyleGAN Architecture: AdaIN



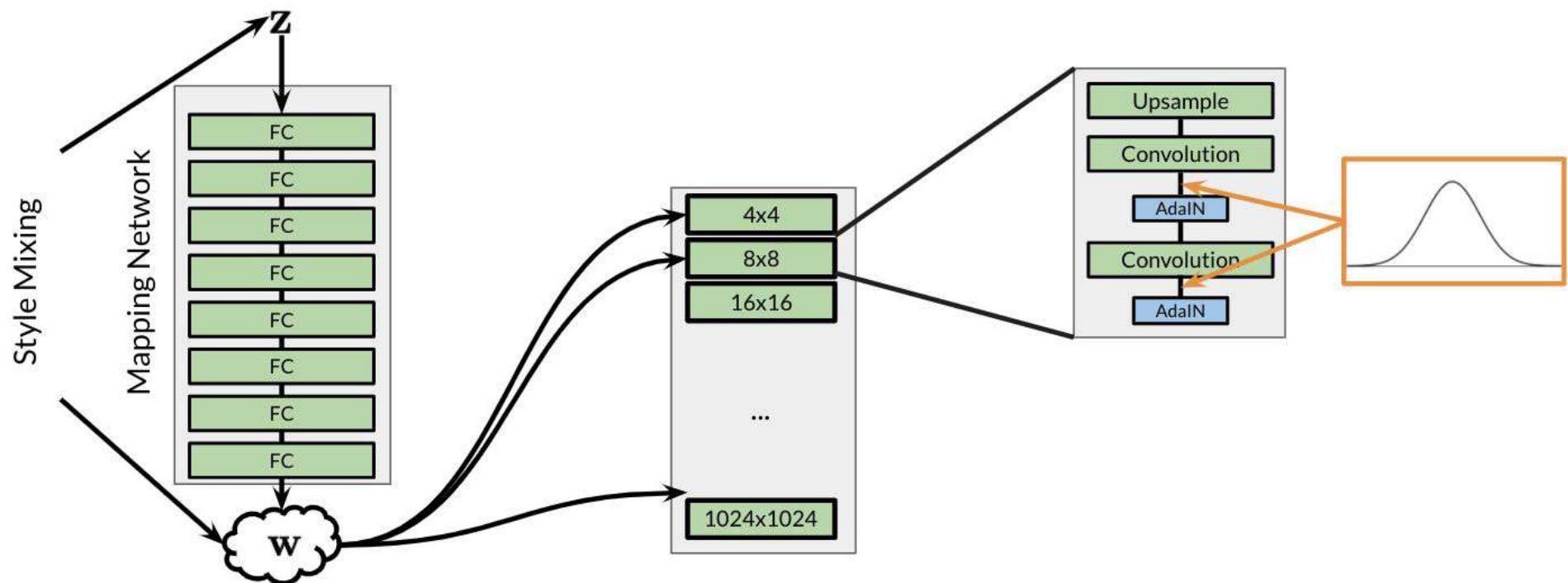
Based on: <https://arxiv.org/abs/1812.04948>

StyleGAN Architecture: Style Mixing



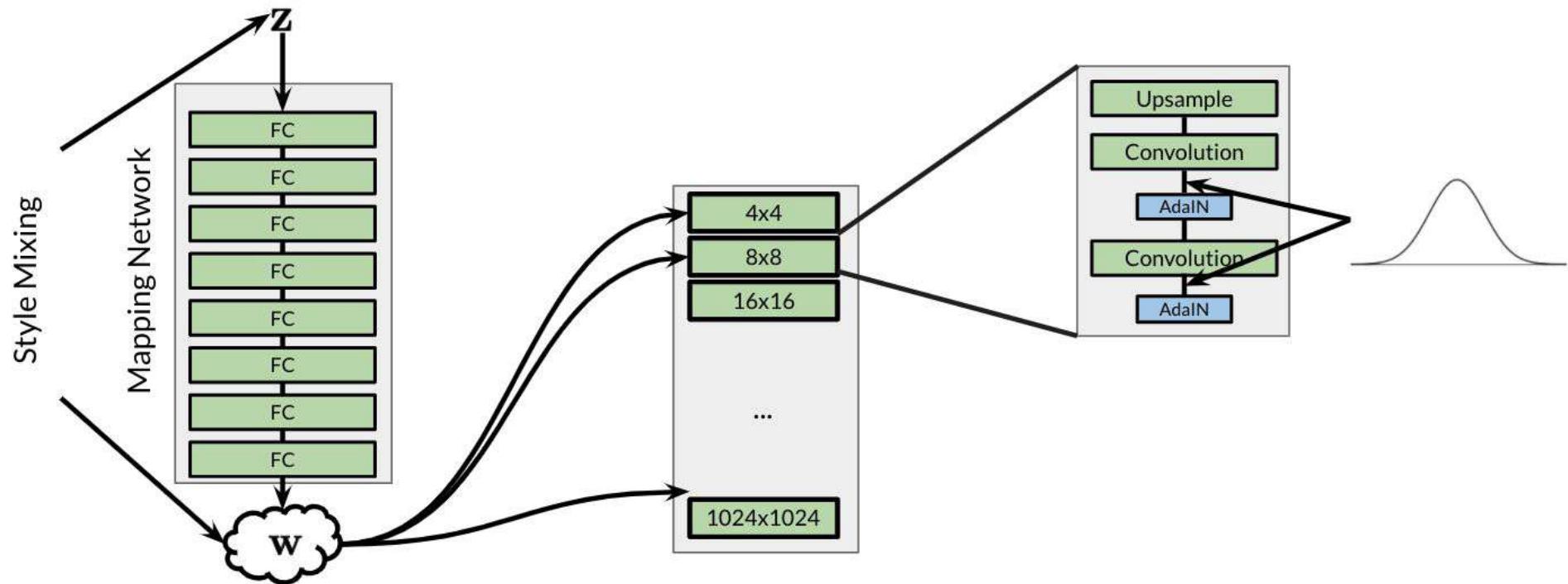
Based on: <https://arxiv.org/abs/1812.04948>

StyleGAN Architecture: Stochastic Noise



Based on: <https://arxiv.org/abs/1812.04948>

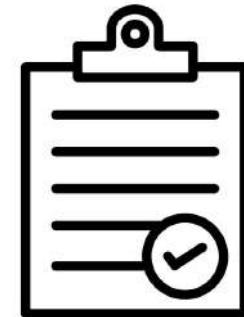
StyleGAN Architecture: That's a Wrap!



Based on: <https://arxiv.org/abs/1812.04948>

Summary

- Main components of StyleGAN:
 - Progressive Growing
 - Noise Mapping Network
 - AdaIN
 - Style Mixing
 - Stochastic Noise



Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>



deeplearning.ai

Overview of GAN Applications

Outline

- GAN applications
 - Image-to-image translation – and extensions to other modalities such as text, audio, and video
 - Image editing, art, and media
 - Medicine and climate change
- GAN adversarial concept use in other research areas

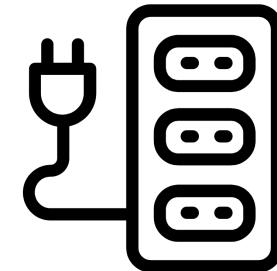
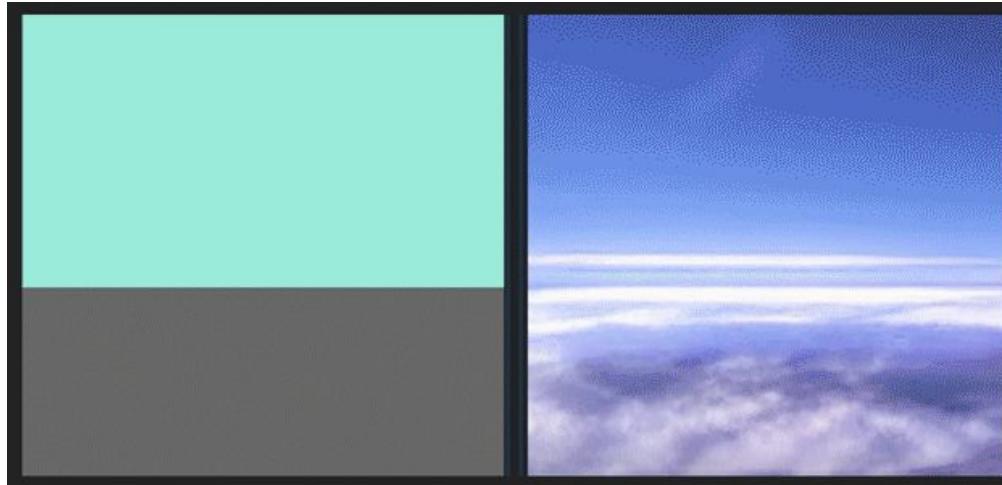


Image-to-Image



GauGAN

Available from: <https://arxiv.org/abs/1903.07291>

Image-to-Image



Original image

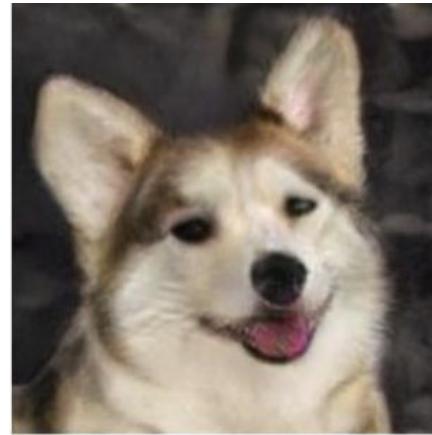
Super-Resolution GAN



Sharpened image

Available from: <https://arxiv.org/abs/1609.04802>

Image-to-Image



Multimodal image-to-image translation

Available from: <https://github.com/NVlabs/MUNIT>

Text-to-Image

“The bird is black with
green and has a very
short beak.”



Available from: <https://arxiv.org/abs/1612.03242>

Image-and-Landmark-to-Video



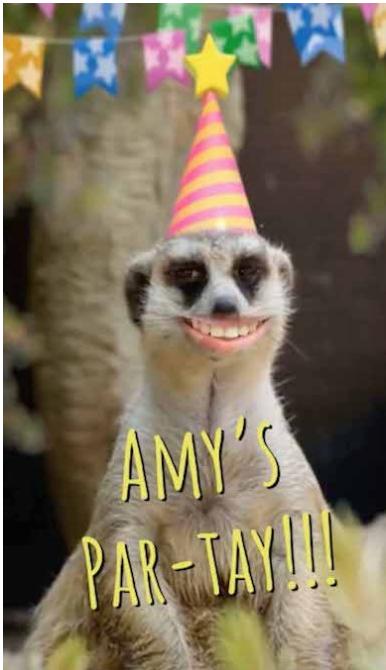
Image

Face landmark

Talking heads

Available from: <https://arxiv.org/abs/1905.08233>

Application Areas: Image Filters



Available from: <https://www.snapchat.com>

Application Areas: Image Editing



Image



Mask

Image editing software

Available from: <https://arxiv.org/abs/1907.11922>

Application Areas: Image Editing



Image



Mask

Edited mask

Image editing software

Available from: <https://arxiv.org/abs/1907.11922>

Application Areas: Image Editing



Image



Mask



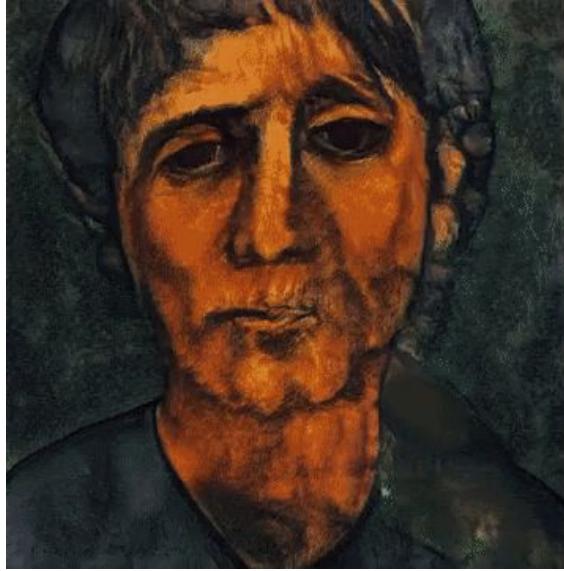
Edited mask

Edited image

Image editing software

Available from: <https://arxiv.org/abs/1907.11922>

Application Areas: Stylized Images



Democratized art

Available from: <https://www.youtube.com/watch?v=85I961MmY8Y>

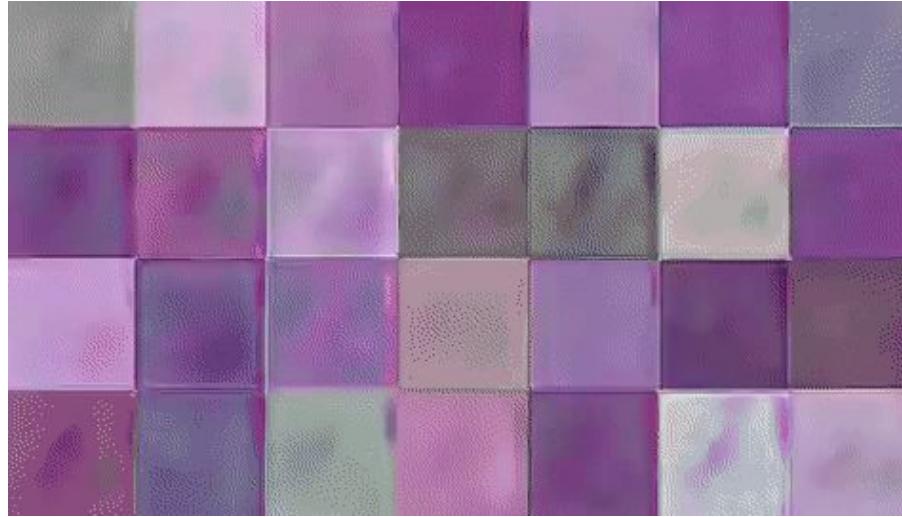
Application Areas: Data Augmentation



Increasing dataset size and diversity

Available from: <https://arxiv.org/abs/1711.04340>

Application Areas: Medicine



Simulating tissues

Available from: <https://twitter.com/realSharonZhou/status/1182877446690852867>

Application Areas: Climate Change



Real input



Generated output

Available from: <https://iopscience.iop.org/article/10.1088/2632-2153/ab7657/meta>

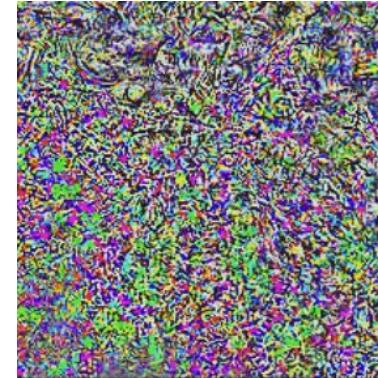
Application Areas: Media



Both pretty cute, actually

Available from: https://en.wikipedia.org/wiki/File:Deepfake_example.gif

Adversarial Research Areas



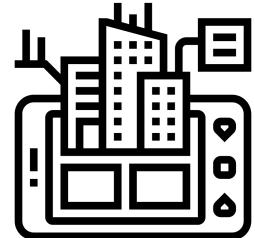
Predicted class: airliner
Predicted probability: 96.8%

Adversarial examples & robustness

Available from: <https://adversarial-ml-tutorial.org/introduction/>

Summary

- Image translation generalizes to many tasks
- Many immediate application areas, including data augmentation
- Other fields use adversarial techniques for realism and robustness



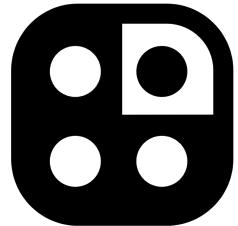


deeplearning.ai

Data Augmentation Methods and Uses

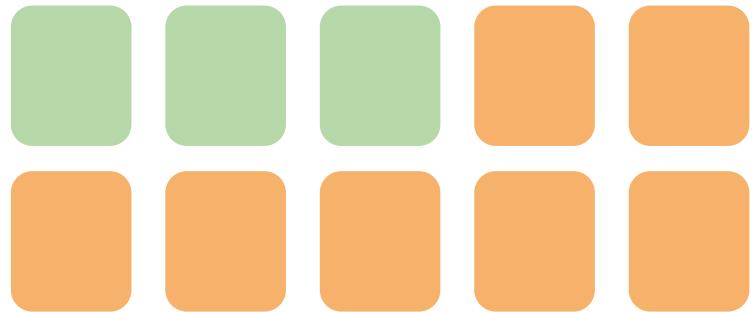
Outline

- Data augmentation and use cases
- Implementation of data augmentation



Data Augmentation

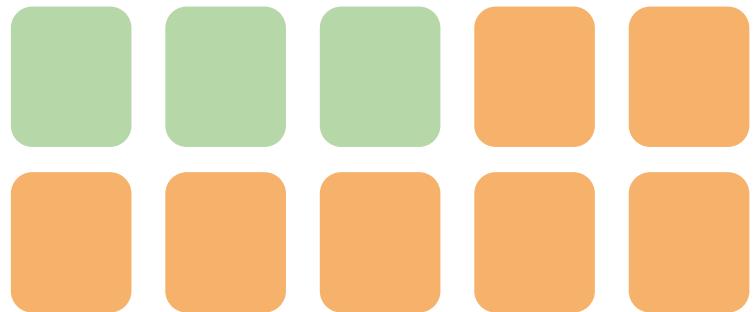
- Supplement data when real data is...
 - Too expensive
 - Too rare



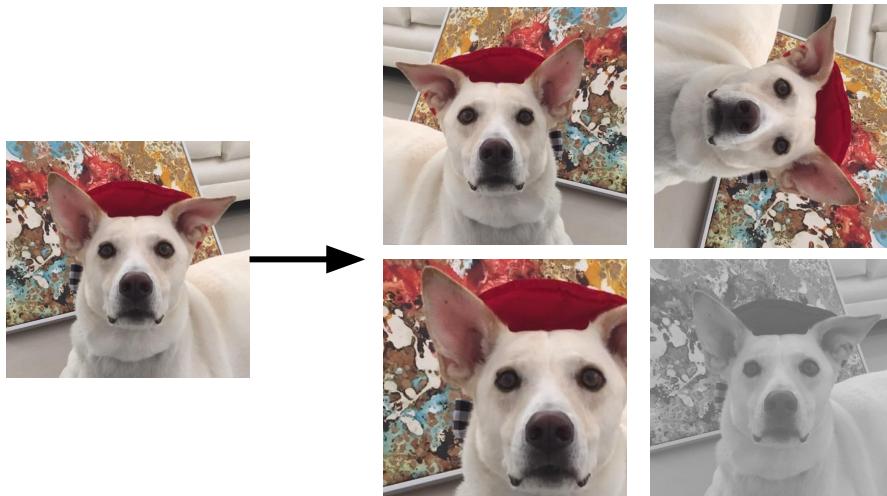
Data Augmentation

- Supplement data when real data is...
 - Too expensive
 - Too rare

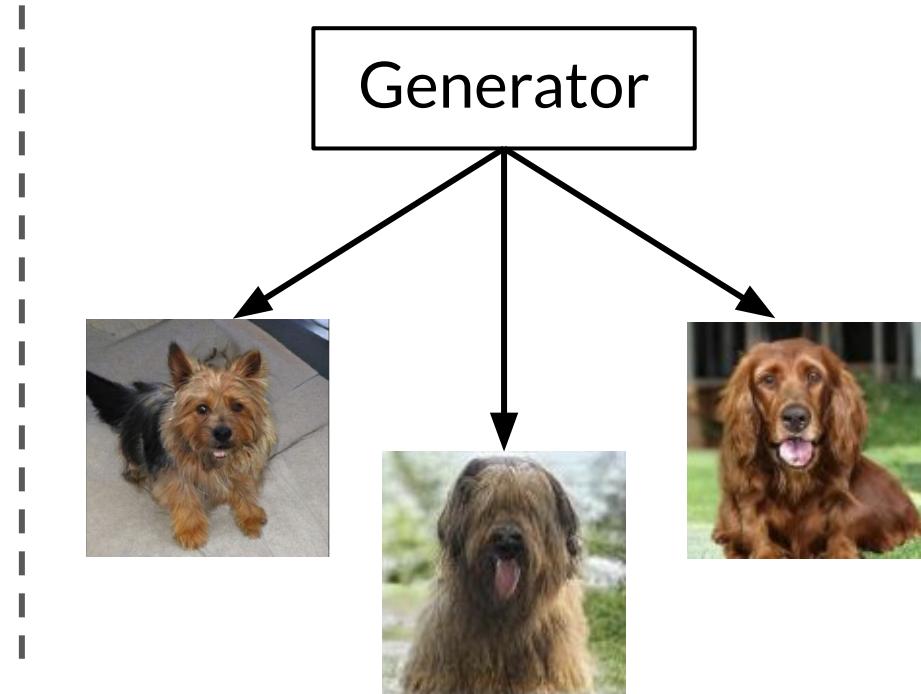
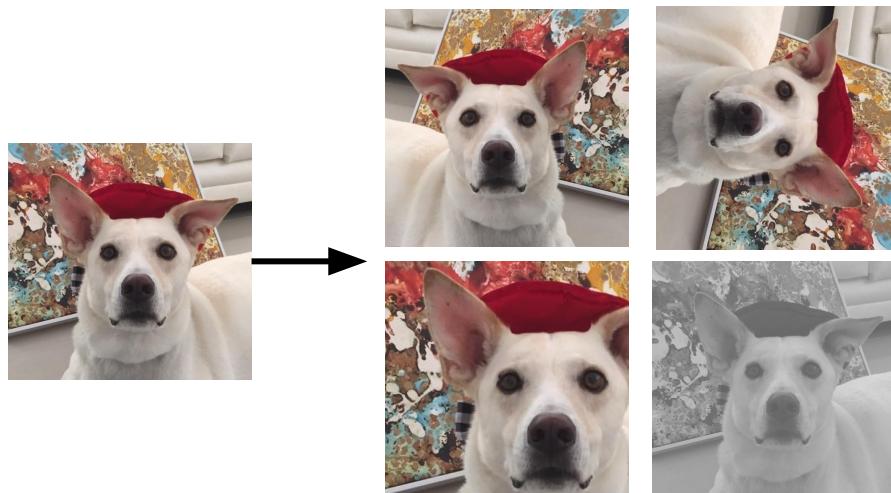
GANs are well suited for this



How to Augment Data

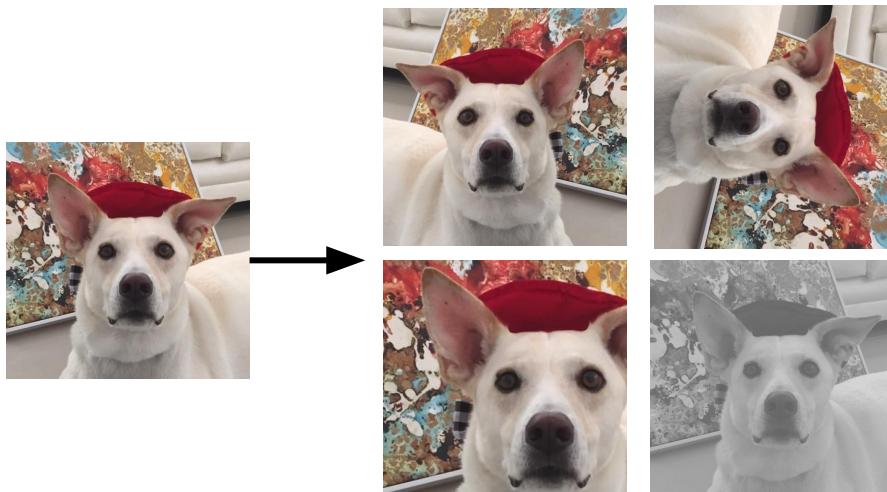


How to Augment Data

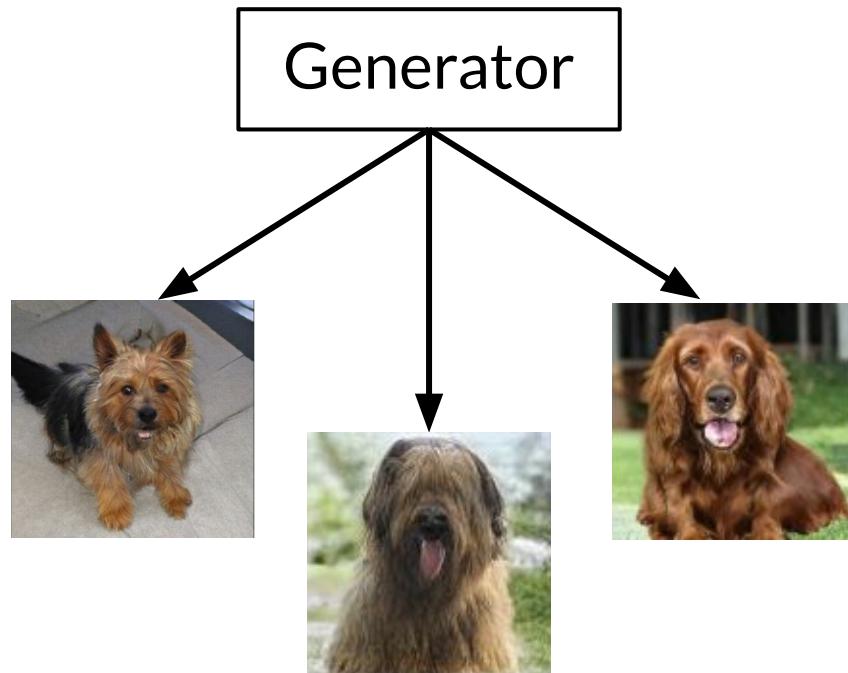


(Right) Available from: <https://arxiv.org/abs/1809.11096>

How to Augment Data



Can mix data augmentation techniques!



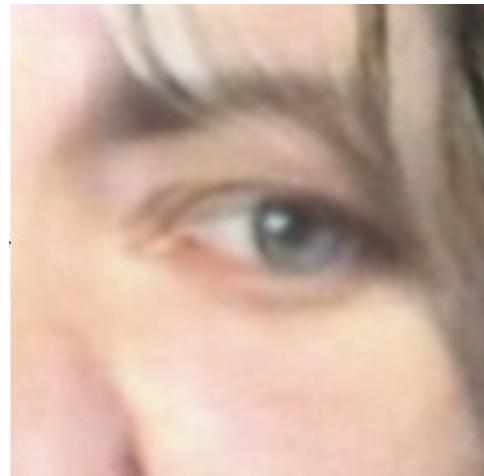
(Right) Available from: <https://arxiv.org/abs/1809.11096>

Use Cases

Synthetic



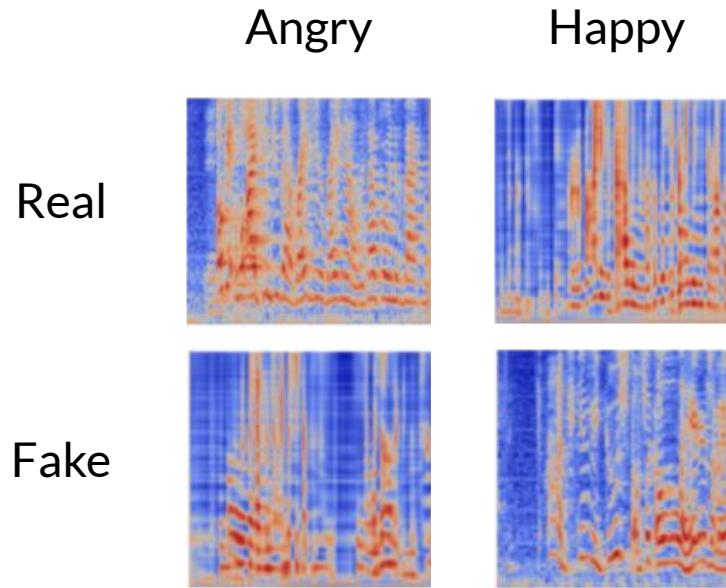
Generated



Gaze detection

Available from: <https://arxiv.org/abs/1711.09767>

Use Cases

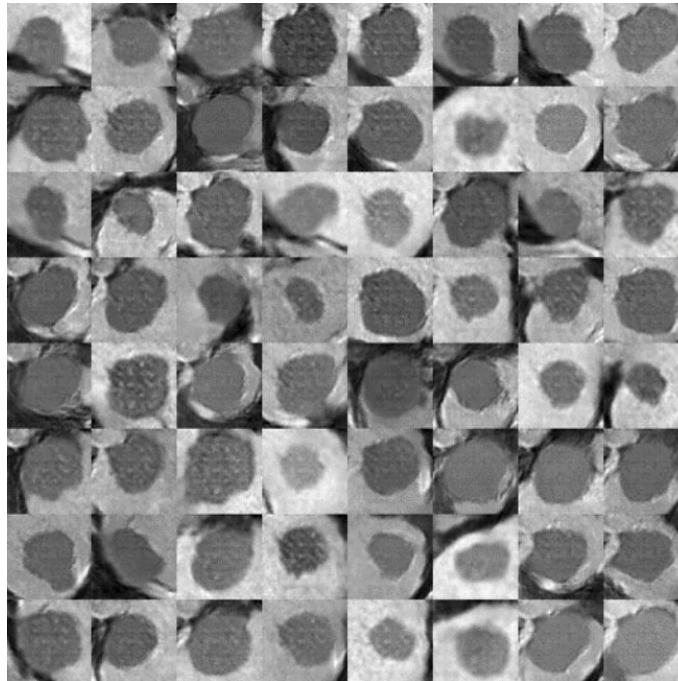


Speech emotion recognition

Available from: <https://pdfs.semanticscholar.org/395b/ea6f025e599db710893acb6321e2a1898a1f.pdf>

Use Cases

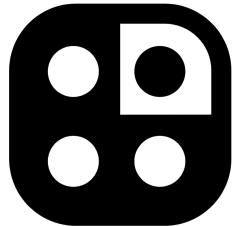
Synthetic liver
lesions



Available from: <https://arxiv.org/abs/1803.01229>

Summary

- Use GANs to generate fake data when real data is too scarce
- GANs have various use cases in data augmentation and beyond!



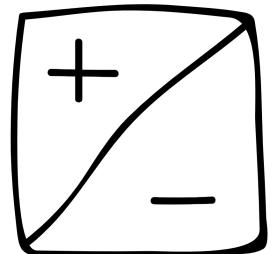


deeplearning.ai

Data Augmentation: Pros & Cons

Outline

- Pros and cons of data augmentation
- Various use cases



Pros of GAN Data Augmentation

Better than hand-crafted
synthetic examples

Synthetic



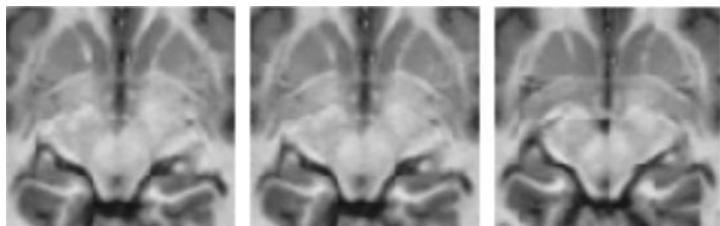
GAN refined



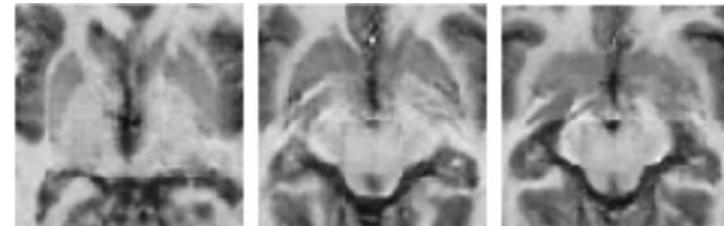
Available from: <https://arxiv.org/abs/1711.09767>

Pros of GAN Data Augmentation

Generate more labeled examples



Training set
(reals)

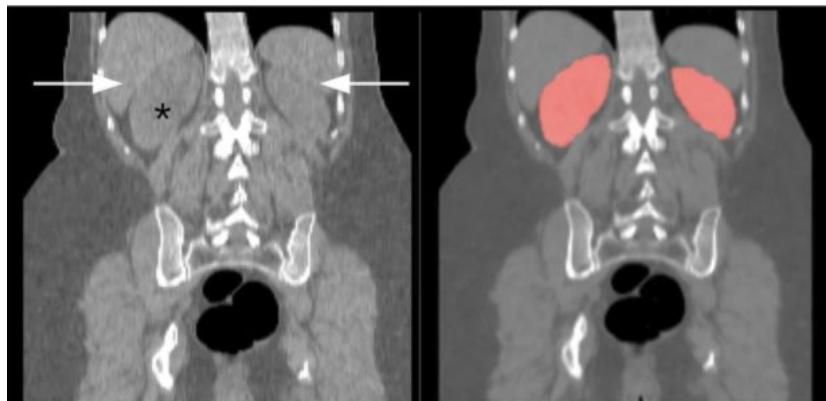


Labeled output
(fakes)

Available from: <https://arxiv.org/abs/1811.10669>

Pros of GAN Data Augmentation

Improve downstream model generalization



CT



Expert



CycleGAN

Standard
Augmentation

Available from: <https://www.nature.com/articles/s41598-019-52737-x/figures/3>

Cons of GAN Data Augmentation

Diversity is limited to the data available



Training set



Generated outputs

Cons of GAN Data Augmentation

Not useful when overfit to real data



Real

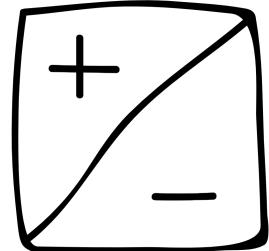


Fake

Available from: <https://arxiv.org/abs/1902.04202>

Summary

- Pros:
 - Can be better than hand-crafted synthetic examples
 - Can generate more labeled examples
 - Can improve a downstream model's generalization
- Cons:
 - Can be limited by the available data in diversity
 - Can overfit to the real training data





deeplearning.ai

GANs for Privacy

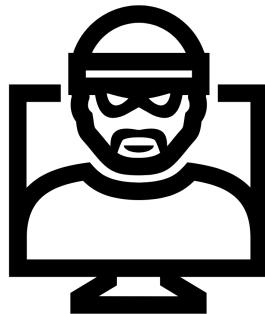
Outline

- GANs for privacy preservation
- Medical privacy as a motivating example



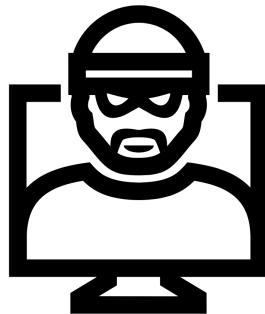
Motivations for Medical Privacy

- Protects real patient data



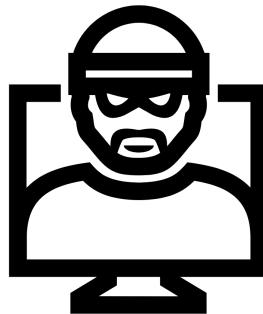
Motivations for Medical Privacy

- Protects real patient data
- Can encourage data-sharing between institutions



Motivations for Medical Privacy

- Protects real patient data
- Can encourage data-sharing between institutions
- Less expensive and more abundant than real data



Privacy Preservation

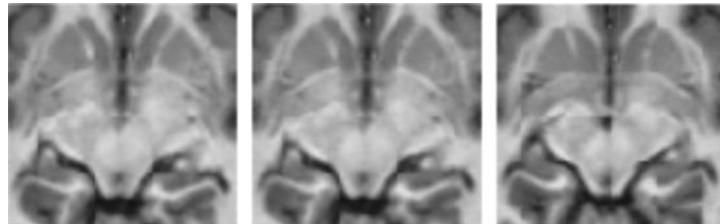
GAN tissue patches look real to pathologists



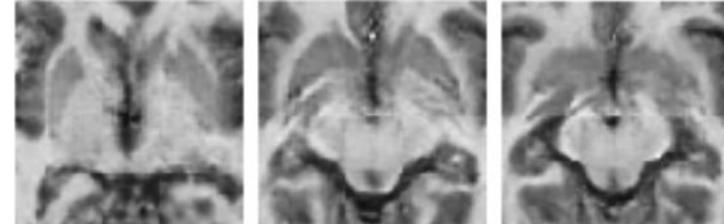
Available from: <https://twitter.com/realSharonZhou/status/1182877446690852867>

Privacy Preservation

GAN MRIs look realistic



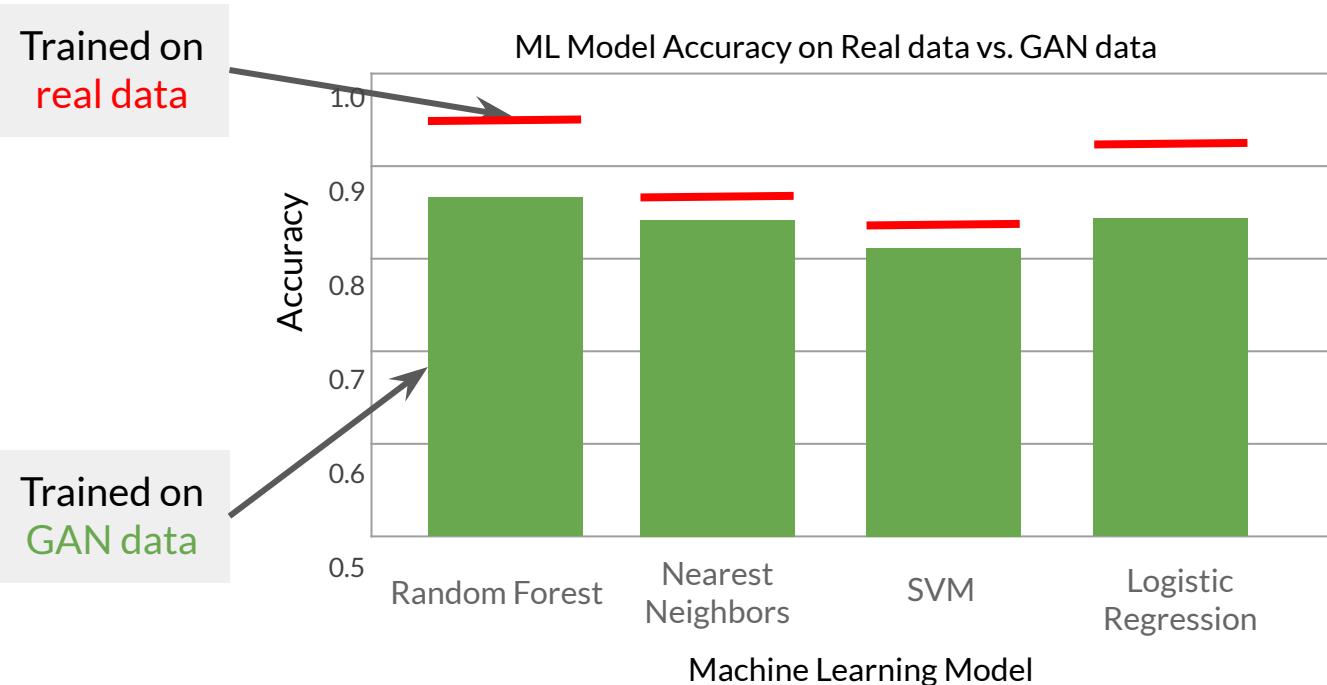
Training set
(reals)



Labeled output
(fakes)

Available from: <https://arxiv.org/abs/1811.10669>

Pro of GANs for Privacy

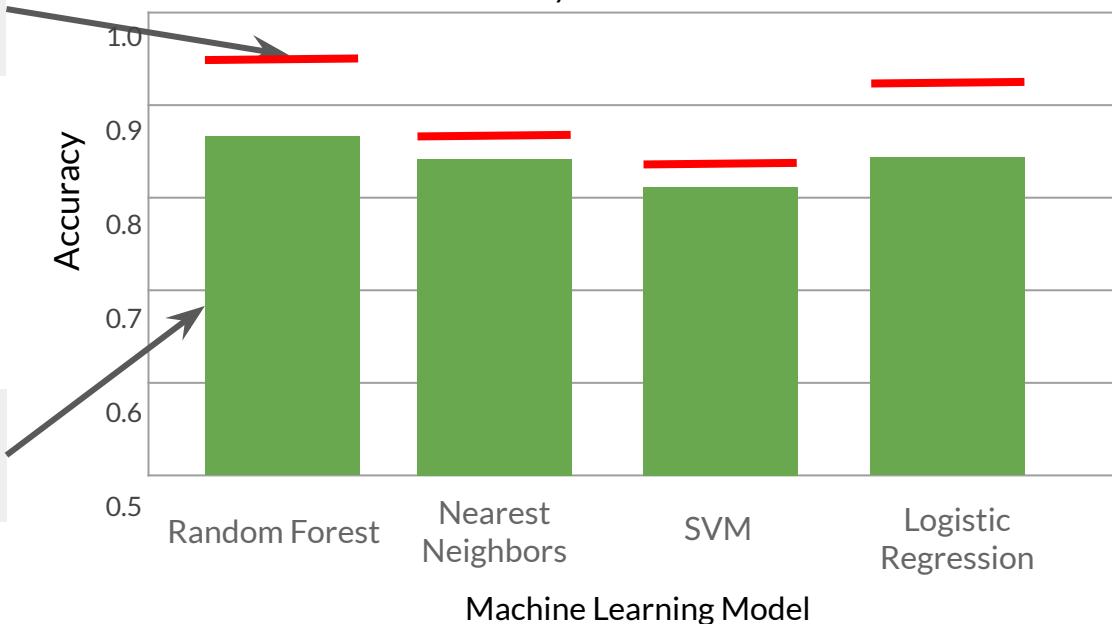


Available from: <https://www.ahajournals.org/doi/epub/10.1161/CIRCOUTCOMES.118.005122>

Pro of GANs for Privacy

Trained on
real data

ML Model Accuracy on Real data vs. GAN data



Trained on
GAN data

Training with GAN
data approaches
real data accuracy

Available from: <https://www.ahajournals.org/doi/epub/10.1161/CIRCOUTCOMES.118.005122>

Con of GANs for Privacy

GAN sample is nearly identical to a real sample



Available from: <https://arxiv.org/abs/1902.04202>

Summary

- GANs can be useful for preserving privacy
 - Sensitive medical data serves as one example
- Caveat: generated samples may mimic the reals too closely
 - Post-processing may help avoid this data leakage



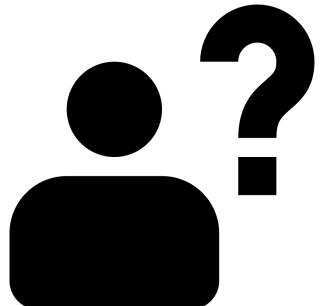


deeplearning.ai

GANs for Anonymity

Outline

- GANs for anonymity
 - Concealing identity
 - Stealing identity
 - DeepFakes



Anonymity



Original image

De-identified image

Available from: <https://arxiv.org/abs/1902.04202>

Anonymity



Available from: <https://arxiv.org/abs/1909.04538>

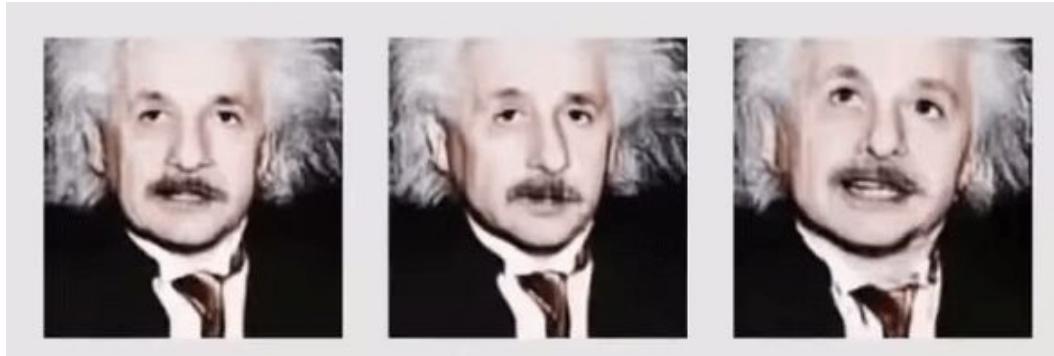
Pro of GANs for Anonymity

- Provide safe environment for expression to:
 - Stigmatized groups
 - Assault victims
 - Witnesses
 - Activists



Con of GANs for Anonymity

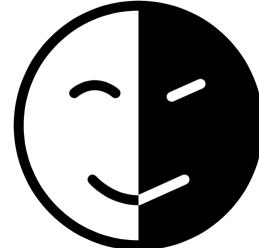
Deepfakes put words into people's mouths



Available from: <https://arxiv.org/abs/1905.08233>

Summary

- GANs can enable healthy anonymous expression for stigmatized groups
- GANs for anonymization can be used for good or evil
 - Identity theft is not good
 - Use your powers for good



Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>



deeplearning.ai

Image-to-Image Translation

Outline

- Image-to-image translation
- Other types of translation

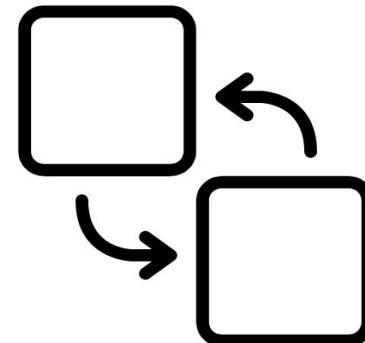
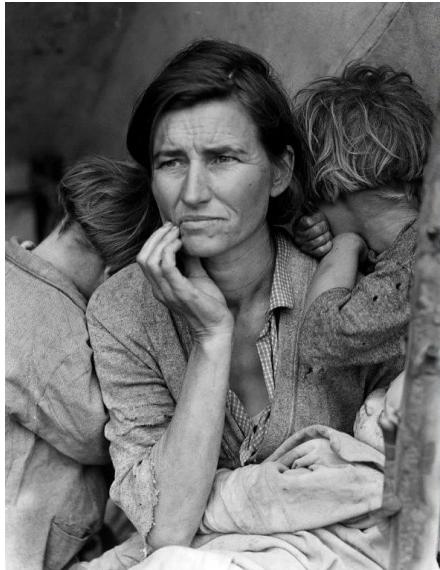
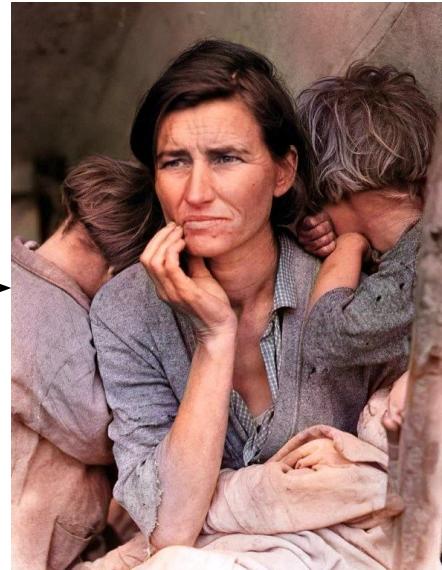


Image-to-Image Translation



Transformation



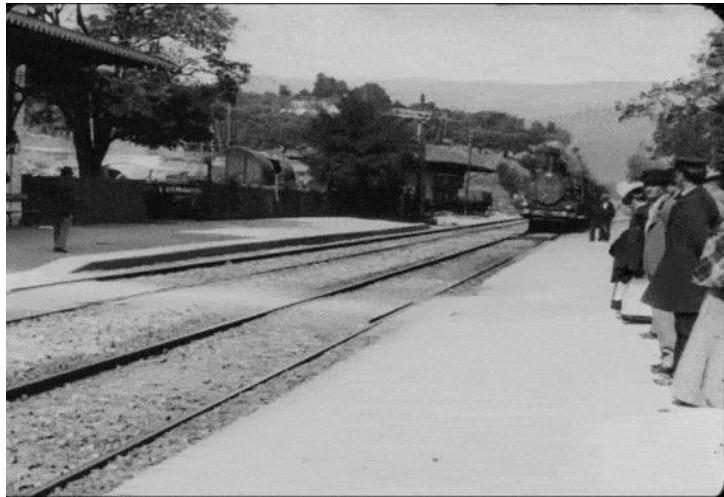
Available from: <https://twitter.com/citnaj/status/1124904251128406016>

Image-to-Image Translation



Available from: <https://github.com/NVIDIA/pix2pixHD>

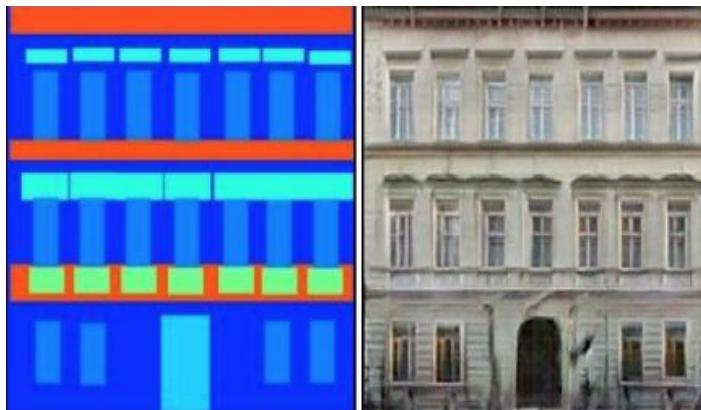
Image-to-Image Translation



Available from: <https://youtu.be/3RYNThid23g>

Paired Image-to-Image Translation

Labels to facade



Black-and-white to color



Available from: <https://arxiv.org/abs/1611.07004>

Paired Image-to-Image Translation

Day to night



Edges to photo

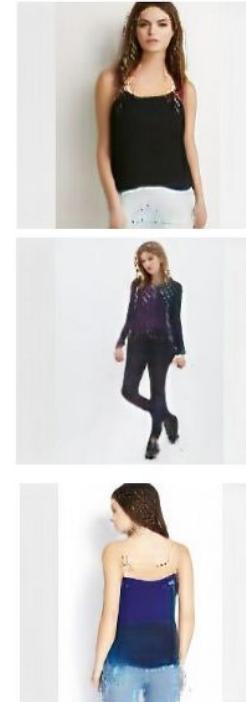


Available from: <https://arxiv.org/abs/1611.07004>

Paired Image-to-Image Translation



Clothes and pose to
pose with clothes



Available from: <https://arxiv.org/abs/1705.09368>

Other Translations

“This bird is red with white and has a very short beak”



Available from: <https://arxiv.org/abs/1711.10485>

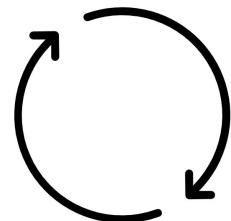
Other Translations



Available from: <https://arxiv.org/abs/1905.08233>

Summary

- Image-to-image translation transforms images into different styles
- GANs' realistic generation abilities are well-suited to image-to-image translation tasks
- Other types of translation include text-to-image or image-to-video



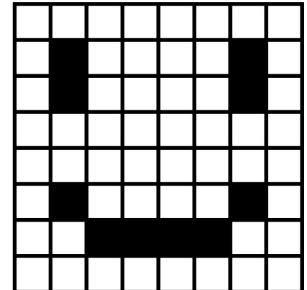


deeplearning.ai

Pix2Pix Overview

Outline

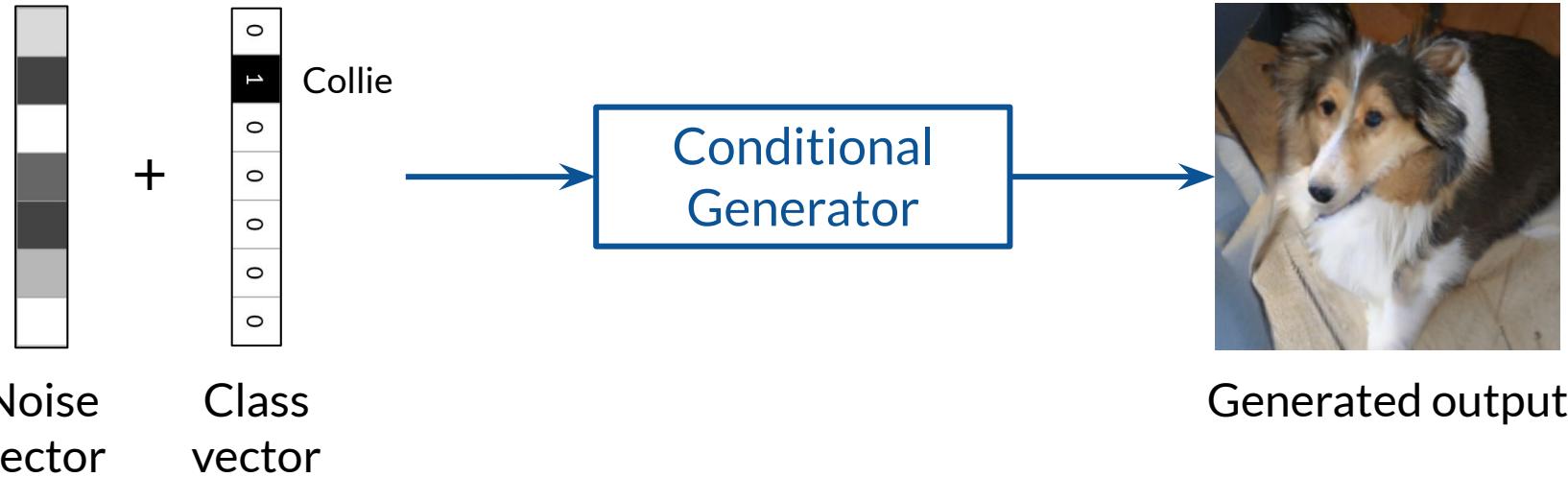
- Overview of Pix2Pix
- Comparison with conditional GAN
- Upgraded generator and discriminator architectures



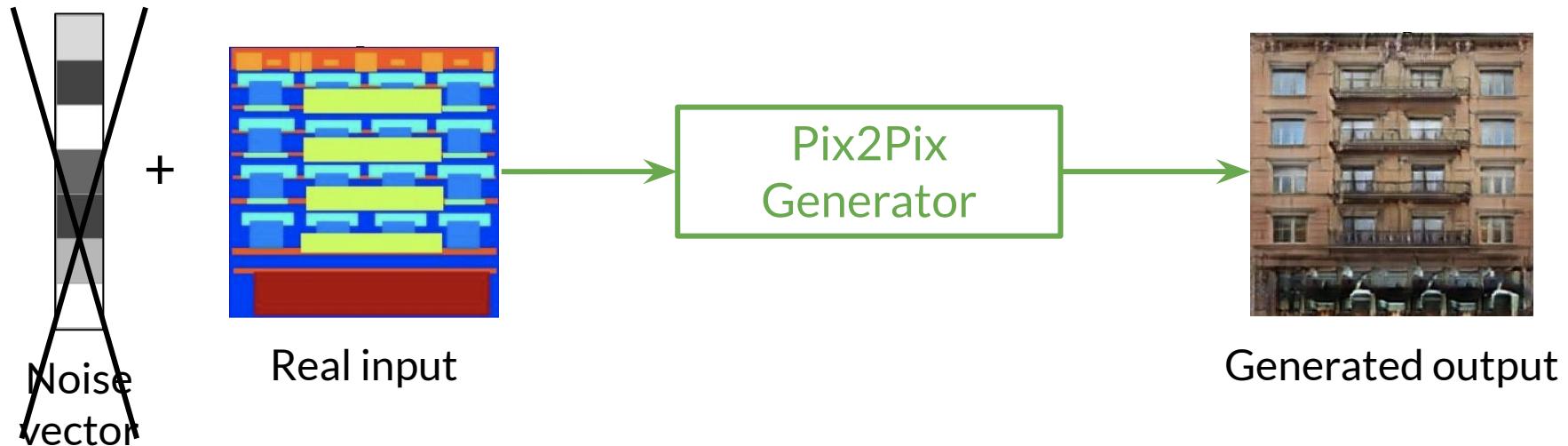
Pix2Pix for Paired Image-to-Image Translation

Image-to-Image \longrightarrow *Pix*-to-*Pix* \longrightarrow *Pix2Pix*

Pix2Pix Generator

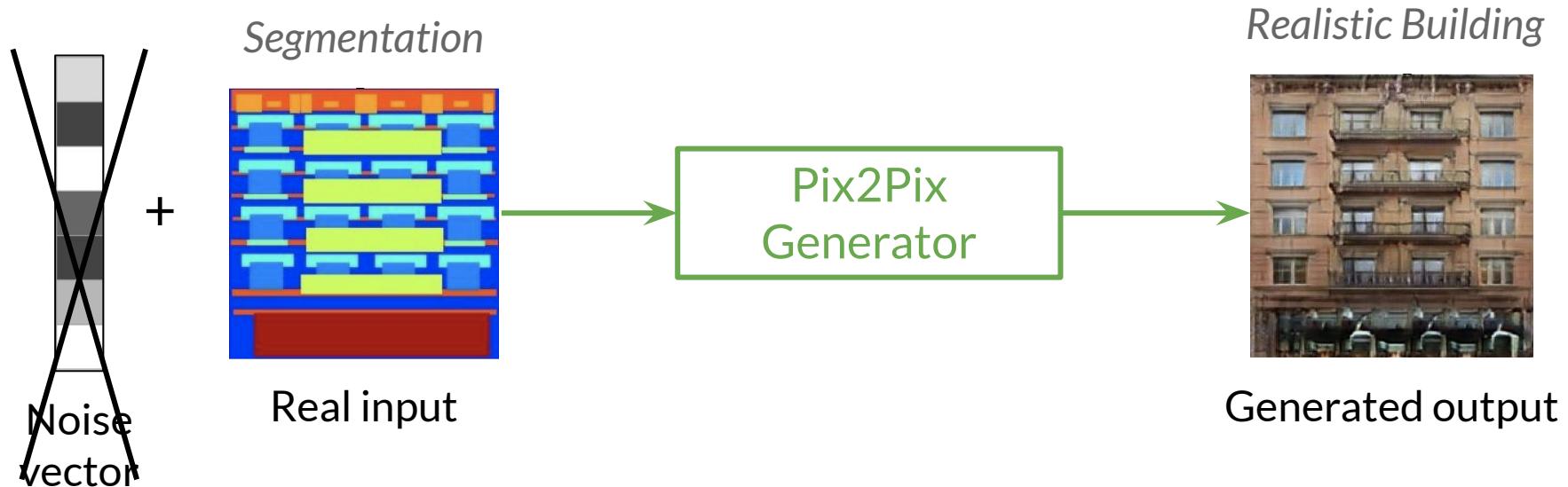


Pix2Pix Generator



Available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix Generator



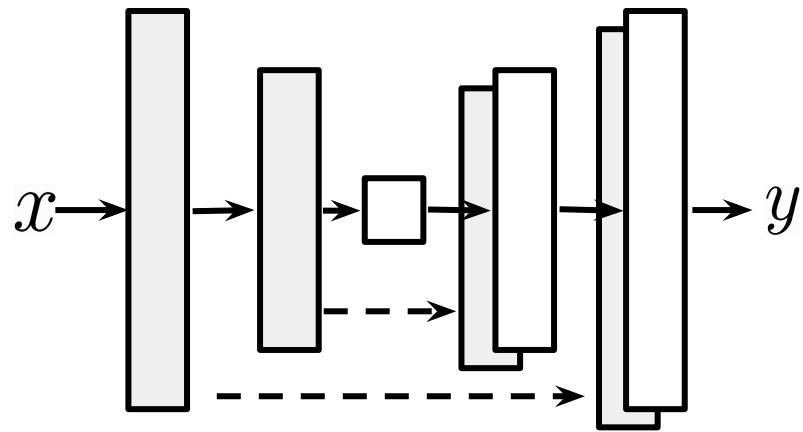
Available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix Discriminator



Available from: <https://arxiv.org/abs/1611.07004>

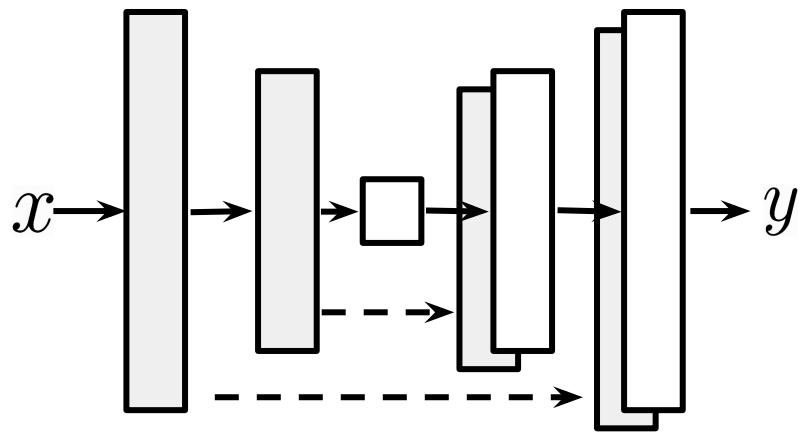
Pix2Pix Upgrades



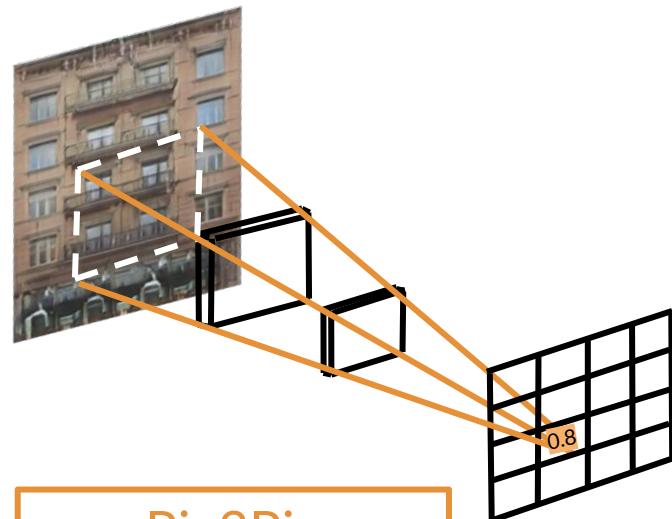
Pix2Pix
Generator

Based on: <https://arxiv.org/abs/1611.07004>

Pix2Pix Upgrades



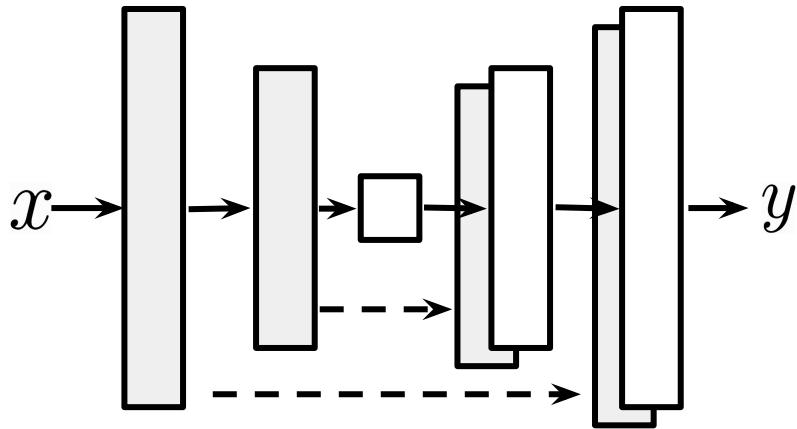
Pix2Pix
Generator



Pix2Pix
Discriminator

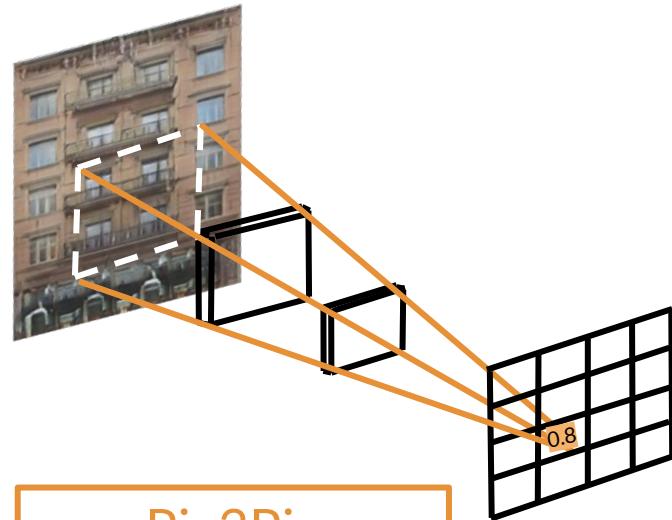
(Left) Based on: <https://arxiv.org/abs/1611.07004>
(Right) Based on: <https://arxiv.org/abs/1803.07422>

Pix2Pix Upgrades



Pix2Pix
Generator

Goal is still to produce realistic outputs!

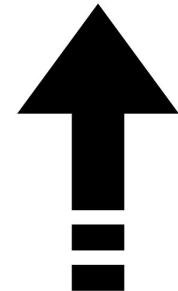


Pix2Pix
Discriminator

(Left) Based on: <https://arxiv.org/abs/1611.07004>
(Right) Based on: <https://arxiv.org/abs/1803.07422>

Summary

- Inputs and outputs of Pix2Pix are similar to a conditional GAN
 - Take in the original image, instead of the class vector
 - No explicit noise as input
- Generator and discriminator models are upgraded



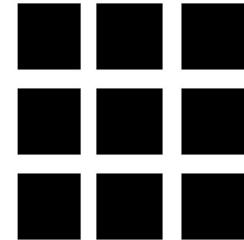


deeplearning.ai

Pix2Pix: PatchGAN

Outline

- PatchGAN discriminator architecture
- Matrix output vs. single output



Pix2Pix Discriminator: PatchGAN

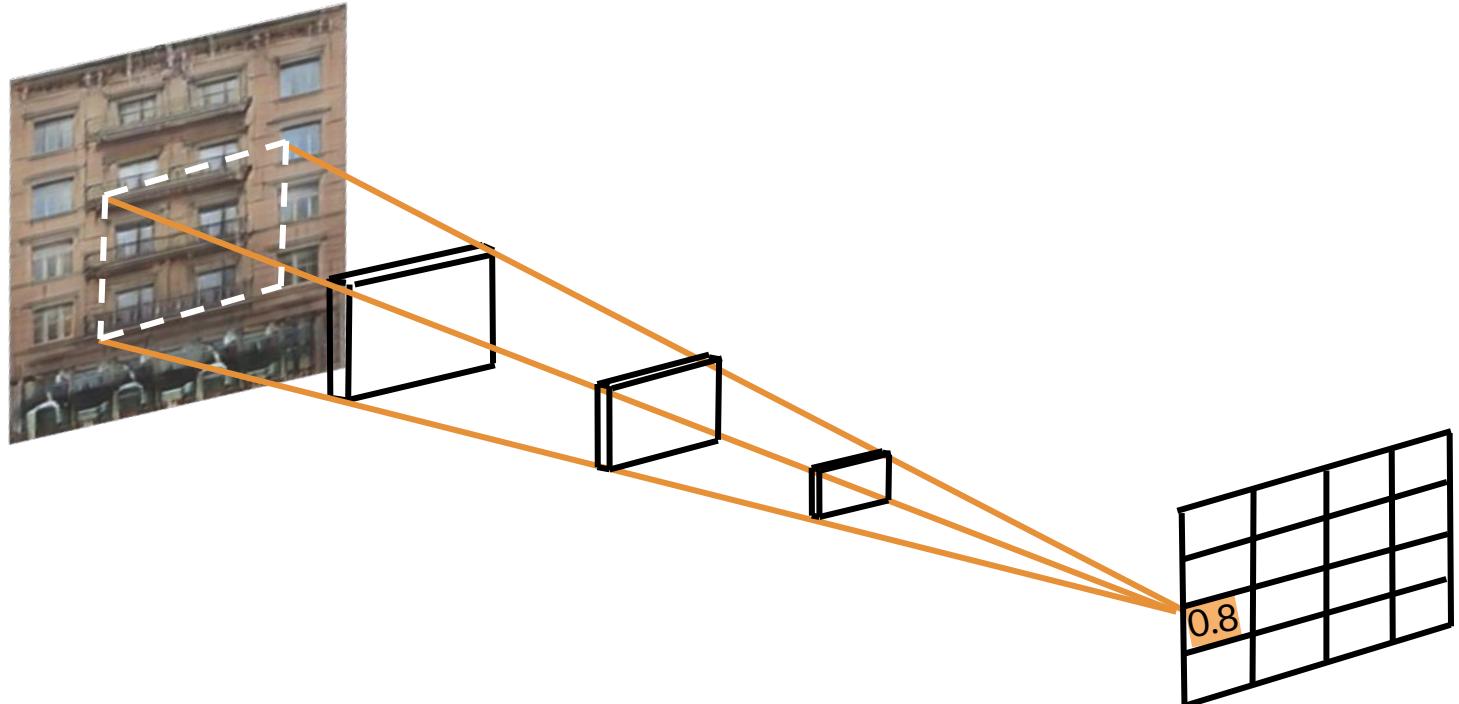


Image available from: <https://arxiv.org/abs/1611.07004>
Based on: <https://arxiv.org/abs/1803.07422>

Pix2Pix Discriminator: PatchGAN

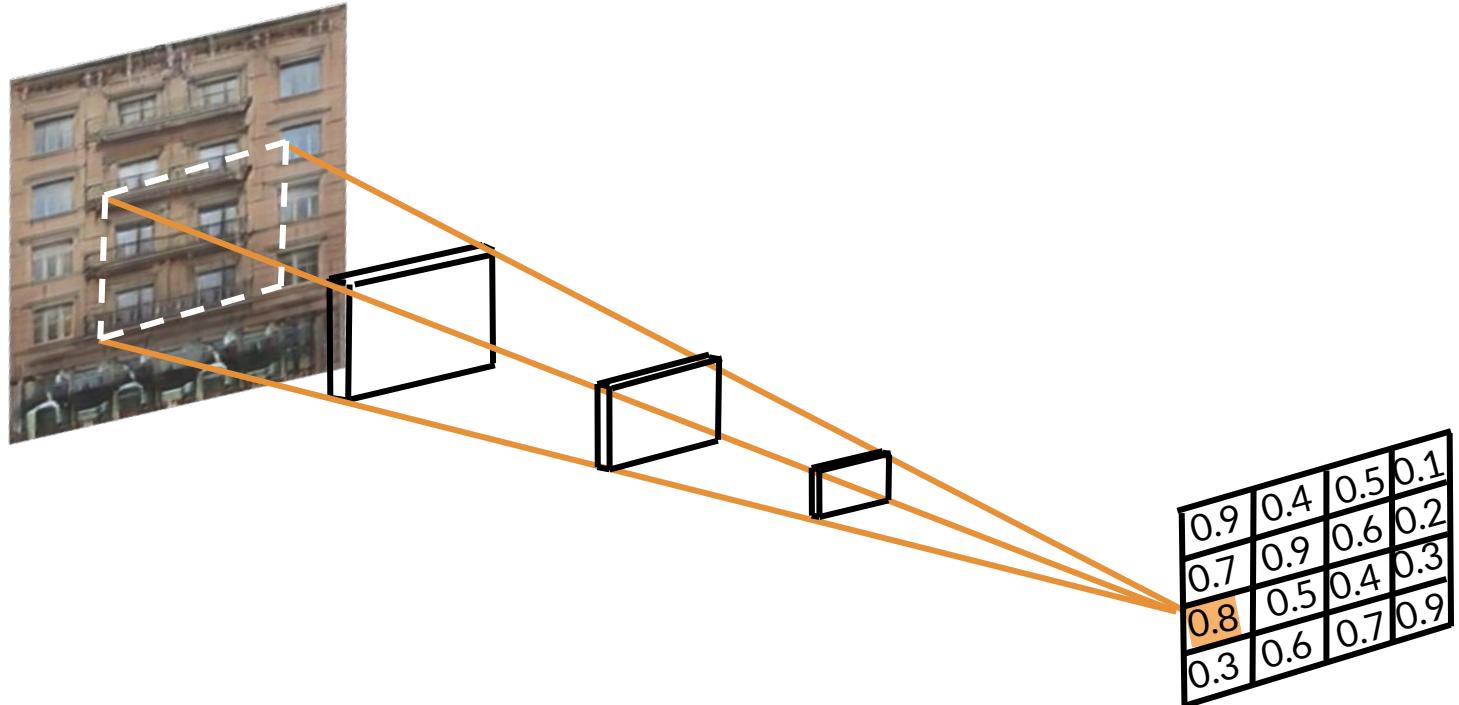


Image available from: <https://arxiv.org/abs/1611.07004>

Based on: <https://arxiv.org/abs/1803.07422>

Pix2Pix Discriminator: PatchGAN

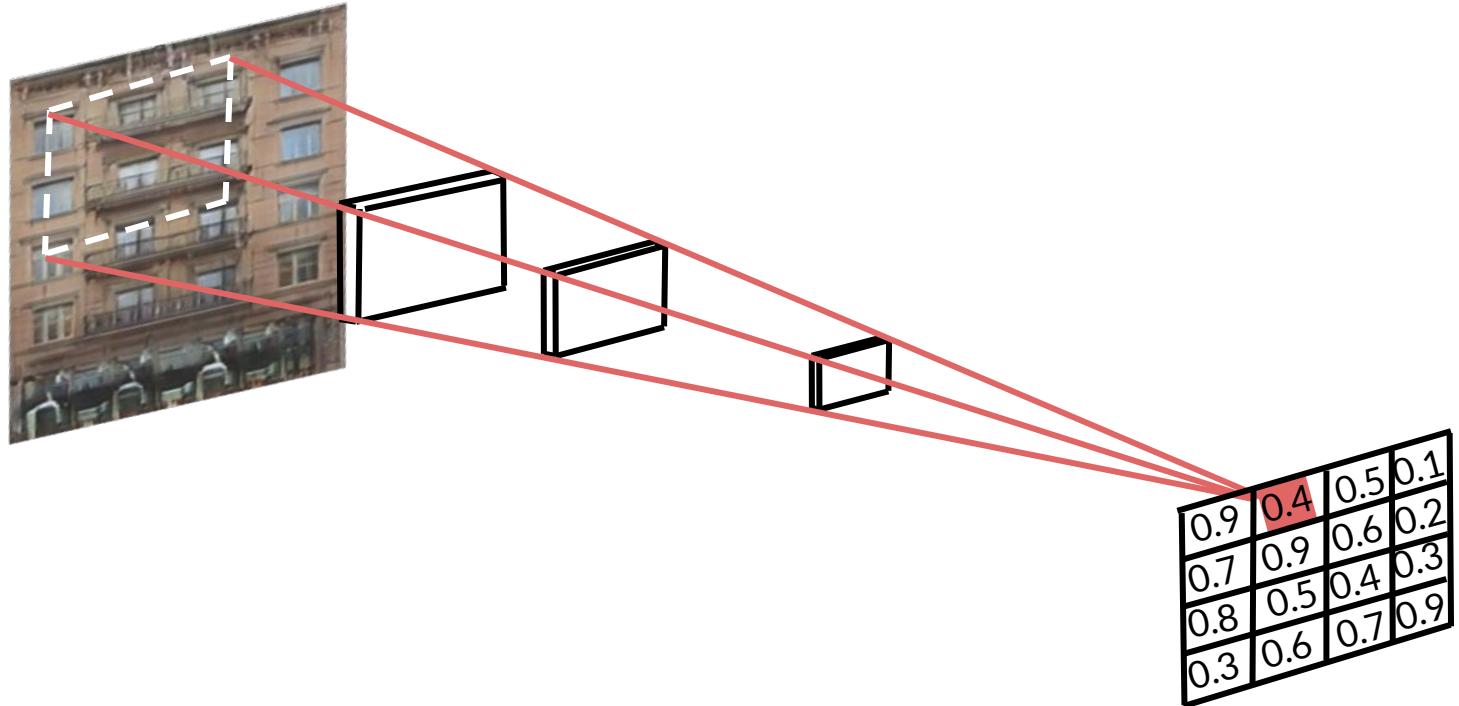


Image available from: <https://arxiv.org/abs/1611.07004>

Based on: <https://arxiv.org/abs/1803.07422>

Pix2Pix Discriminator: PatchGAN

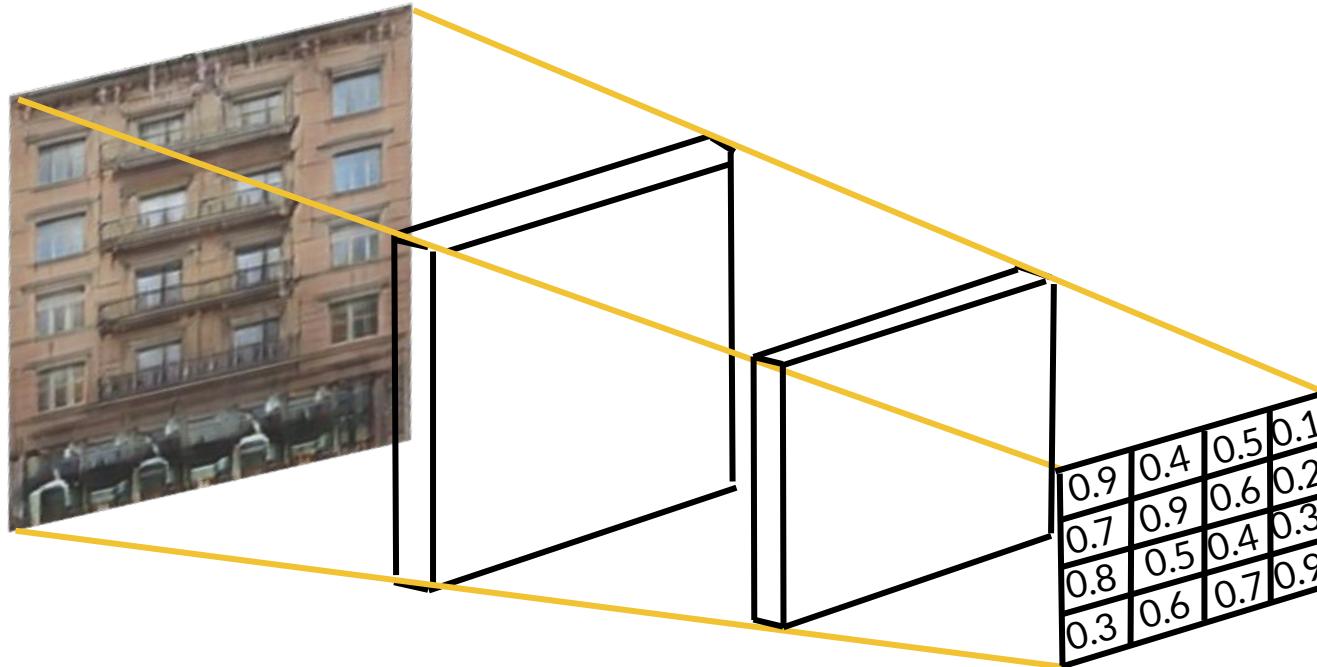
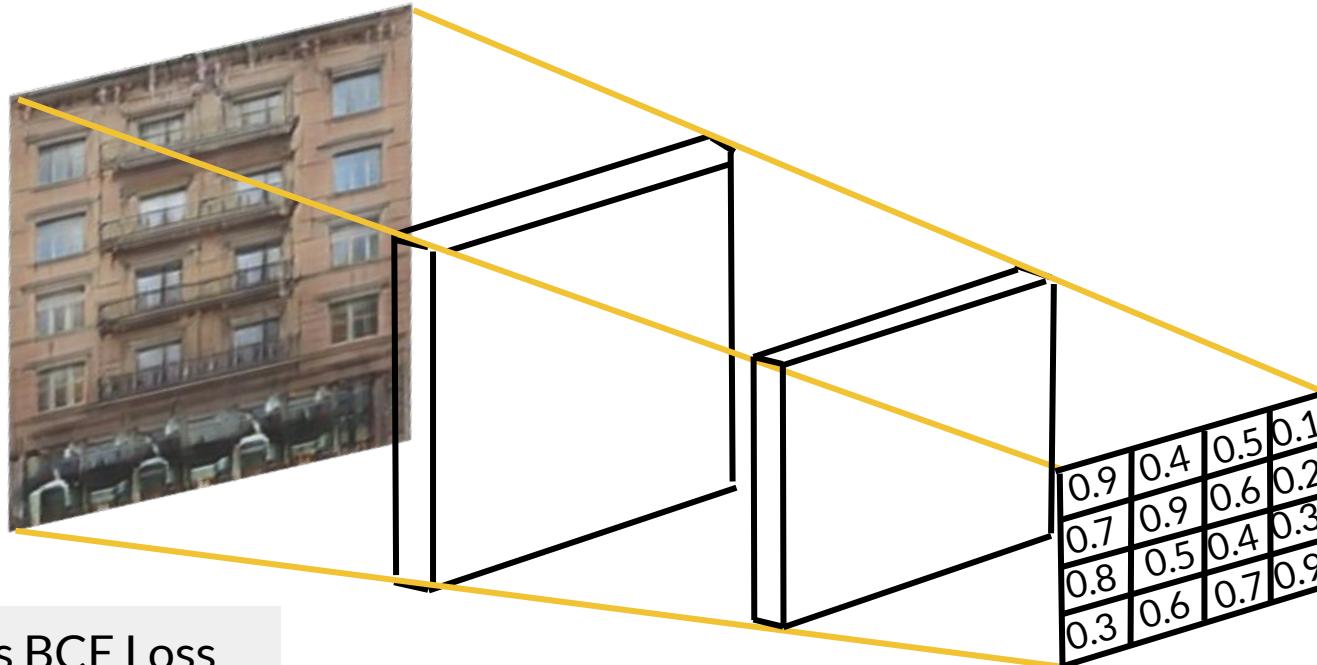


Image available from: <https://arxiv.org/abs/1611.07004>

Based on: <https://arxiv.org/abs/1803.07422>

Pix2Pix Discriminator: PatchGAN



Still uses BCE Loss

Image available from: <https://arxiv.org/abs/1611.07004>

Based on: <https://arxiv.org/abs/1803.07422>

Pix2Pix Discriminator: PatchGAN

Values closer to $0 = \text{fake}$ label

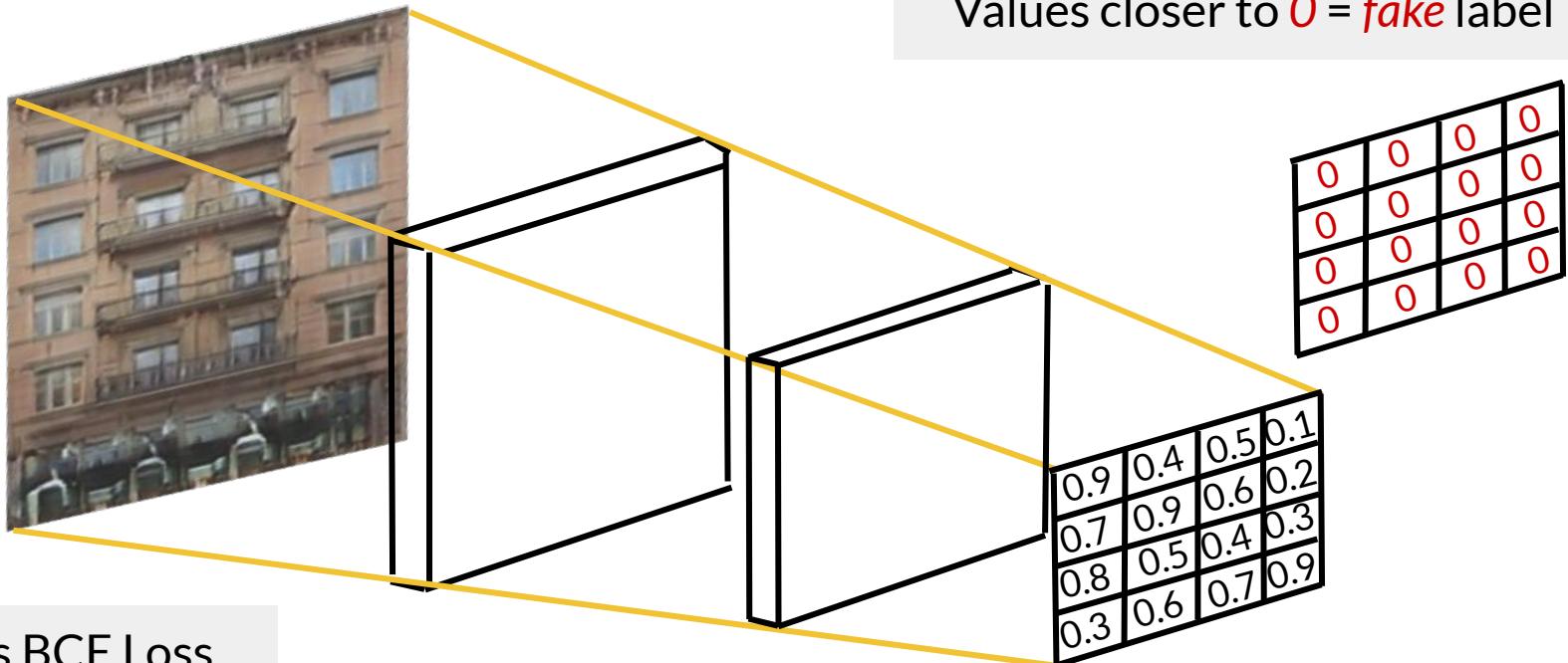
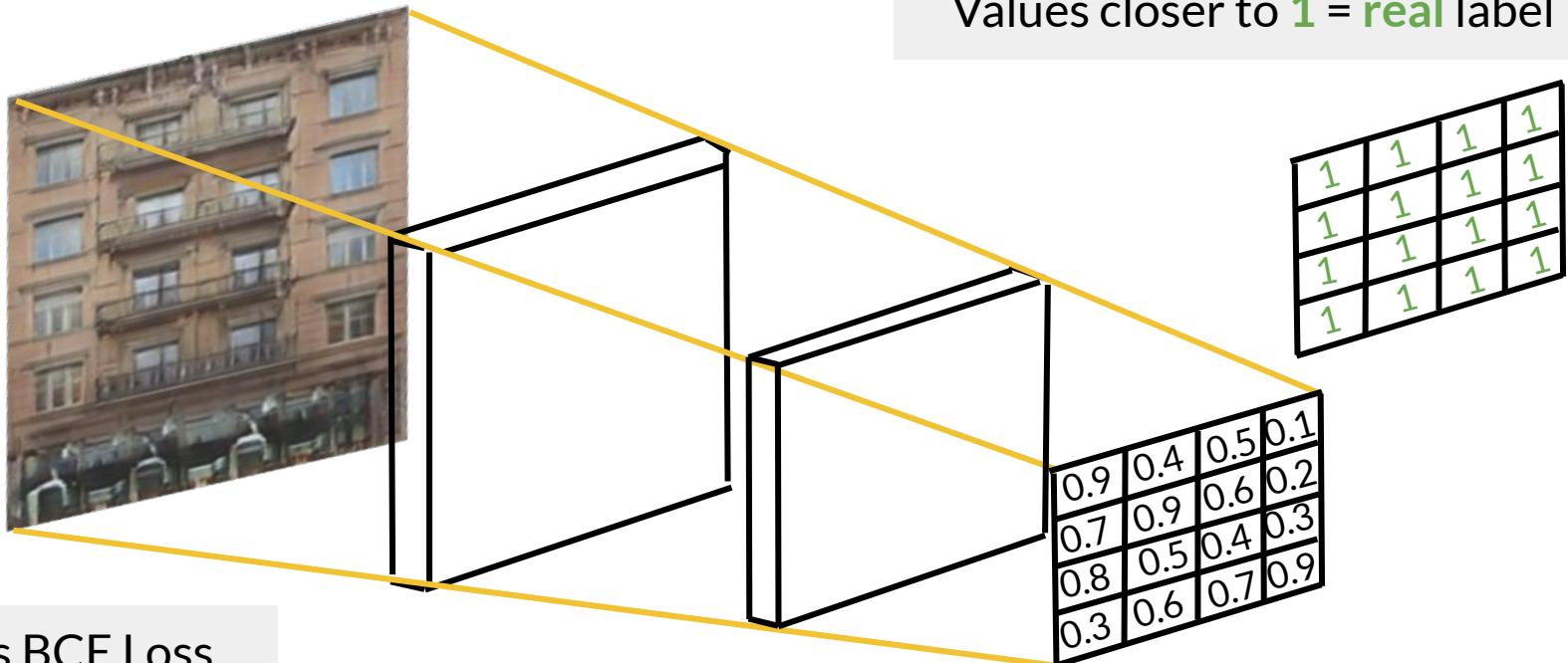


Image available from: <https://arxiv.org/abs/1611.07004>

Based on: <https://arxiv.org/abs/1803.07422>

Pix2Pix Discriminator: PatchGAN

Values closer to **1** = **real** label



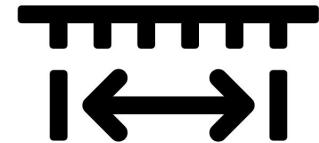
Still uses BCE Loss

Image available from: <https://arxiv.org/abs/1611.07004>

Based on: <https://arxiv.org/abs/1803.07422>

Summary

- PatchGAN discriminator outputs a matrix of values, each between 0 and 1
- Label matrices:
 - 0's = fake
 - 1's = real





deeplearning.ai

Pix2Pix: U-Net

Outline

- Net framework
 - Encoder-Decoder
- U-Skip connections
- Pix2Pix generator

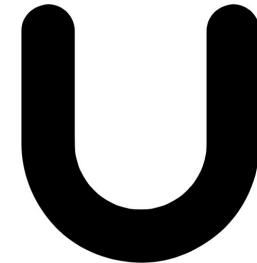


Image Segmentation

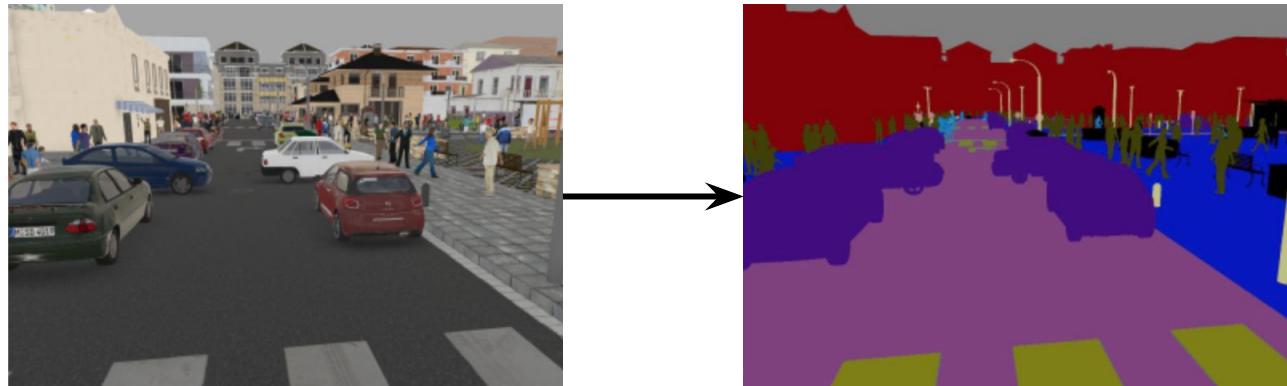


Image Segmentation

Available from: <https://developer.nvidia.com/blog/image-segmentation-using-digits-5/>

Image Segmentation

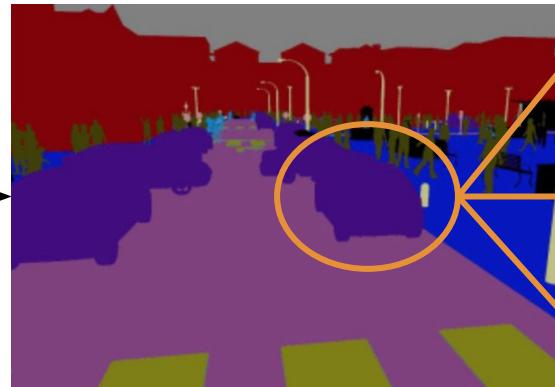
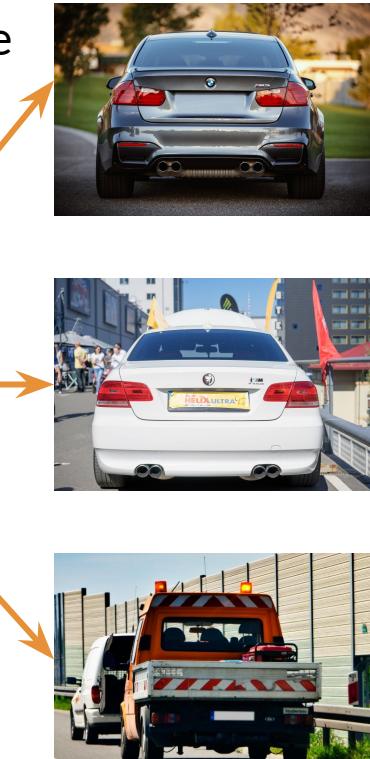


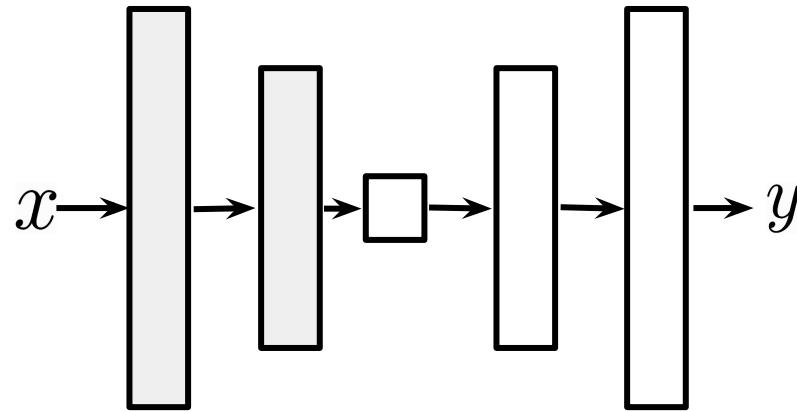
Image Segmentation

Image-to-Image Translation



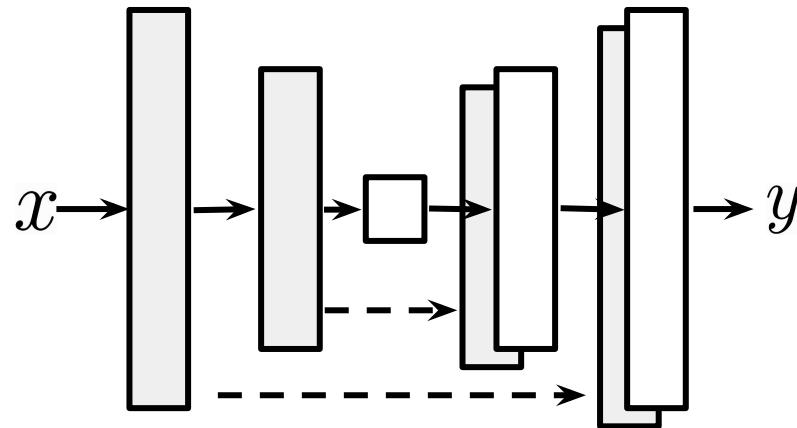
Available from: <https://developer.nvidia.com/blog/image-segmentation-using-digits-5/>

U-Net Framework: Encoder-Decoder



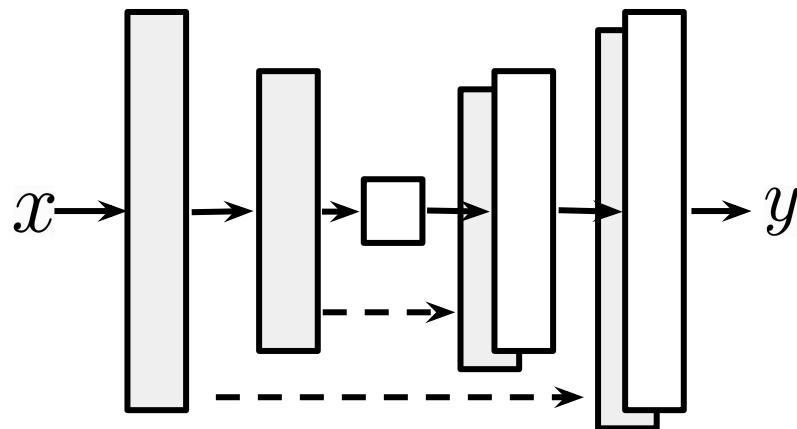
Based on: <https://arxiv.org/abs/1611.07004>

U-Net Framework: Skip Connections



Based on: <https://arxiv.org/abs/1611.07004>

U-Net Framework: Skip Connections



Forward pass

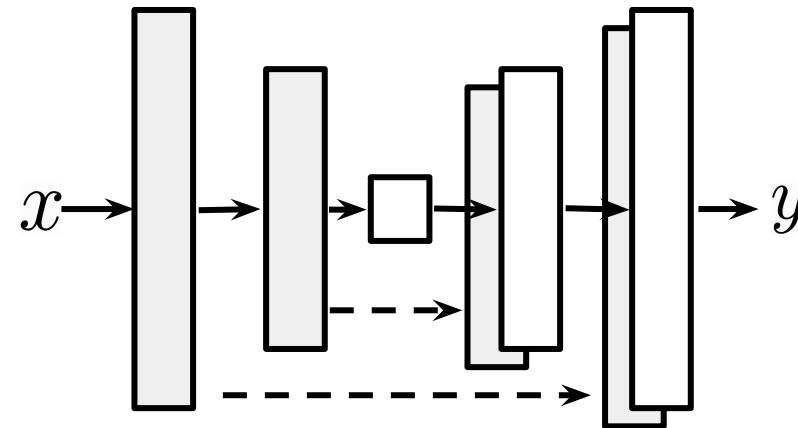
Skip connections
allow information
flow to the decoder

Based on: <https://arxiv.org/abs/1611.07004>

U-Net Framework: Skip Connections

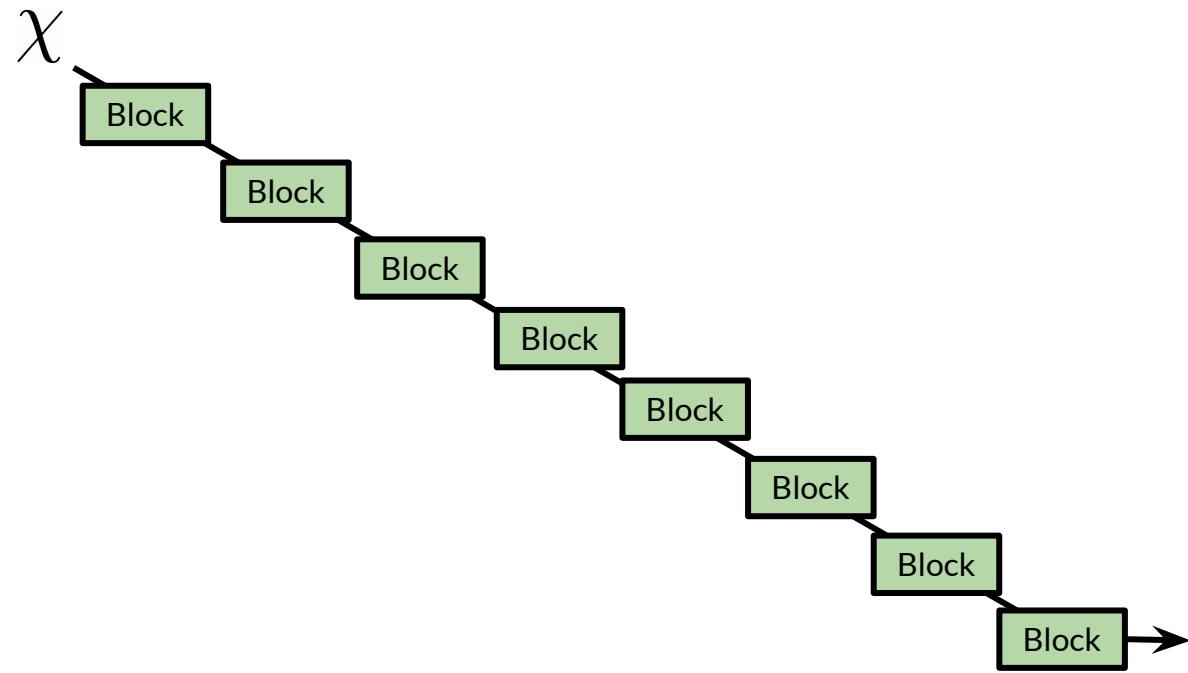
Backward pass

Skip connections
improve gradient
flow to encoder



Based on: <https://arxiv.org/abs/1611.07004>

Pix2Pix Encoder



Pix2Pix Encoder

χ

Input size: 256 x 256 x 3

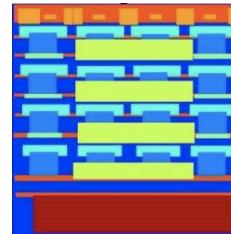
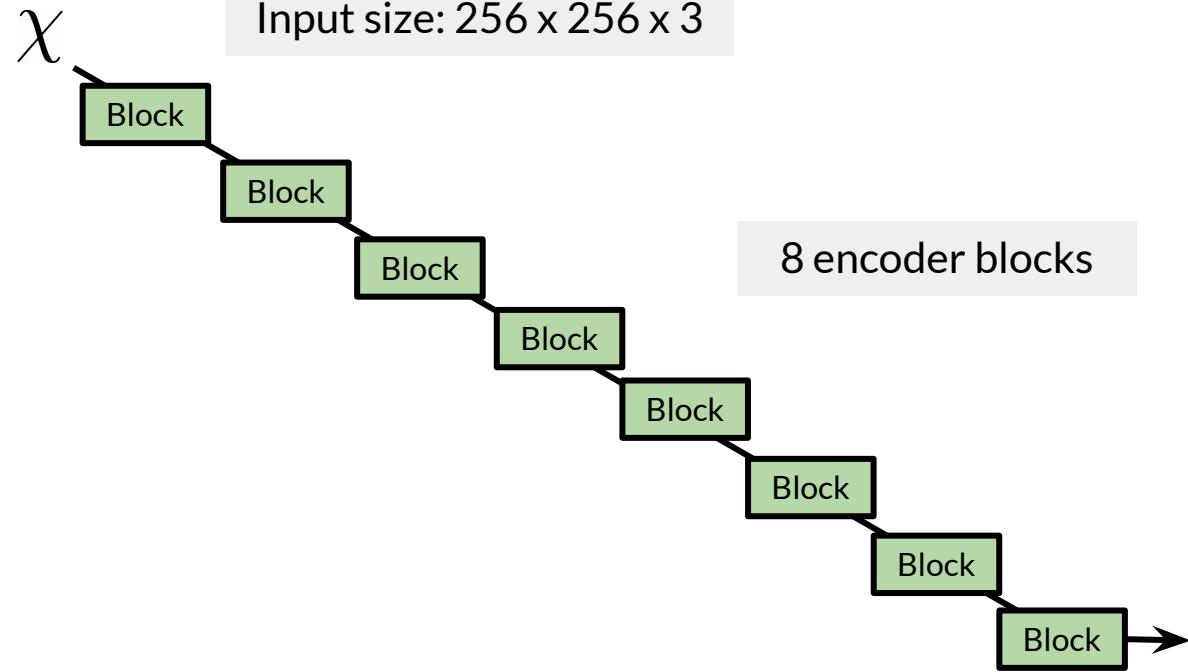


Image available from: <https://arxiv.org/abs/1611.07004>

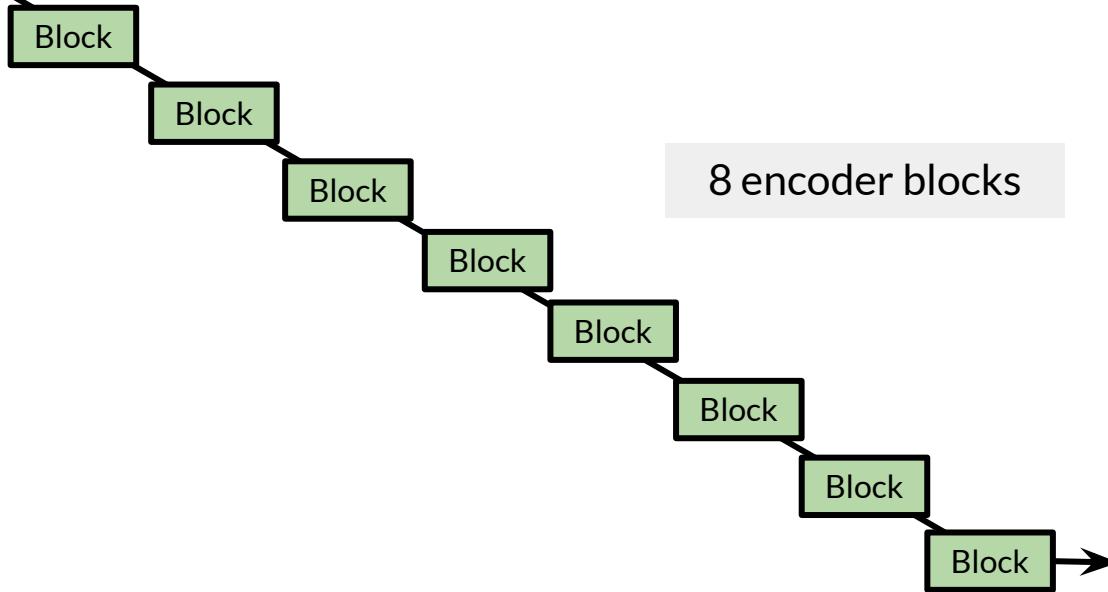
Pix2Pix Encoder



Pix2Pix Encoder

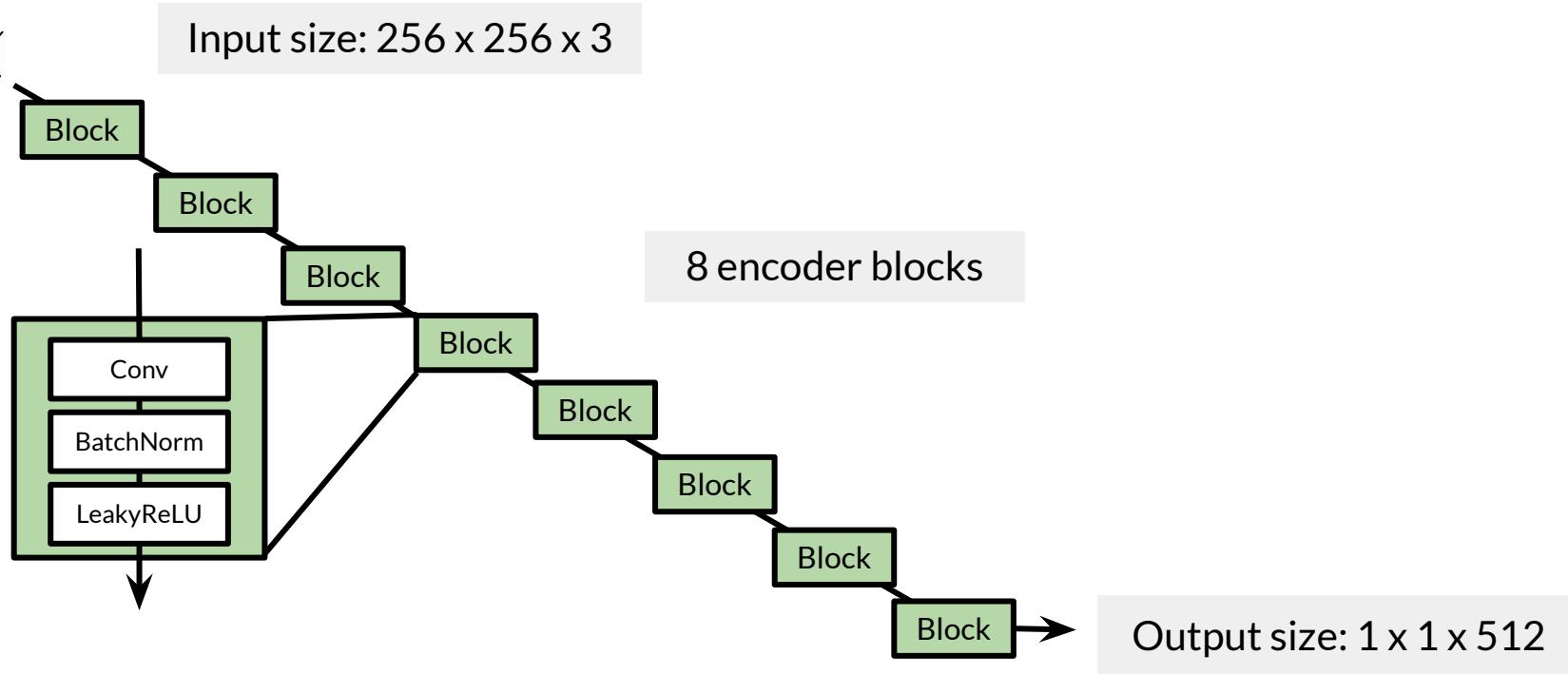
χ

Input size: 256 x 256 x 3

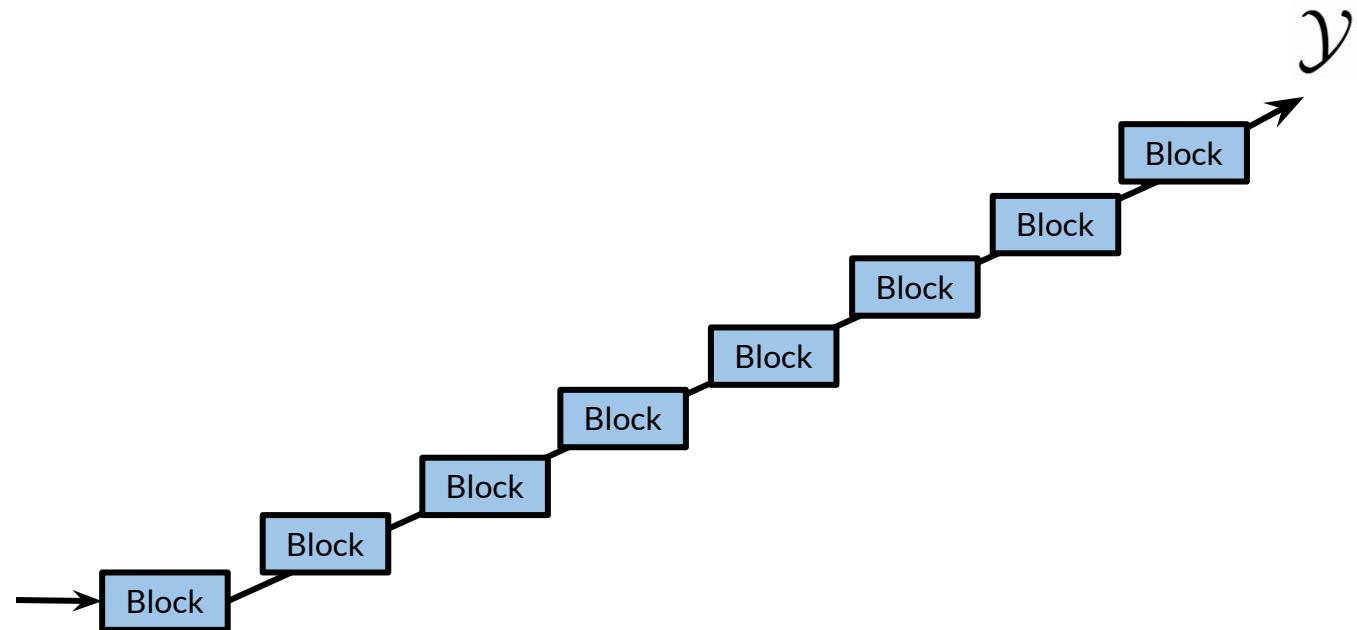


Pix2Pix Encoder

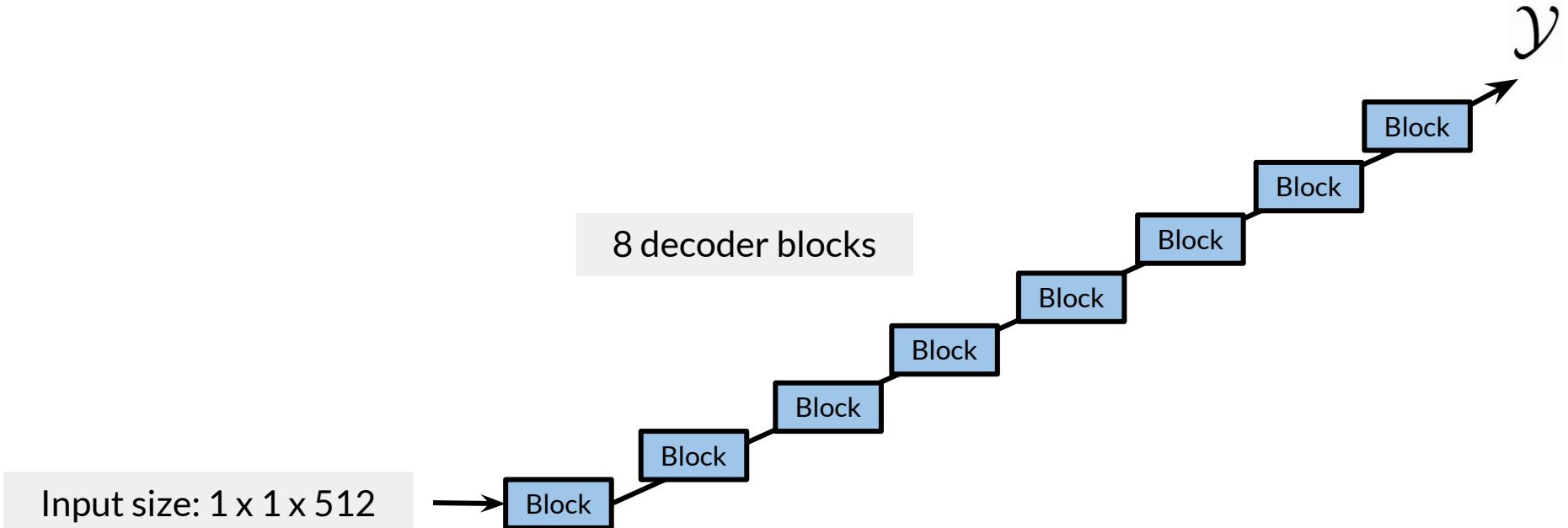
χ



Pix2Pix Decoder



Pix2Pix Decoder

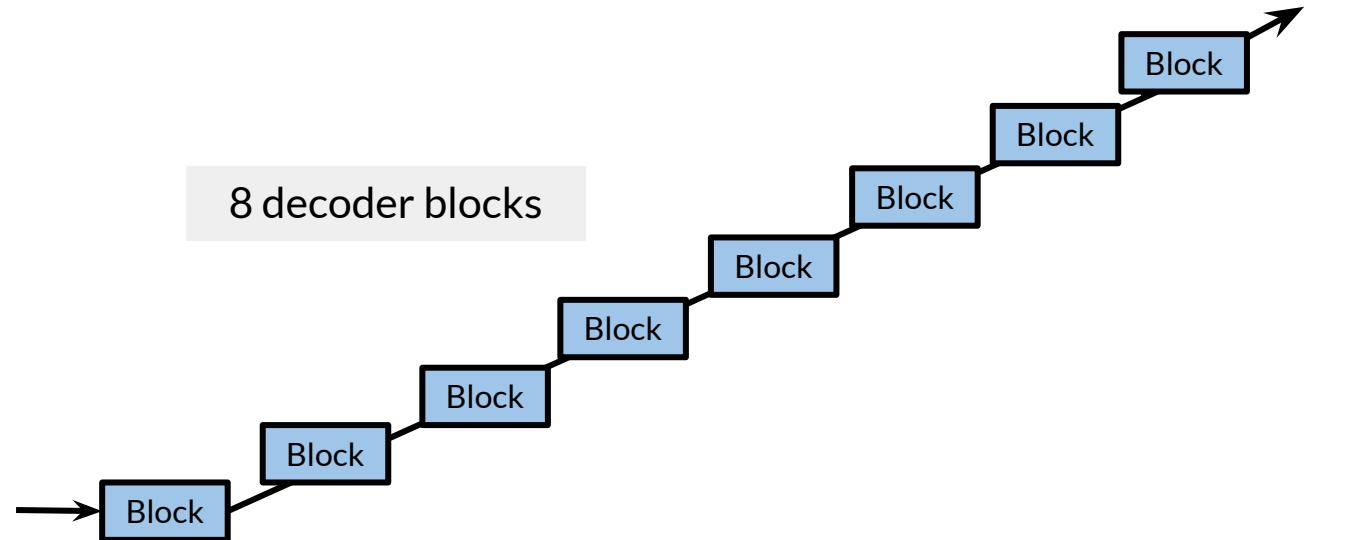


Pix2Pix Decoder

Output size: 256 x 256 x 3



y



Input size: $1 \times 1 \times 512$

Image available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix Decoder

Output size: 256 x 256 x 3



y

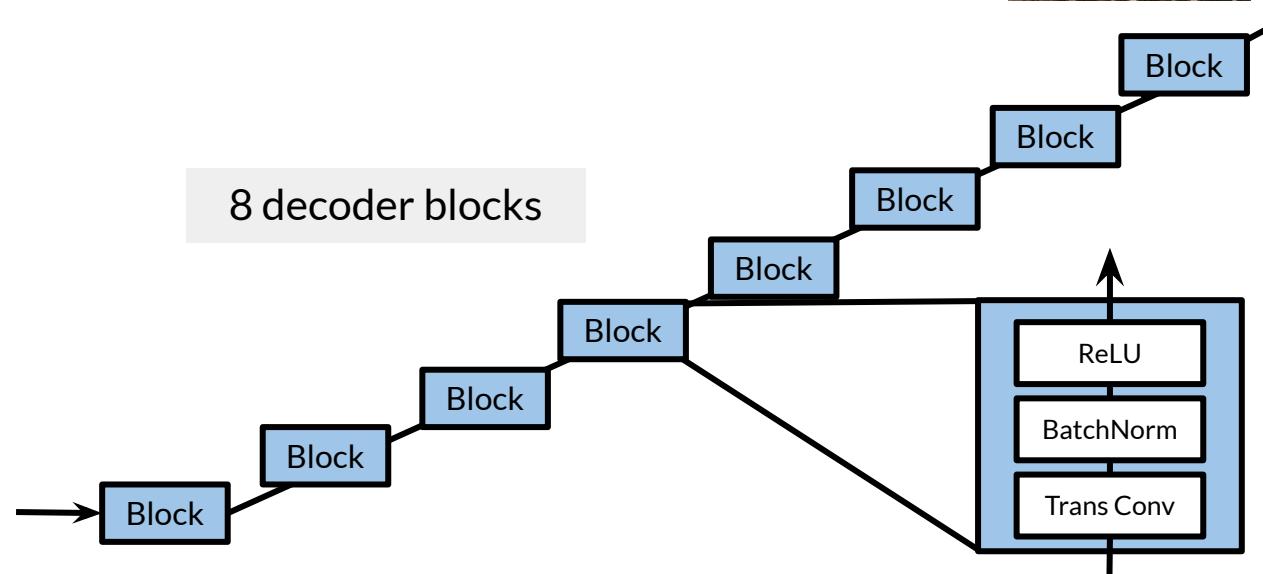


Image available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix Decoder

Output size: 256 x 256 x 3



۲۷۰

Dropout in some decoder blocks
adds noise to the network

Input size: 1 x 1 x 512

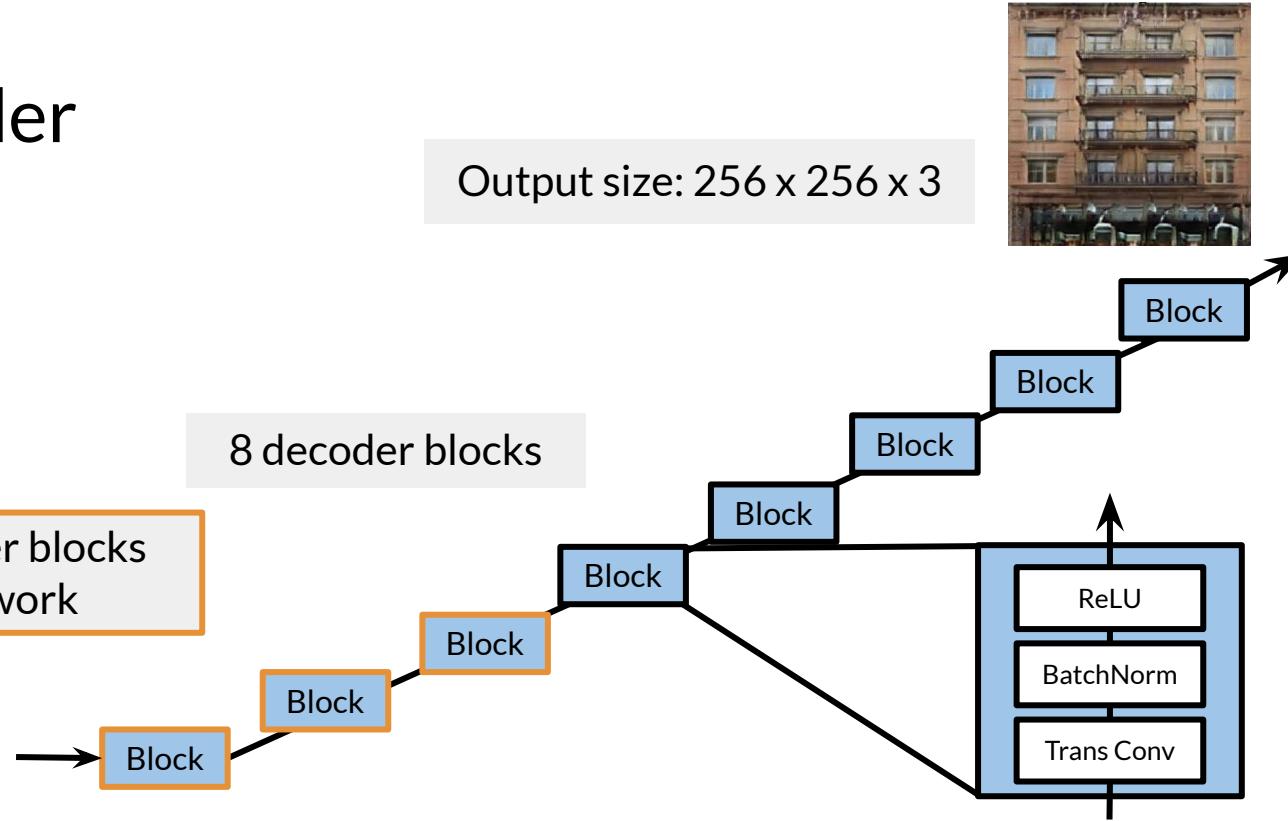
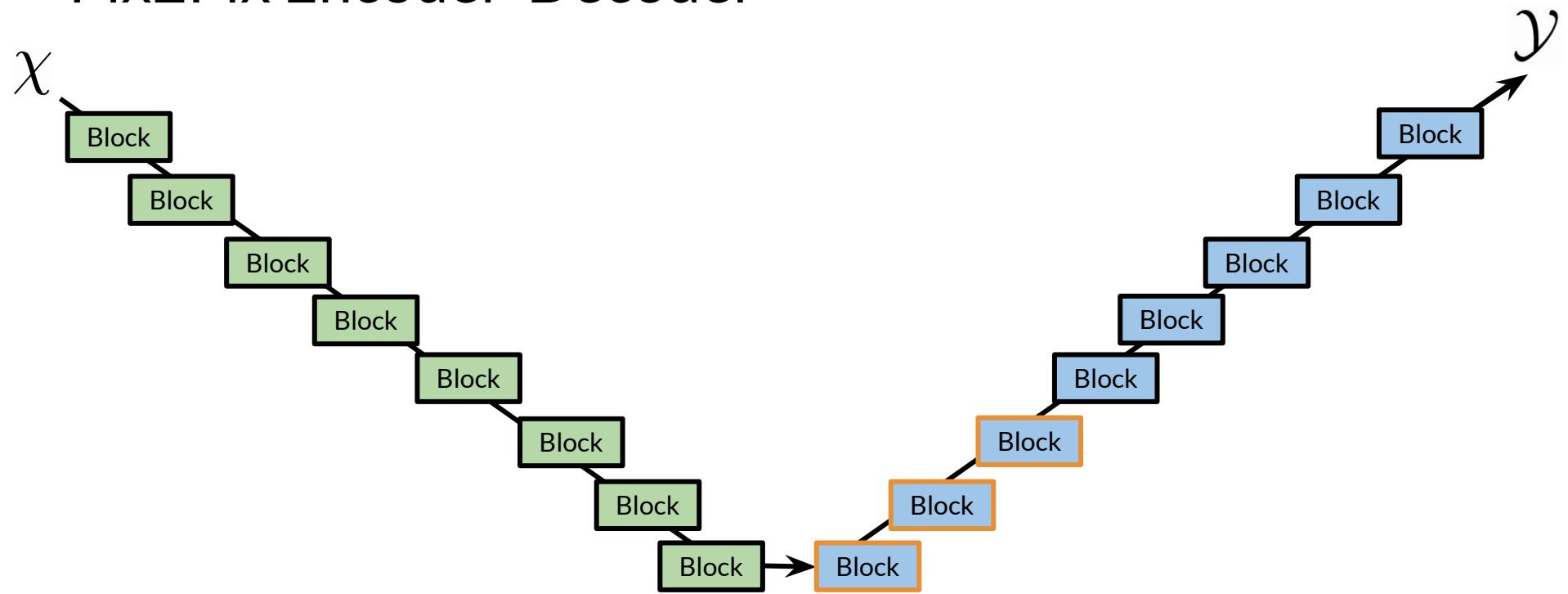
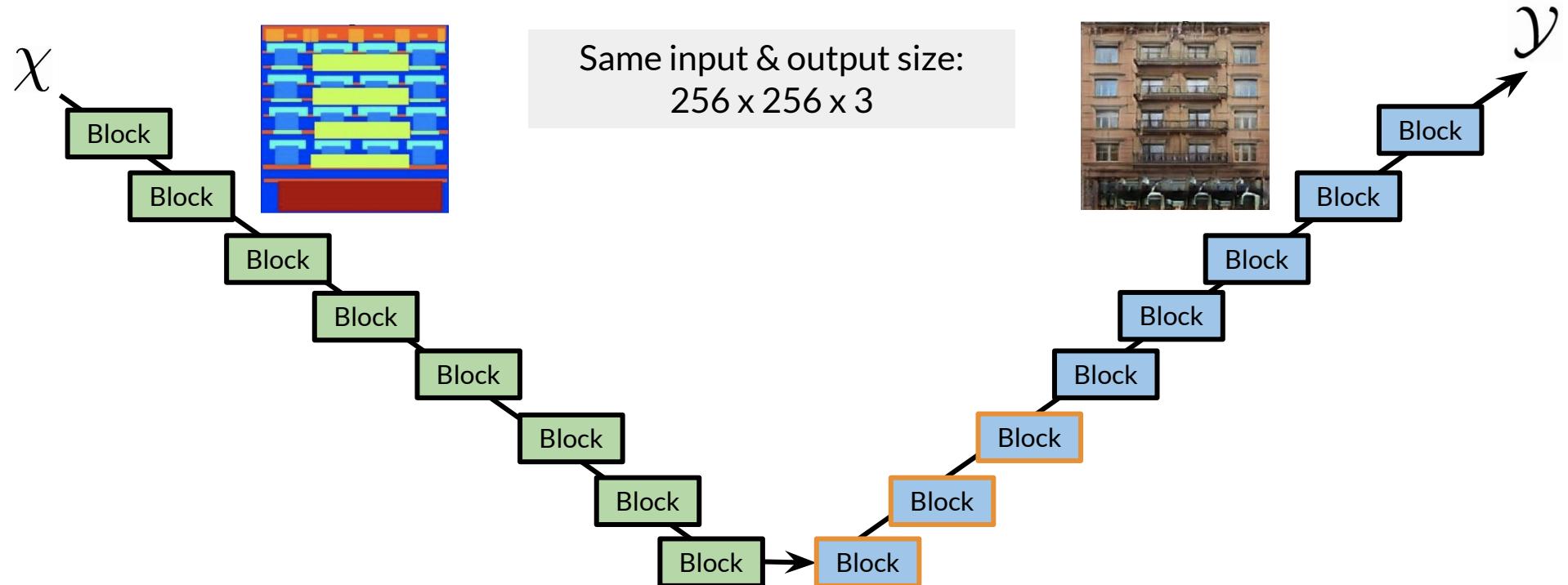


Image available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix Encoder-Decoder

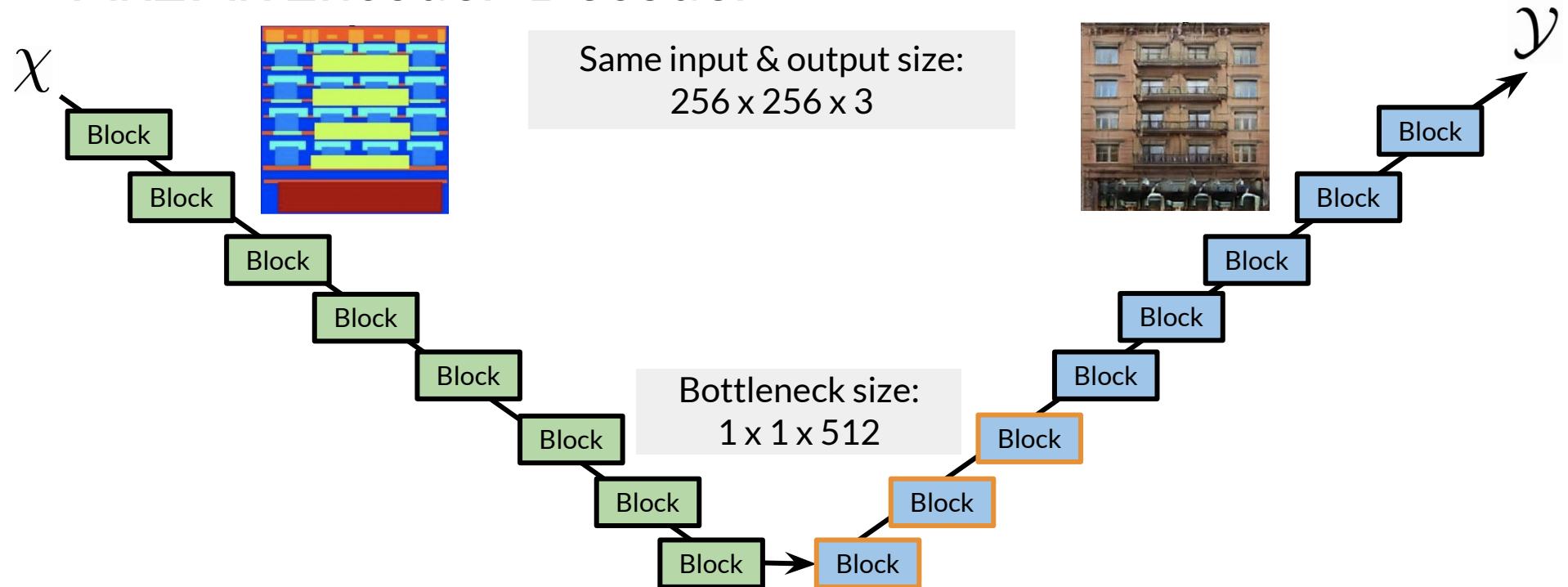


Pix2Pix Encoder-Decoder



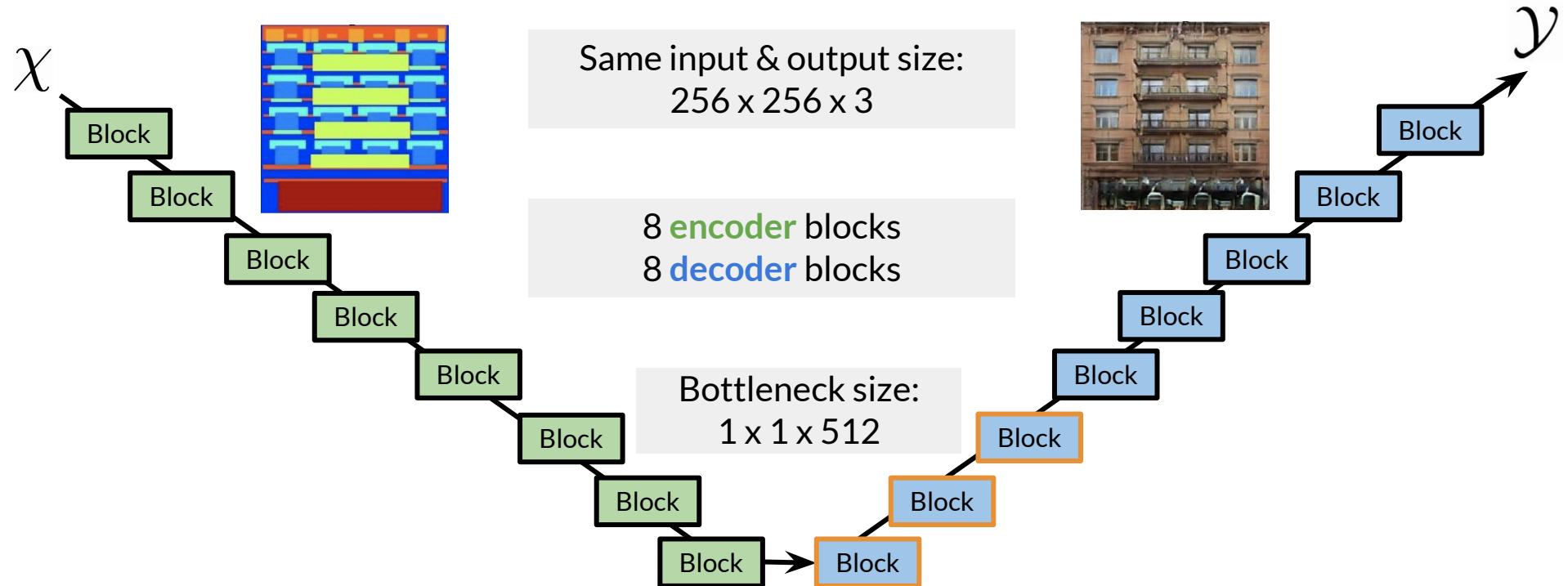
Images available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix Encoder-Decoder



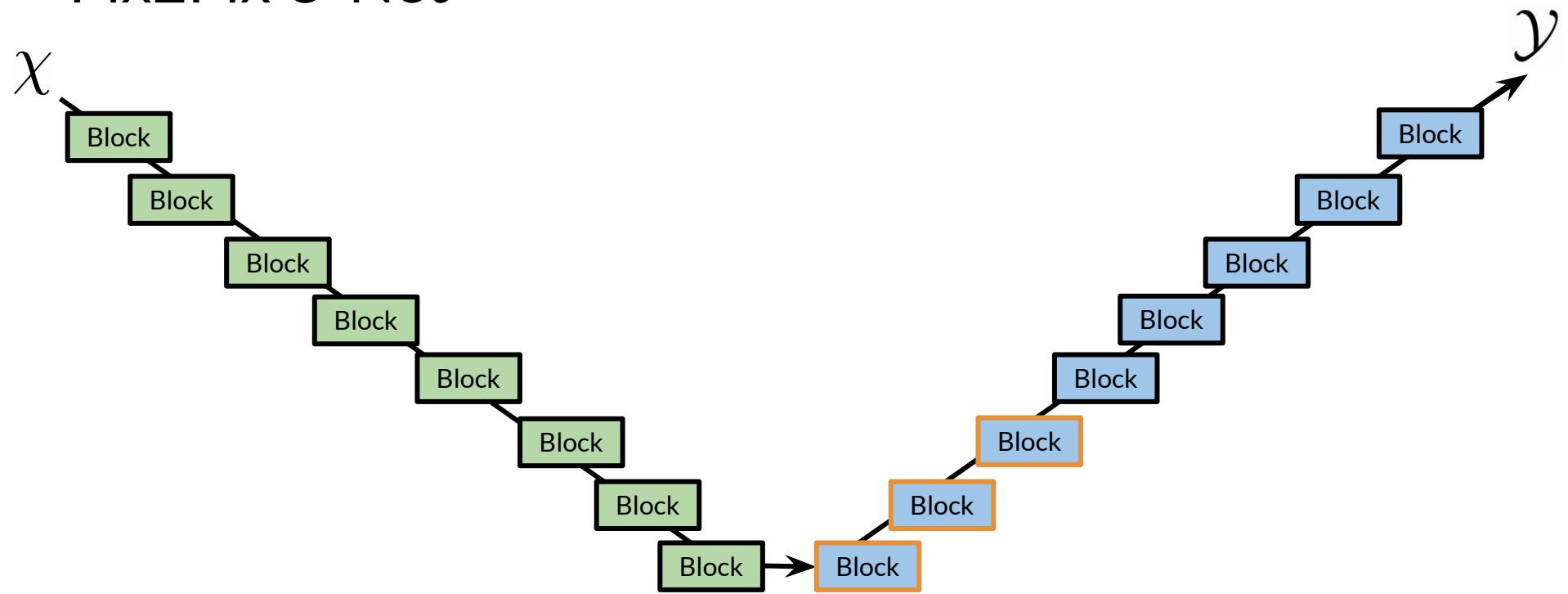
Images available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix Encoder-Decoder

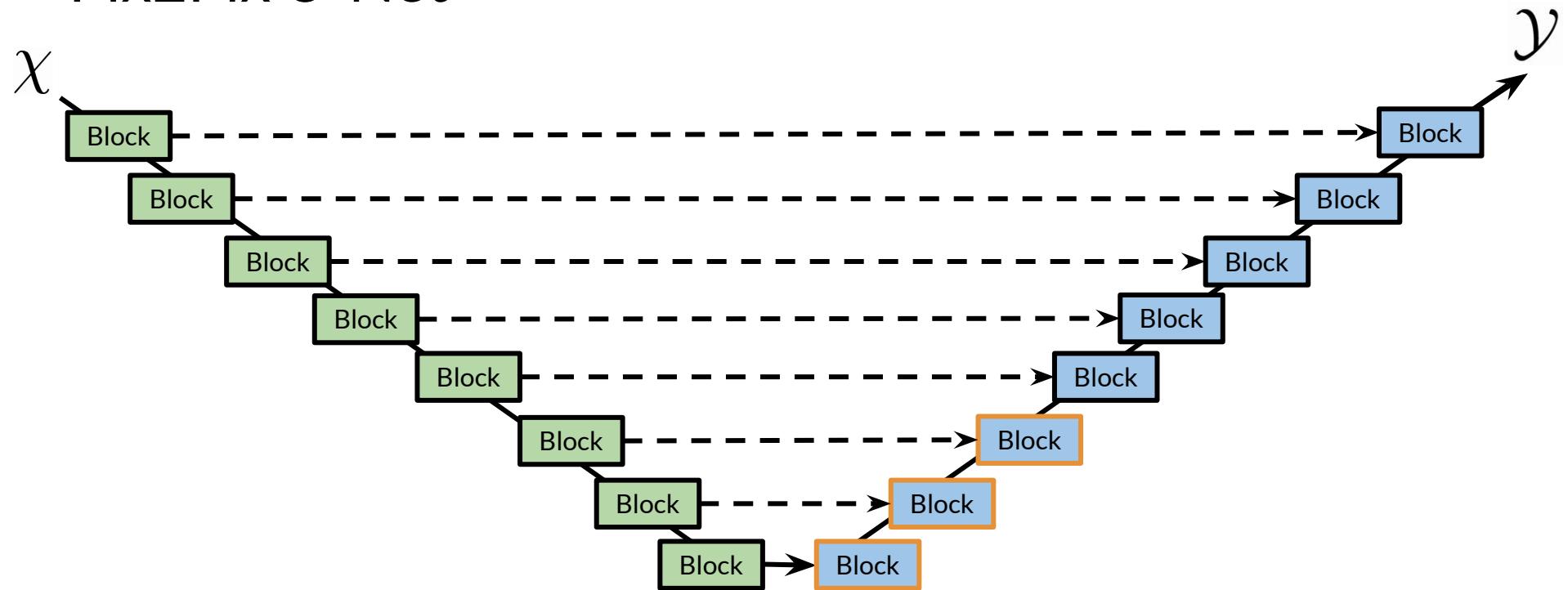


Images available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix U-Net

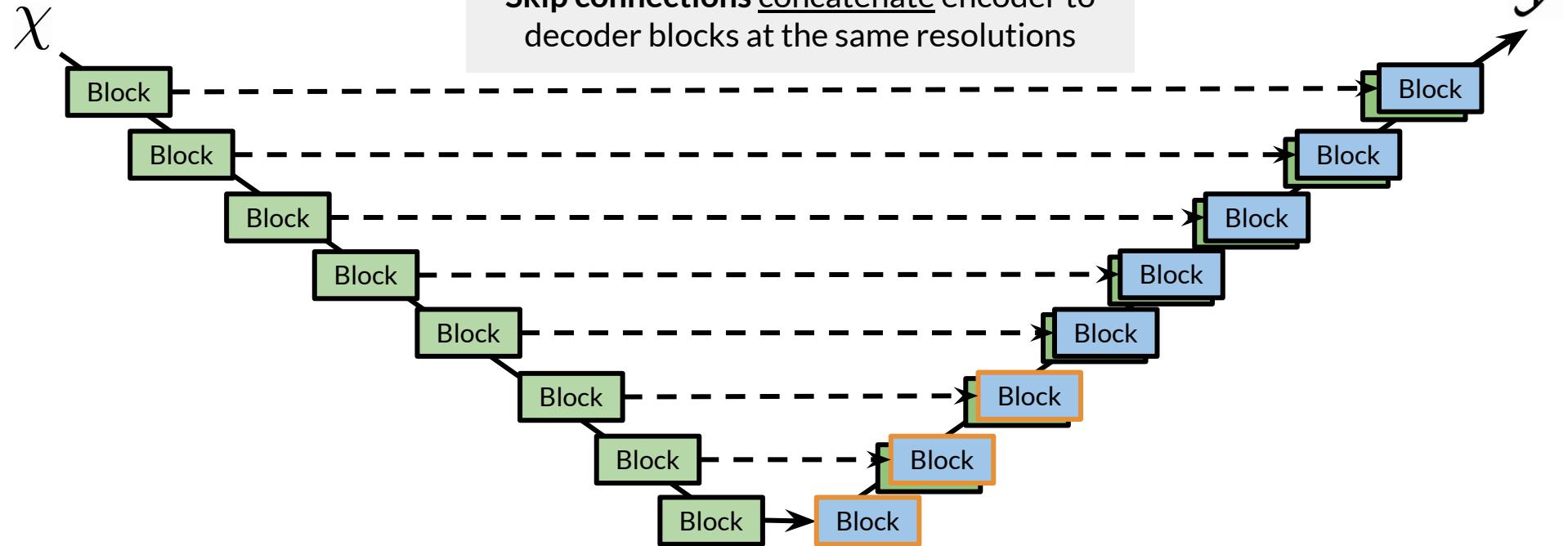


Pix2Pix U-Net



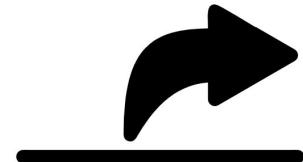
Pix2Pix U-Net

Skip connections concatenate encoder to decoder blocks at the same resolutions



Summary

- Pix2Pix's generator is a U-Net
- U-Net is an encoder-decoder, with same-size inputs and outputs
- U-Net uses skip connections
 - Skip connections help the decoder learn details from the encoder directly
 - Skip connections allow the encoder to learn from more gradients flowing from the decoder



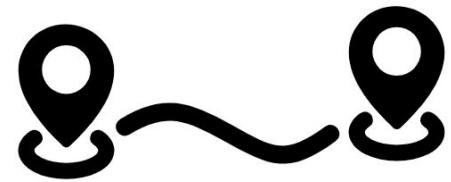


deeplearning.ai

Pix2Pix: Pixel Distance Loss Term

Outline

- Regularization and additional loss term
- Encourage pixel distance between generated and real outputs
- Additional loss term for Pix2Pix generator



Additional Loss Term

$$\min_g \max_c \text{Adversarial Loss} + \lambda * \text{Other loss term}$$

Additional Loss Term

$$\min_g \max_c \text{ Adversarial Loss} + \lambda * \text{Pixel loss term}$$

Pixel Distance Loss Term

$$\sum_{i=1}^n$$



Generated output

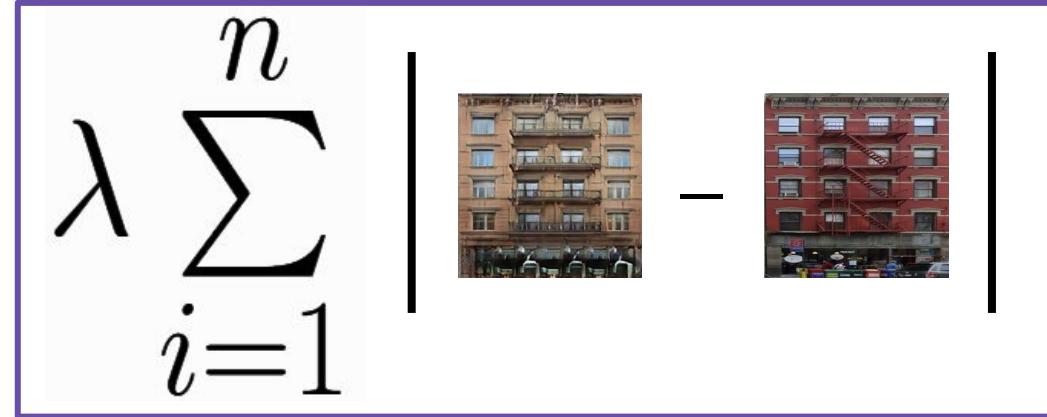


Real output

Available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix Generator Loss

BCE Loss +

$$\lambda \sum_{i=1}^n |$$


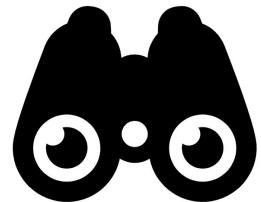
Available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix Generator Loss

$$\text{BCE Loss} + \lambda \sum_{i=1}^n | generated_output - real_output |$$

Summary

- Pix2Pix adds a Pixel Distance Loss term to the generator loss function
- This loss term calculates the difference between the fake and the real target outputs
- Softly encourages the generator with this additional supervision
 - The target output labels are the supervision
 - Generator essentially “sees” these labels



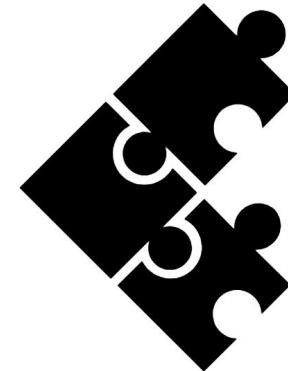


deeplearning.ai

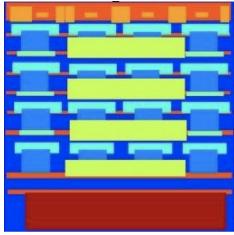
Pix2Pix: Putting It All Together

Outline

- Put the Pix2Pix architecture together!
 - U-Net generator
 - Pixel Distance Loss term
 - PatchGAN discriminator



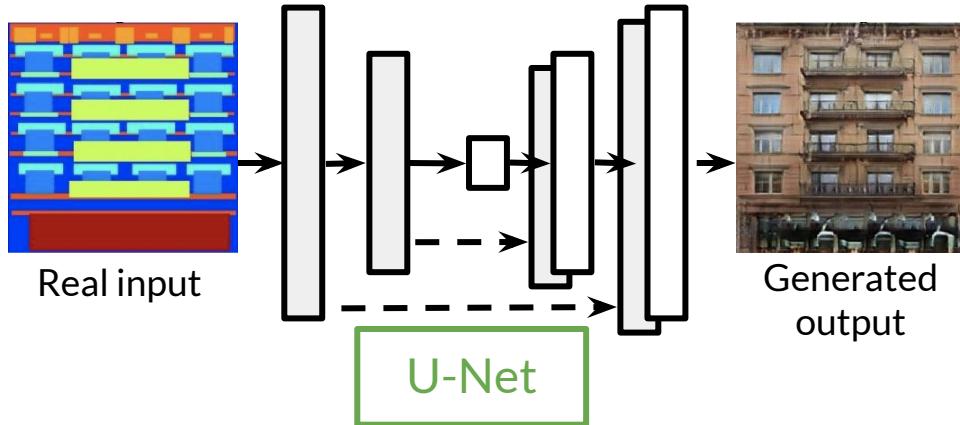
Pix2Pix



Real input

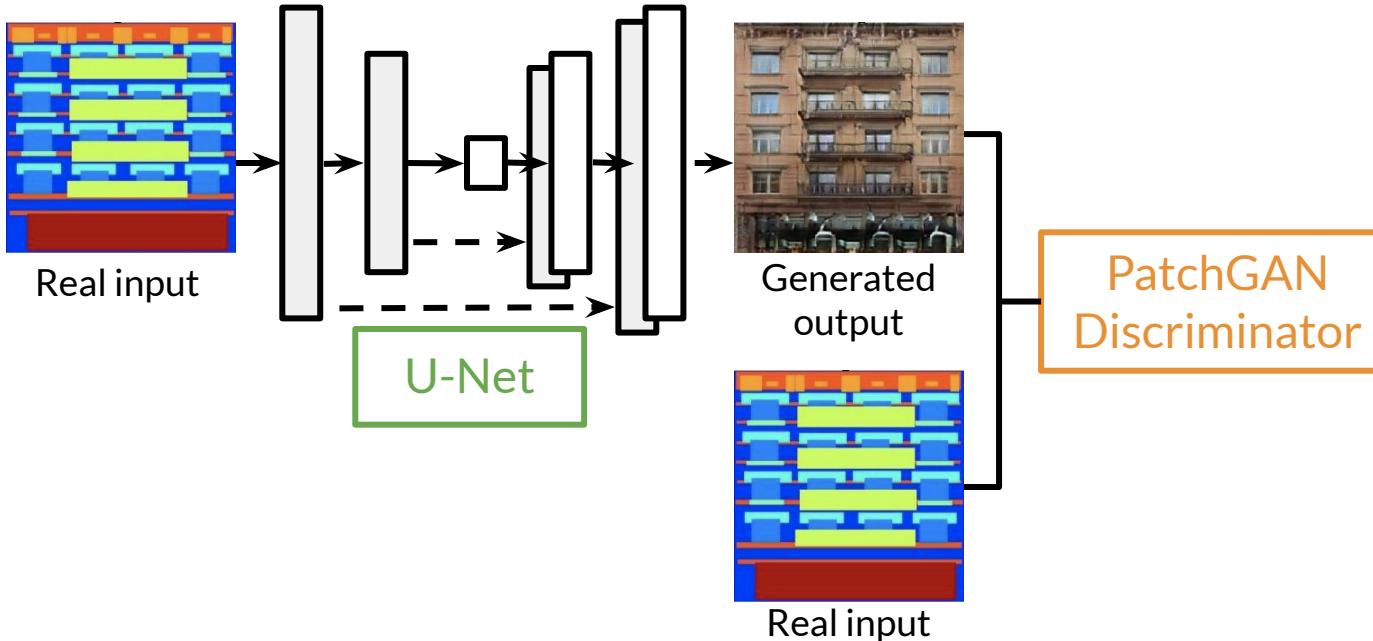
Image available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix



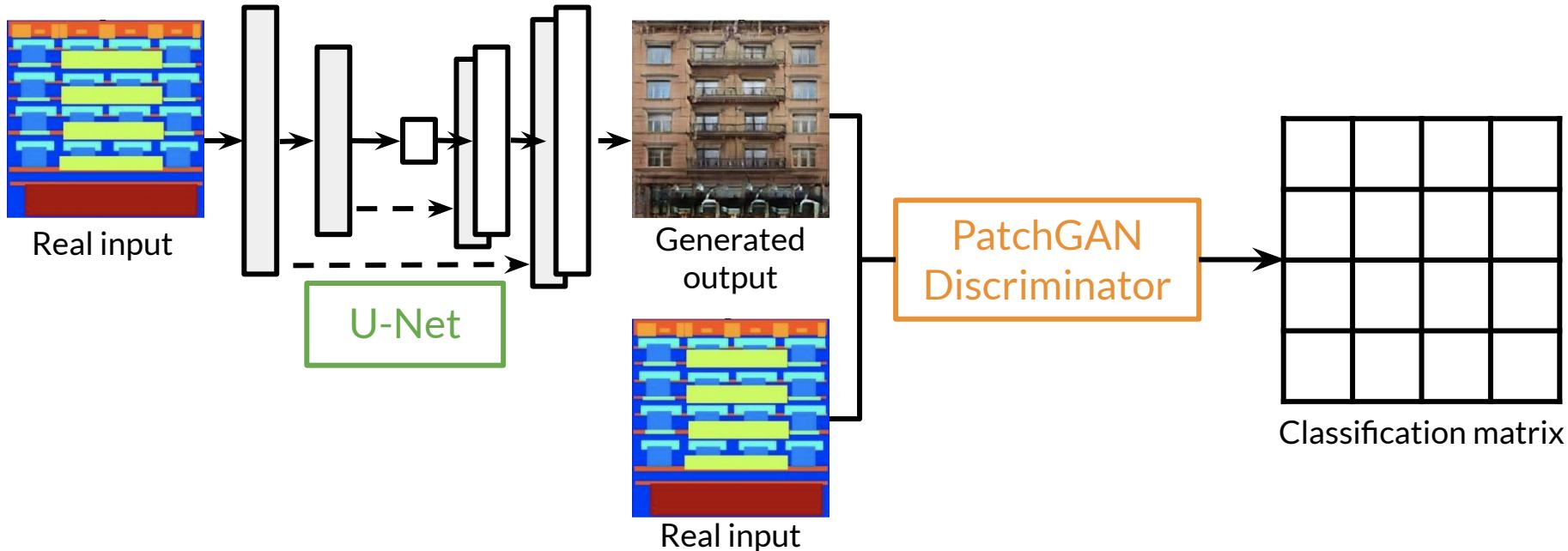
Images available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix



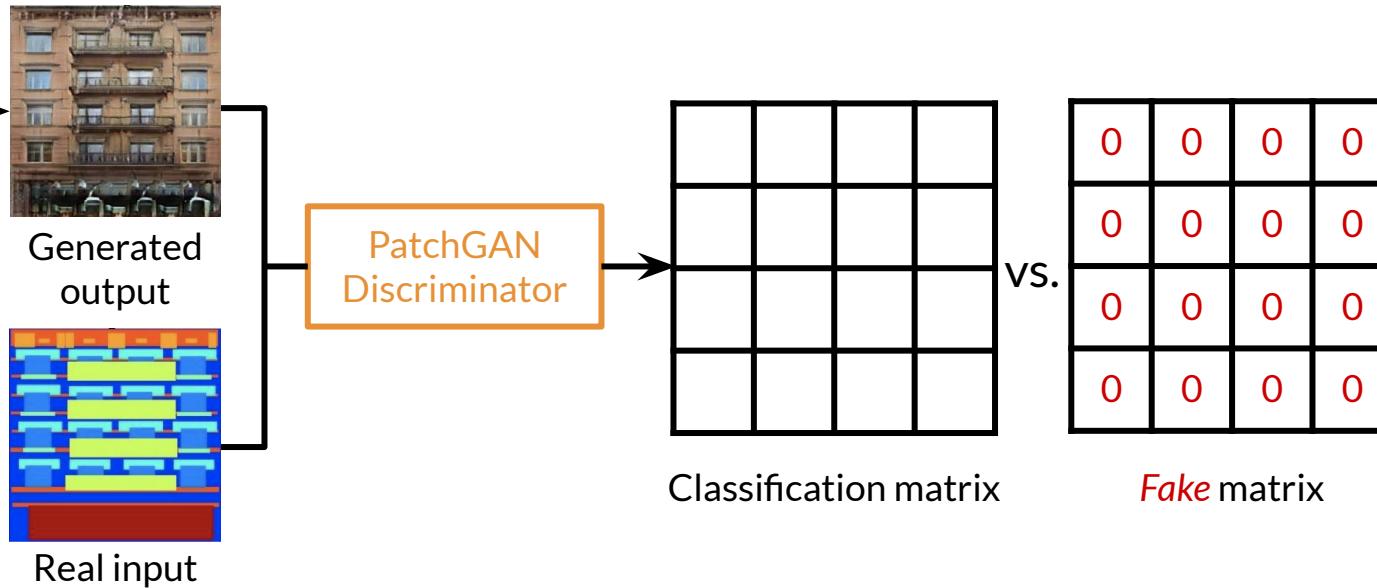
Images available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix



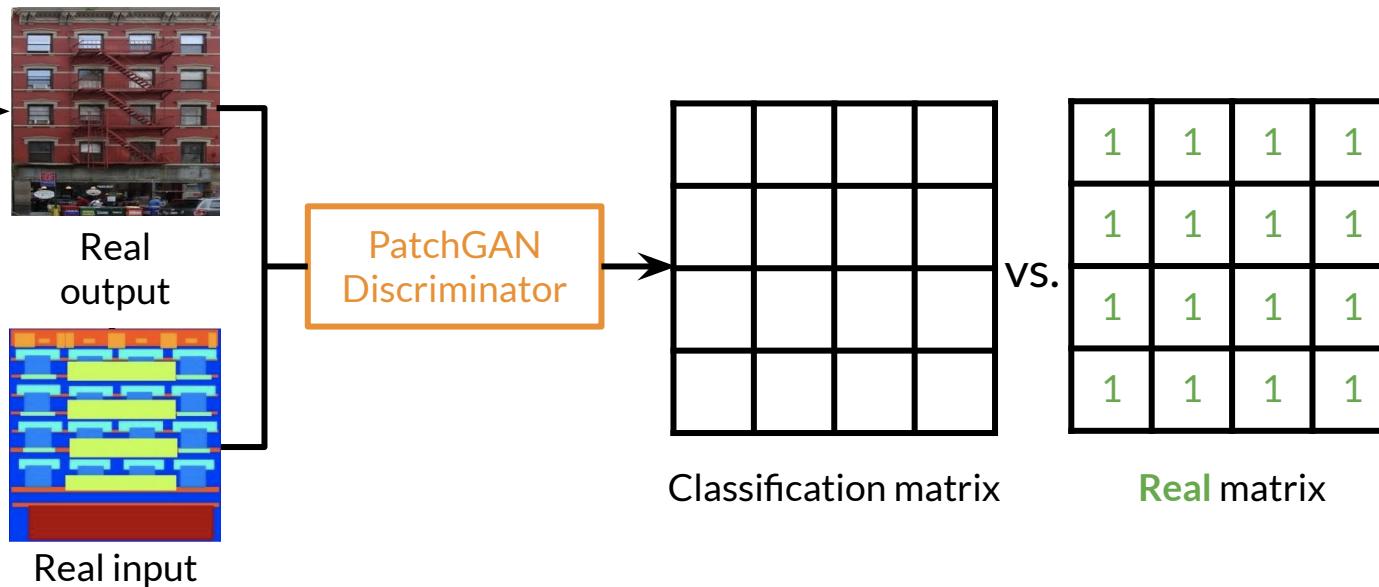
Images available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix: Discriminator Loss



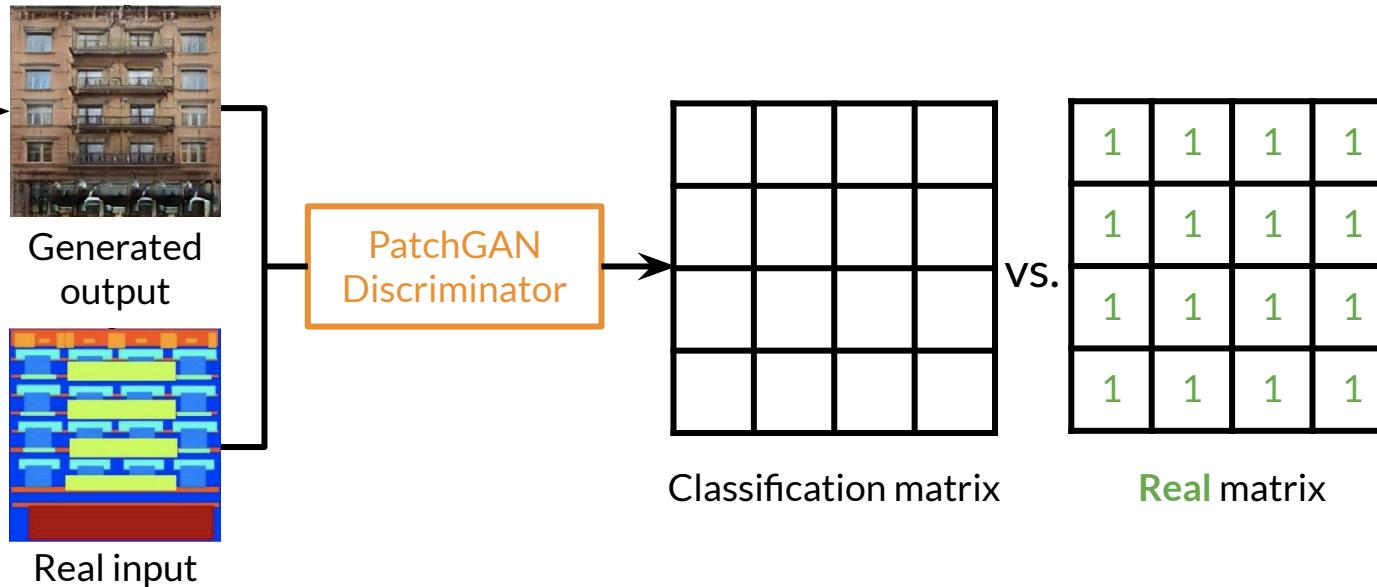
Images available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix: Discriminator Loss



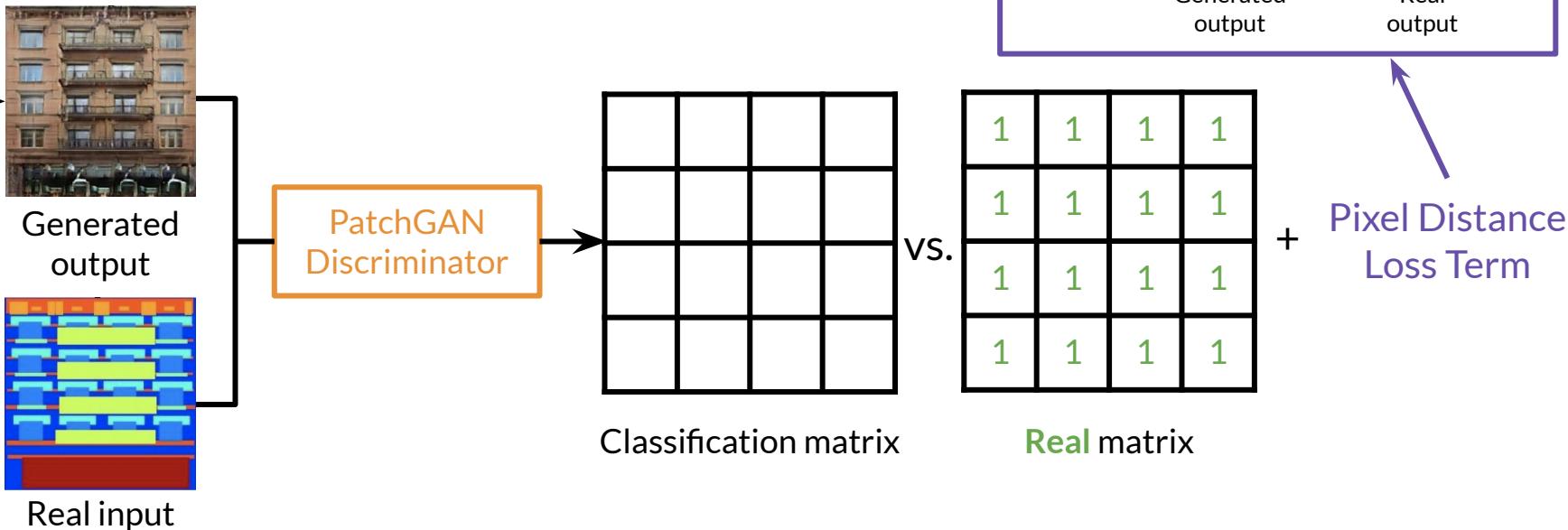
Images available from: <https://arxiv.org/abs/1611.07004>

Pix2Pix: Generator Loss



Images available from: <https://arxiv.org/abs/1611.07004>

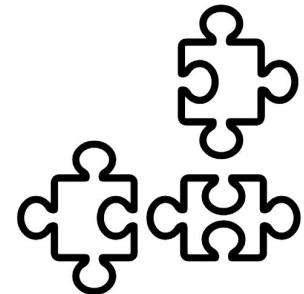
Pix2Pix: Generator Loss



Images available from: <https://arxiv.org/abs/1611.07004>

Summary

- U-Net generator: image → image
- PatchGAN discriminator
 - Inputs input image and paired output (either real target or fake)
 - Outputs classification matrix
- Generator loss has a regularization term



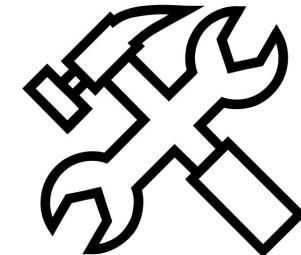


deeplearning.ai

Pix2Pix Advancements

Outline

- Improvements and extensions of Pix2Pix for paired image-to-image translation
 - Higher resolution images
 - Image editing

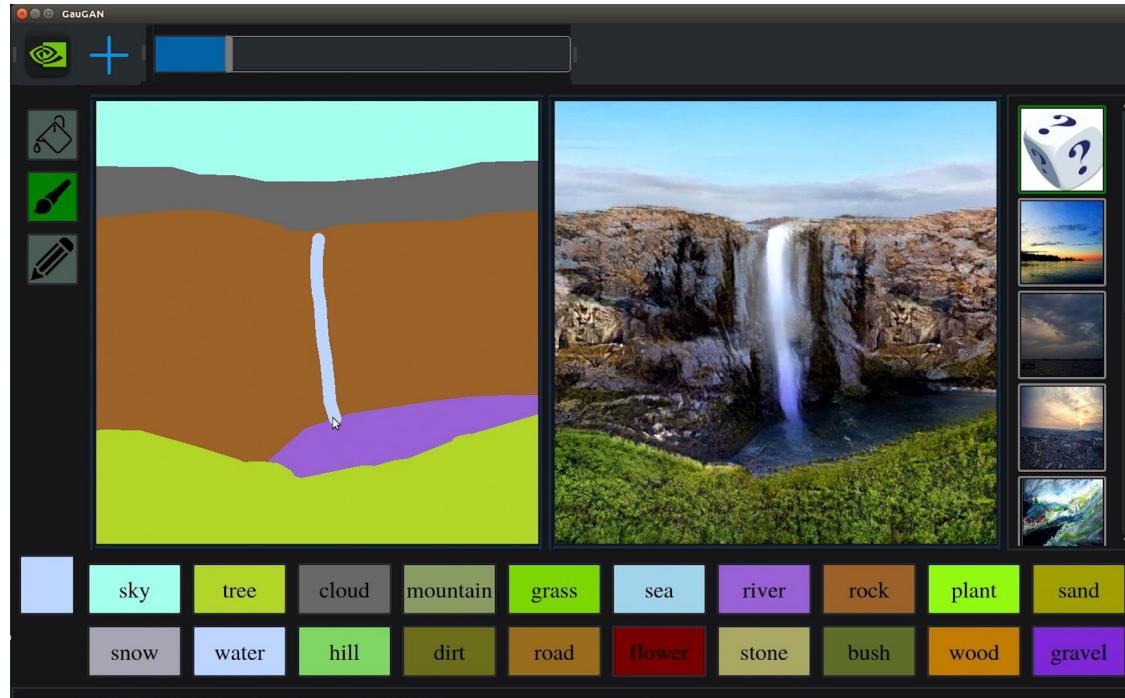


Pix2PixHD



Available from: <https://github.com/NVIDIA/pix2pixHD>

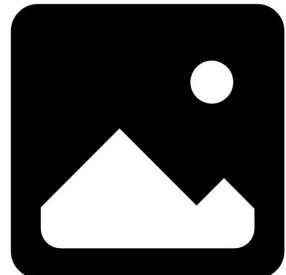
GauGAN



Available from: <https://blogs.nvidia.com/blog/2019/03/18/gaugan-photorealistic-landscapes-nvidia-research/>

Summary

- Pix2PixHD and GauGAN are successors of Pix2Pix
- They are designed for higher resolution images
- They highlight opportunities for image editing using paired image-to-image translation
 - Pix2Pix can do this too, of course!



Copyright Notice

These slides are distributed under the Creative Commons License.

DeepLearning.AI makes these slides available for educational purposes. You may not use or distribute these slides for commercial purposes. You may make copies of these slides and use or distribute them for educational purposes as long as you cite DeepLearning.AI as the source of the slides.

For the rest of the details of the license, see

<https://creativecommons.org/licenses/by-sa/2.0/legalcode>



deeplearning.ai

Unpaired Image-to-Image Translation

Outline

- Paired vs. unpaired image-to-image translation
- Unpaired image-to-image translation
 - Mapping between two piles of image styles
 - Finding commonalities and differences

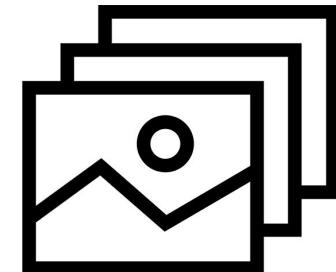


Image-to-Image Translation

Edges to photo



Paired images

Available from: <https://arxiv.org/abs/1611.07004>

Image-to-Image Translation

Edges to photo



Paired images

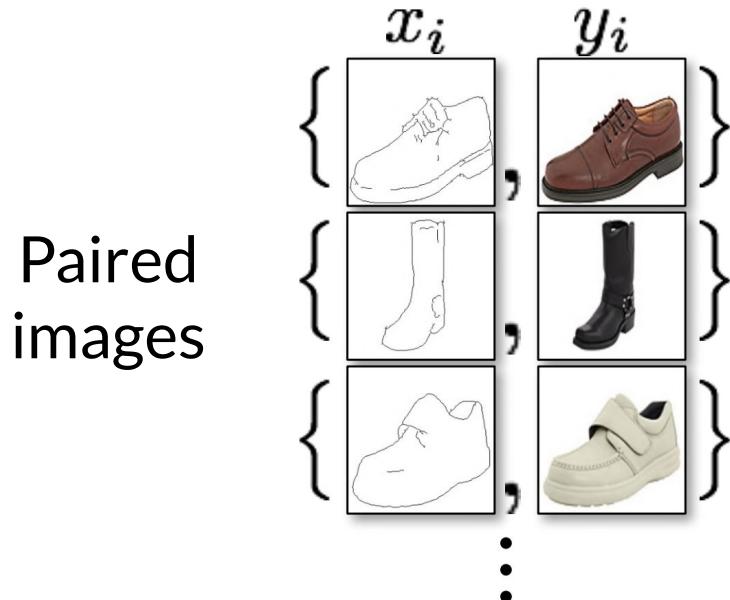
Monet to photo



Unpaired images

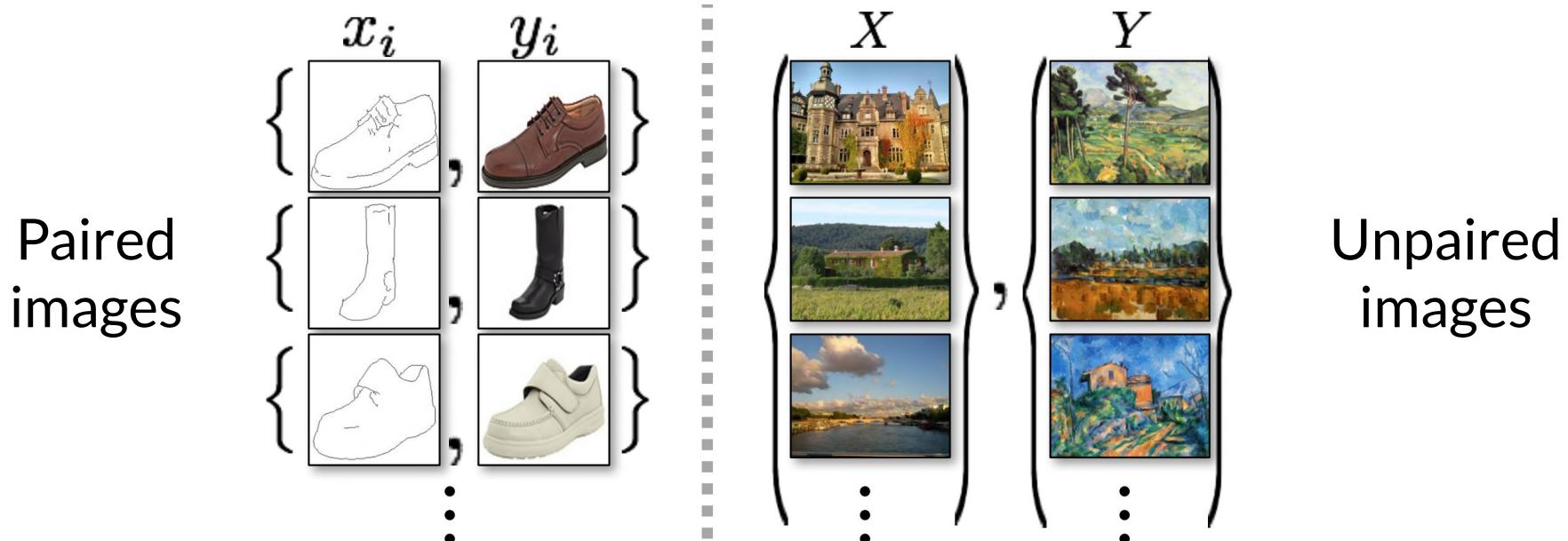
Available from: <https://arxiv.org/abs/1611.07004>

Image-to-Image Translation



Available from: <https://arxiv.org/abs/1703.10593>

Image-to-Image Translation



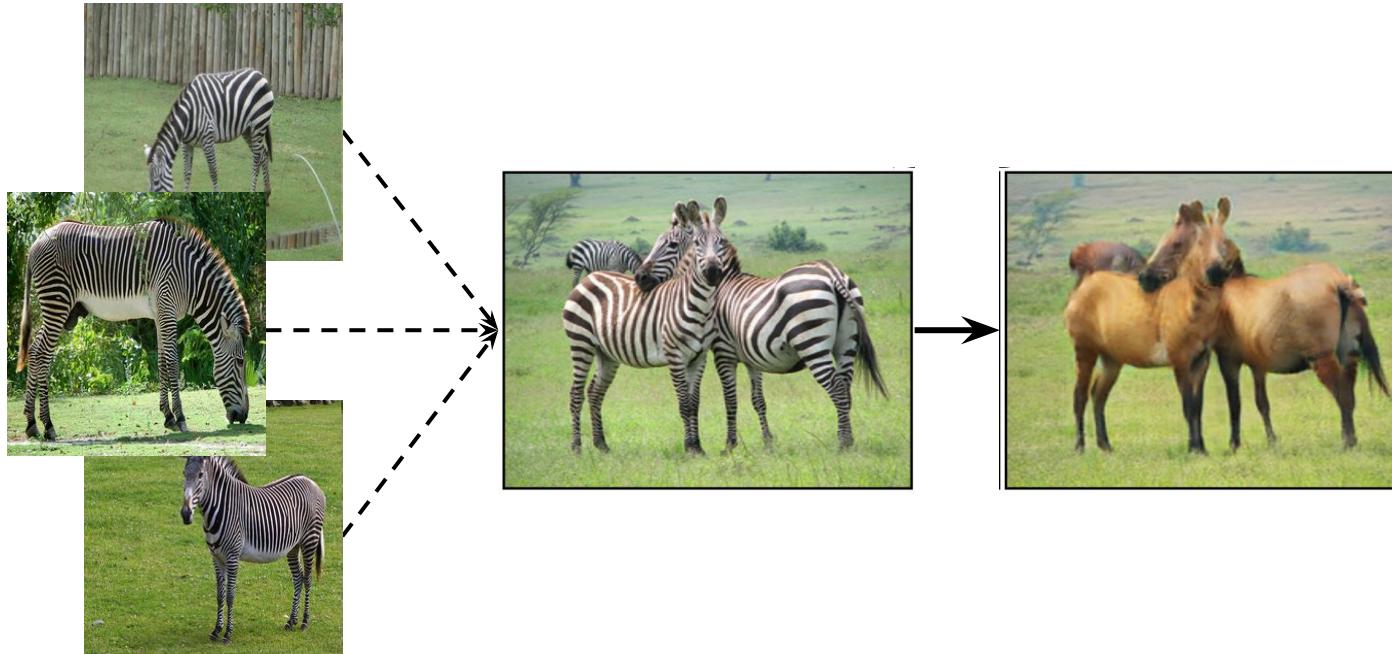
Available from: <https://arxiv.org/abs/1703.10593>

Unpaired Image-to-Image Translation



Available from: <https://arxiv.org/abs/1703.10593>

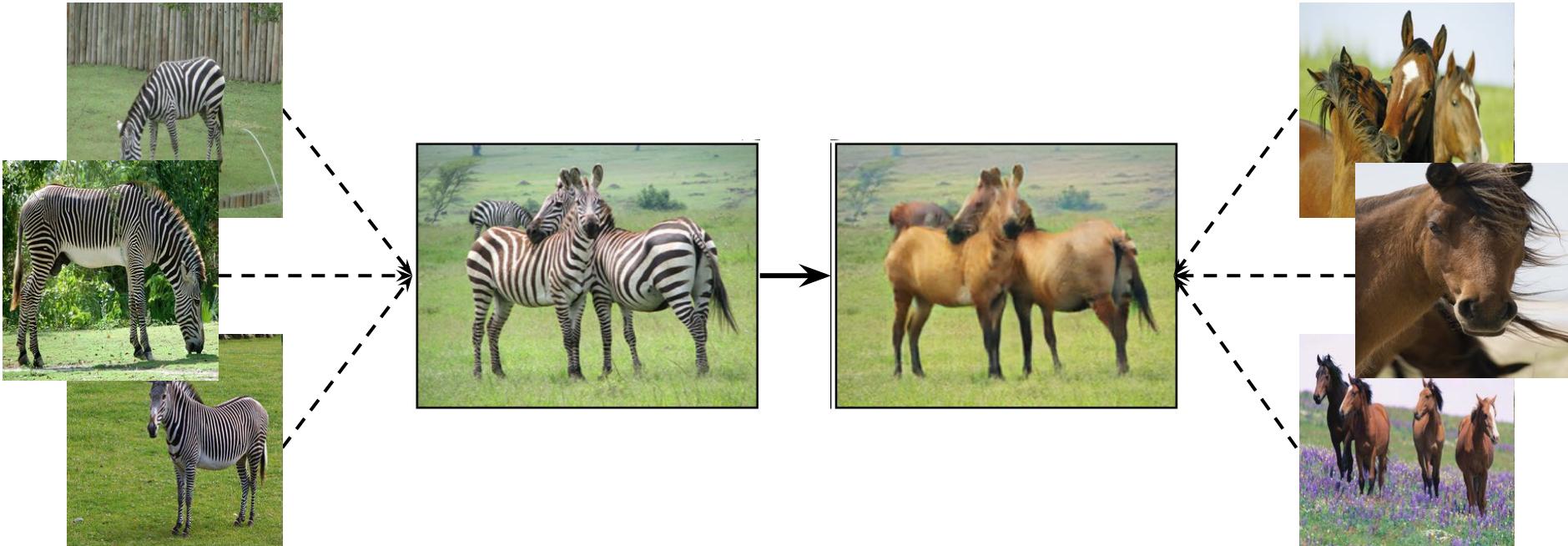
Mapping Between Two Piles



(Center) Images available from: <https://arxiv.org/abs/1703.10593>

(Side) Images available from: <https://github.com/togheppi/CycleGAN>

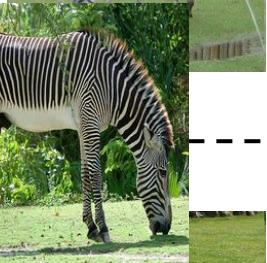
Mapping Between Two Piles



(Center) Images available from: <https://arxiv.org/abs/1703.10593>

(Sides) Images available from: <https://github.com/togheppi/CycleGAN>

Mapping Between Two Piles

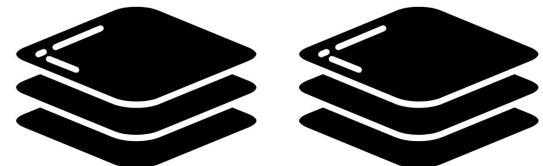


Content = common elements
Style = unique elements

(Center) Images available from: <https://arxiv.org/abs/1703.10593>
(Sides) Images available from: <https://github.com/togheppi/CycleGAN>

Summary

- Unpaired image-to-image translation:
 - Learns a mapping between two piles of images
 - Examines common elements of the two piles (content) and unique elements of each pile (style)
- Unlike paired image-to-image translation,
this method is unsupervised



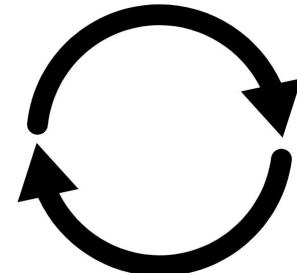


deeplearning.ai

CycleGAN Overview

Outline

- Overview of CycleGAN
 - The “Cycle” in CycleGAN
 - Two GANs!



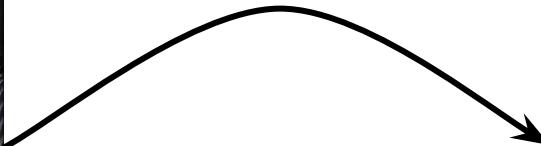
Cycle Consistency



Images available from: <https://github.com/togheppi/CycleGAN>

Cycle Consistency

Real



Fake

Images available from: <https://github.com/togheppi/CycleGAN>

Cycle Consistency

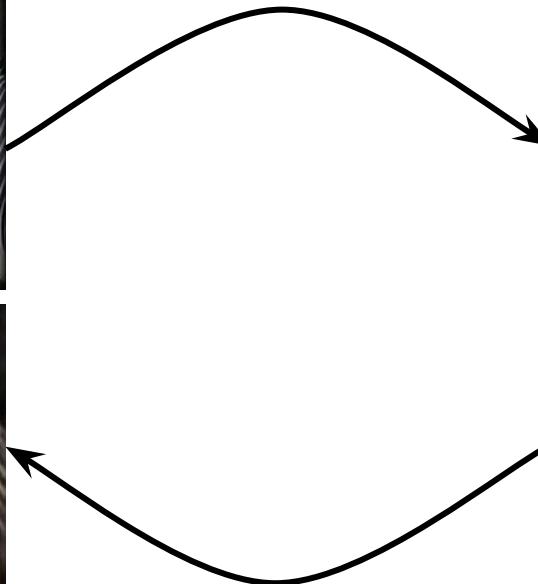
Real



Fake



Fake



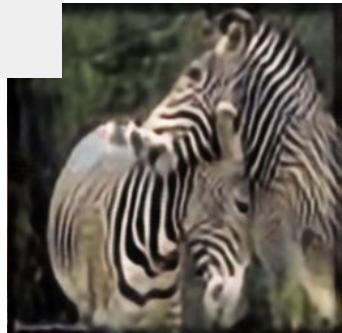
Images available from: <https://github.com/togheppi/CycleGAN>

Cycle Consistency

Real

Should be the
same

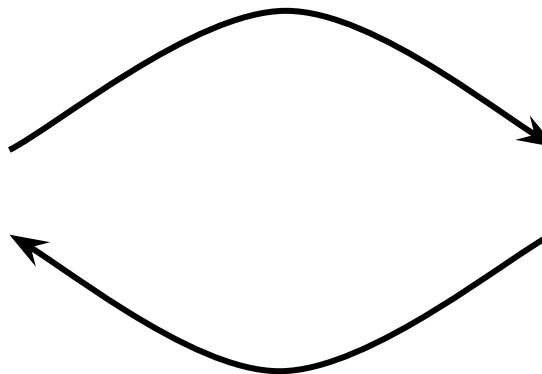
Fake



Fake

Images available from: <https://github.com/togheppi/CycleGAN>

Cycle Consistency

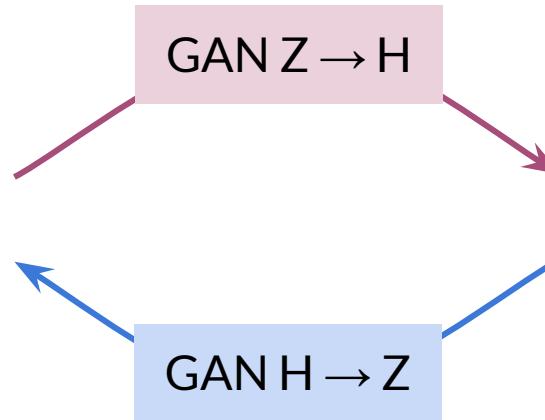


Cycle consistency



Images available from: <https://github.com/togheppi/CycleGAN>

Two GANs



Images available from: <https://github.com/togheppi/CycleGAN>

Two GANs

Real



GAN Z → H

Fake



GAN H → Z

Fake

Images available from: <https://github.com/togheppi/CycleGAN>

Two GANs

Real



GAN $Z \rightarrow H$

Fake



GAN $H \rightarrow Z$

Fake

Real



GAN $H \rightarrow Z$

Fake



GAN $Z \rightarrow H$

Fake

Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN

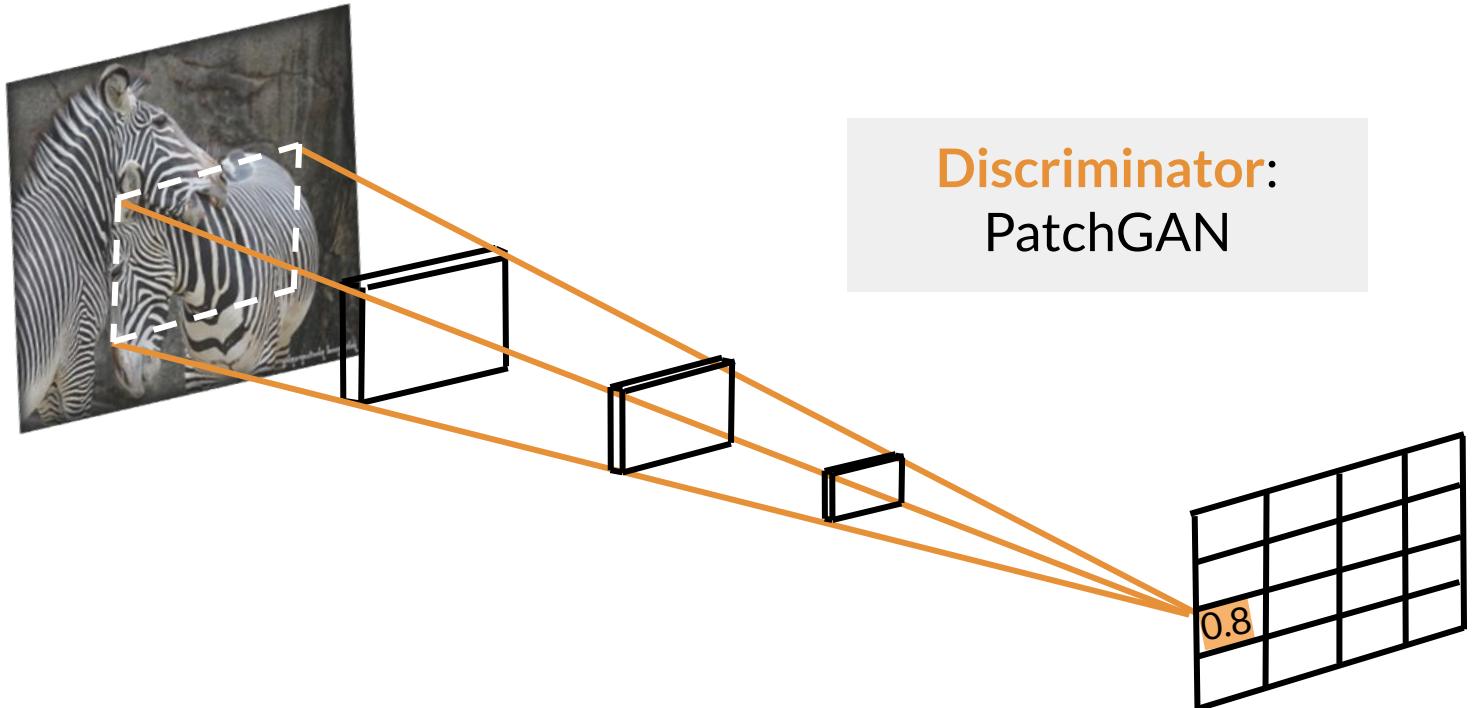
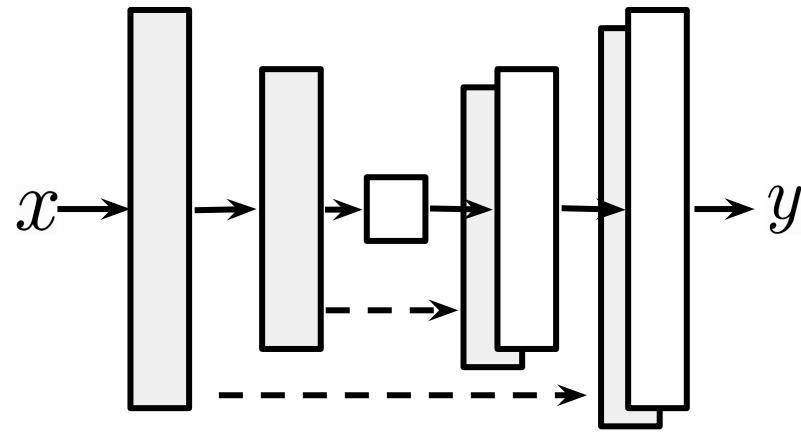


Image available from: <https://github.com/togheppi/CycleGAN>

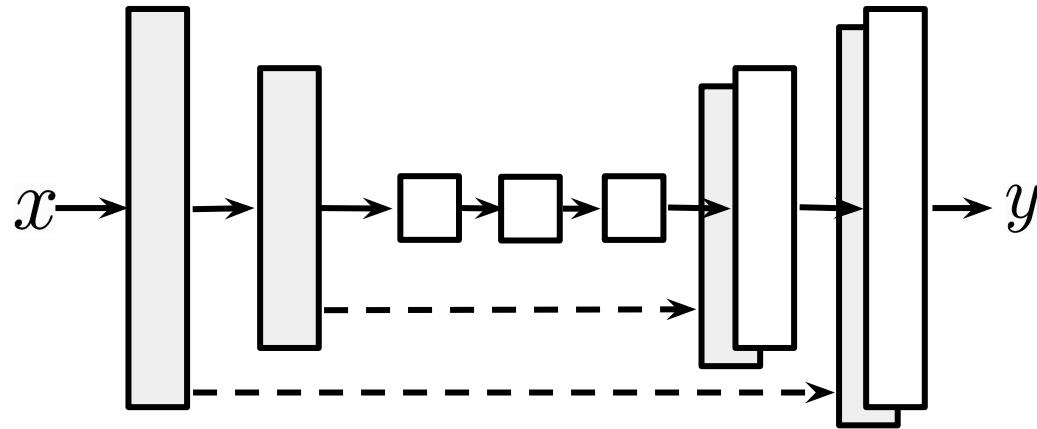
CycleGAN



Generator ≈
U-Net

Available from: <https://arxiv.org/abs/1611.07004>

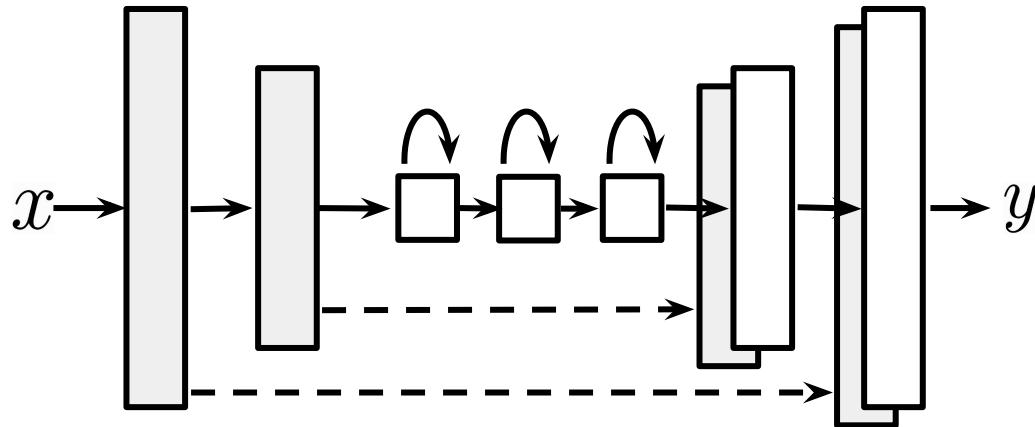
CycleGAN



Generator ≈
U-Net + DCGAN
generator

Available from: <https://arxiv.org/abs/1611.07004>

CycleGAN



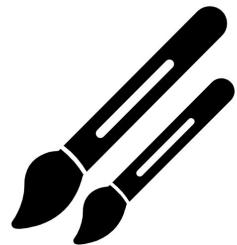
Additional skip connections

Generator ≈
U-Net + DCGAN
generator

Available from: <https://arxiv.org/abs/1611.07004>

Summary

- CycleGAN uses two GANs for unpaired image-to-image translation
- The discriminators are PatchGAN's
- The generators are similar to a U-Net and DCGAN generator with additional skip connections



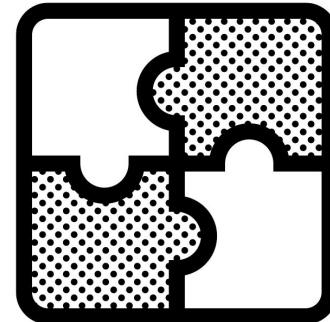


deeplearning.ai

CycleGAN: Two GANs

Outline

- Two GANs, four components
 - Two generators
 - Two discriminators

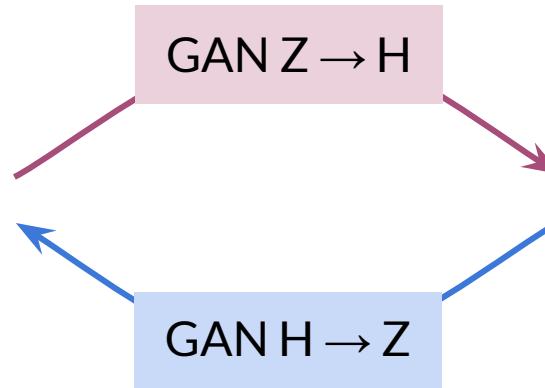


CycleGAN Components



Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN Components

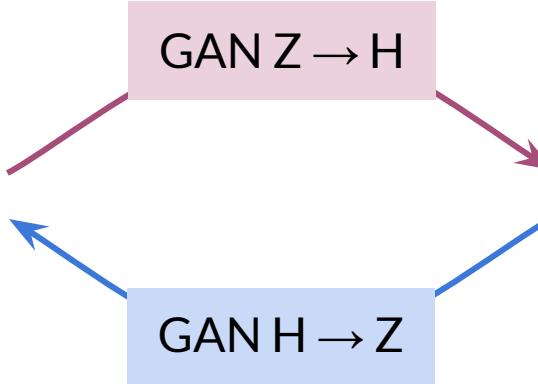


Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN Components

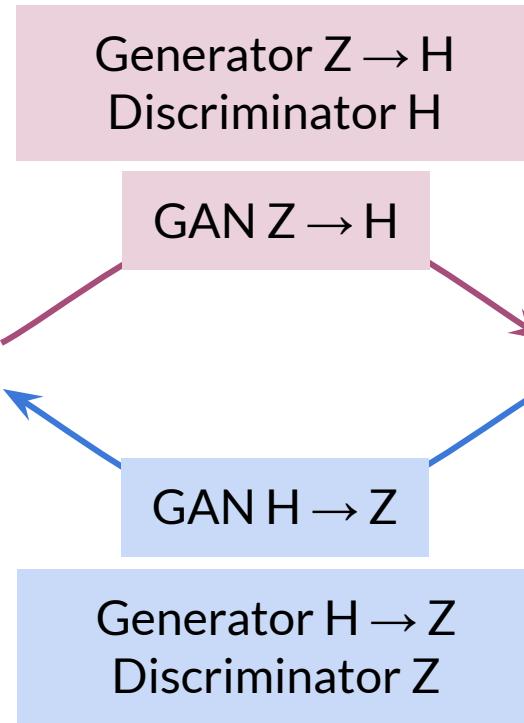
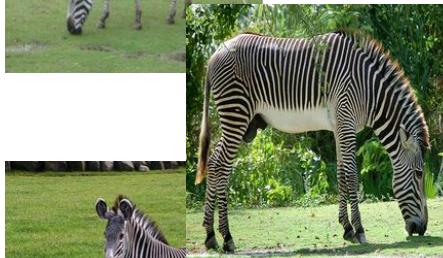


Generator $Z \rightarrow H$
Discriminator H



Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN Components



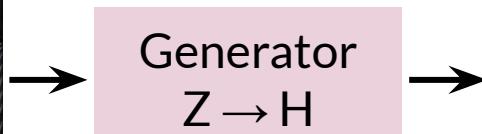
Images available from: <https://github.com/togheppi/CycleGAN>

GAN $Z \rightarrow H$

Real



Fake



Images available from: <https://github.com/togheppi/CycleGAN>

GAN $Z \rightarrow H$

Real



Generator
 $Z \rightarrow H$

Fake



Discriminator
 H

Real



Discriminator
 H

Images available from: <https://github.com/togheppi/CycleGAN>

GAN $Z \rightarrow H$

Real



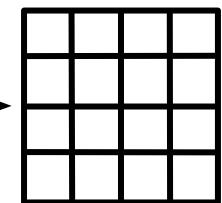
Generator
 $Z \rightarrow H$

Fake



Discriminator
 H

Classification matrix

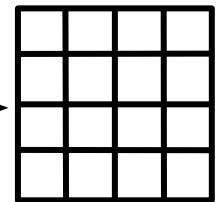


Real



Discriminator
 H

Classification matrix



Images available from: <https://github.com/togheppi/CycleGAN>

GAN $H \rightarrow Z$

Real



Generator
 $H \rightarrow Z$

Fake



Images available from: <https://github.com/togheppi/CycleGAN>

GAN $H \rightarrow Z$

Real



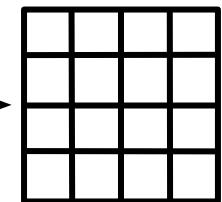
Generator
 $H \rightarrow Z$

Fake

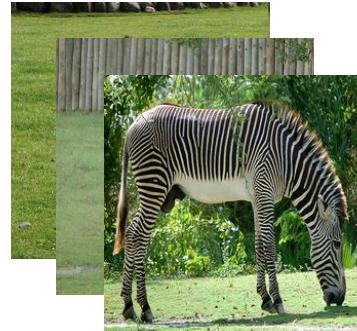


Discriminator
 Z

Classification matrix

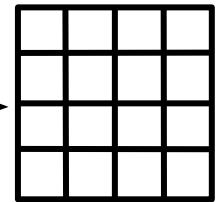


Real



Discriminator
 Z

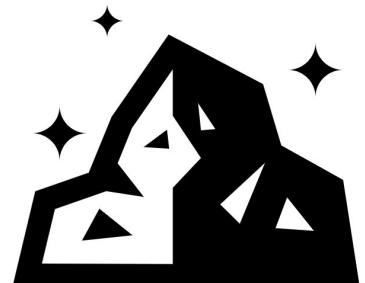
Classification matrix



Images available from: <https://github.com/togheppi/CycleGAN>

Summary

- CycleGAN has four components:
 - Two generators
 - Two discriminators
- The inputs to the generators and discriminators are similar to Pix2Pix, except:
 - There are no real target outputs
 - Each discriminator is in charge of one pile of images



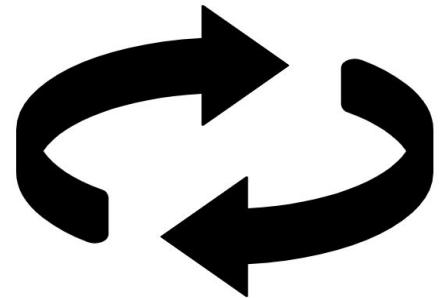


deeplearning.ai

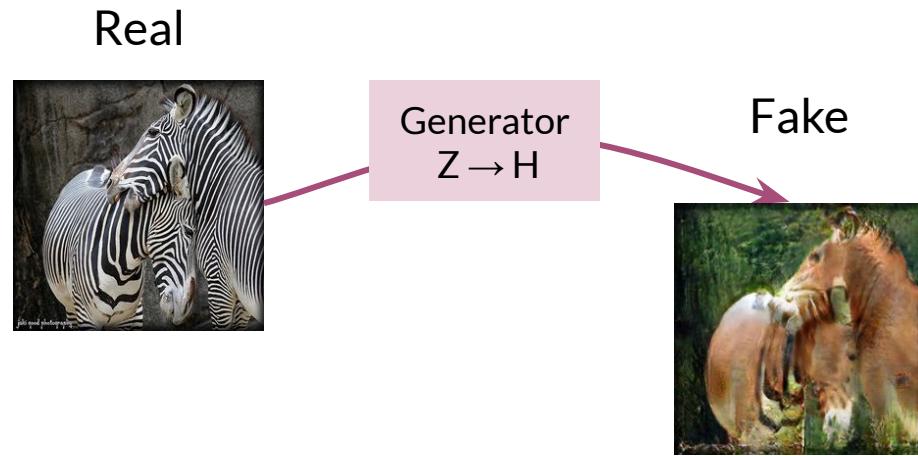
CycleGAN: Cycle Consistency

Outline

- Encouraging cycle consistency
 - Cycle Consistency Loss term
- Loss with cycle consistency for each of two GANs
- How cycle consistency helps



Cycle Consistency Loss



Images available from: <https://github.com/togheppi/CycleGAN>

Cycle Consistency Loss

Real



Generator
 $Z \rightarrow H$

Fake

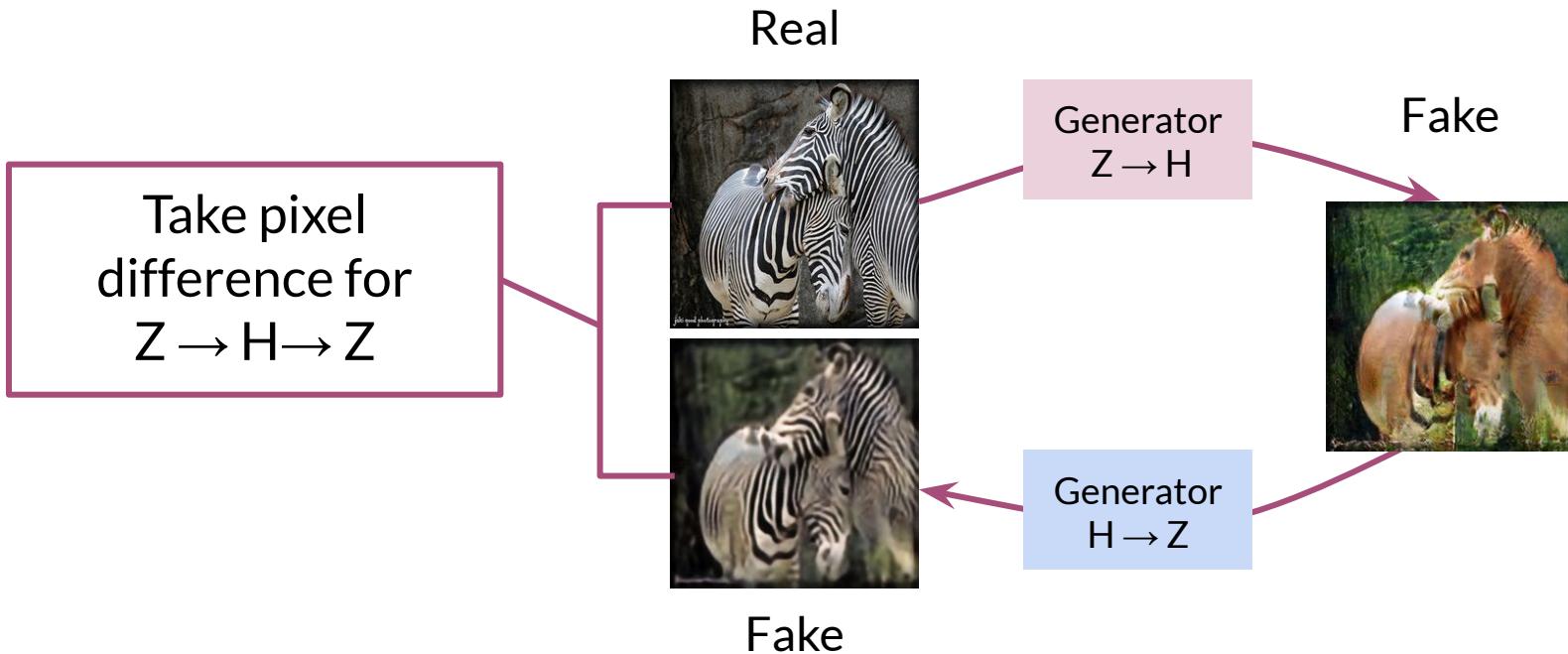


Generator
 $H \rightarrow Z$

Fake

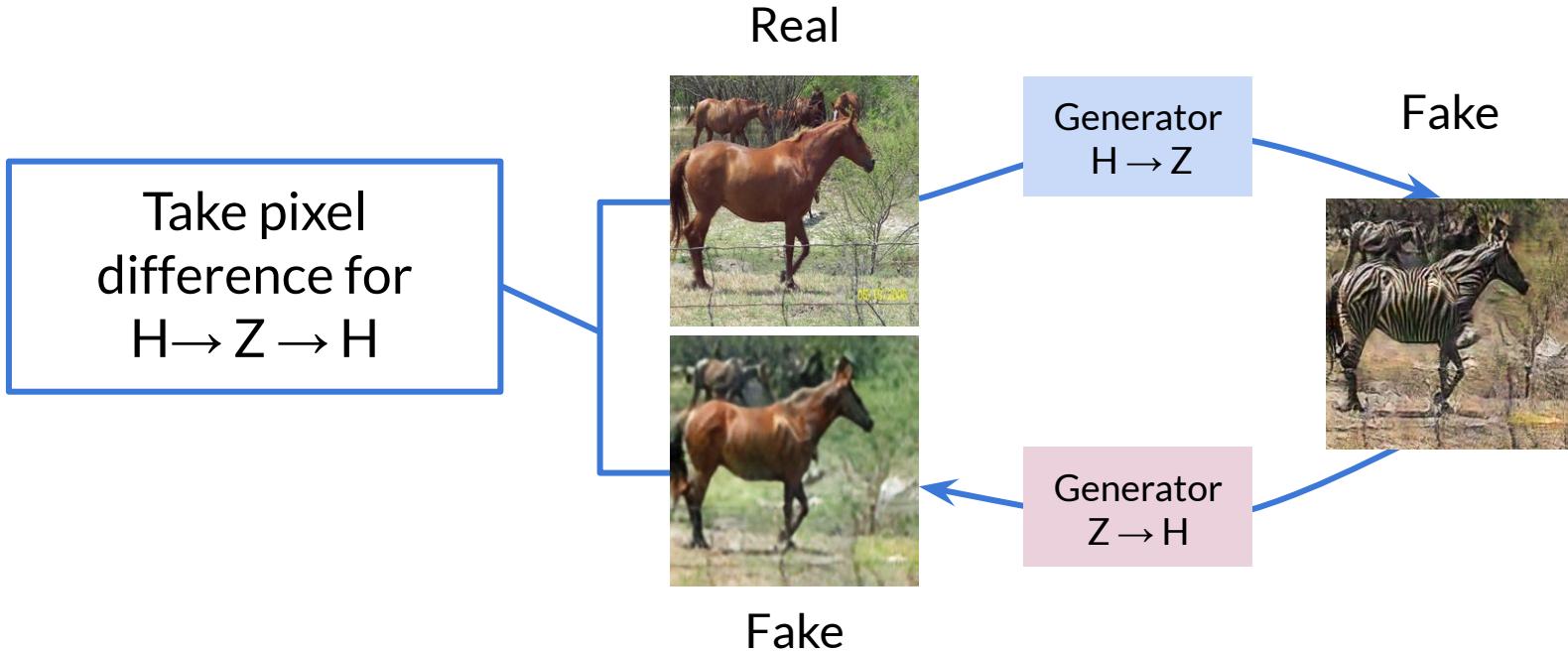
Images available from: <https://github.com/togheppi/CycleGAN>

Cycle Consistency Loss



Images available from: <https://github.com/togheppi/CycleGAN>

Cycle Consistency Loss



Images available from: <https://github.com/togheppi/CycleGAN>

Cycle Consistency Loss

$$\sum_i |$$



—



+

$$\sum_i |$$



—



$Z \rightarrow H \rightarrow Z$

$H \rightarrow Z \rightarrow H$

Cycle Consistency Loss is the
sum of both directions

Cycle Consistency Loss

Adversarial Loss +

$$\sum_i | \begin{array}{c} \text{zebra} \\ \text{---} \\ \text{horse} \end{array} - \begin{array}{c} \text{zebra} \\ \text{---} \\ \text{horse} \end{array} | + \sum_i | \begin{array}{c} \text{horse} \\ \text{---} \\ \text{zebra} \end{array} - \begin{array}{c} \text{horse} \\ \text{---} \\ \text{zebra} \end{array} |$$

$Z \rightarrow H \rightarrow Z$ $H \rightarrow Z \rightarrow H$

Cycle Consistency Loss

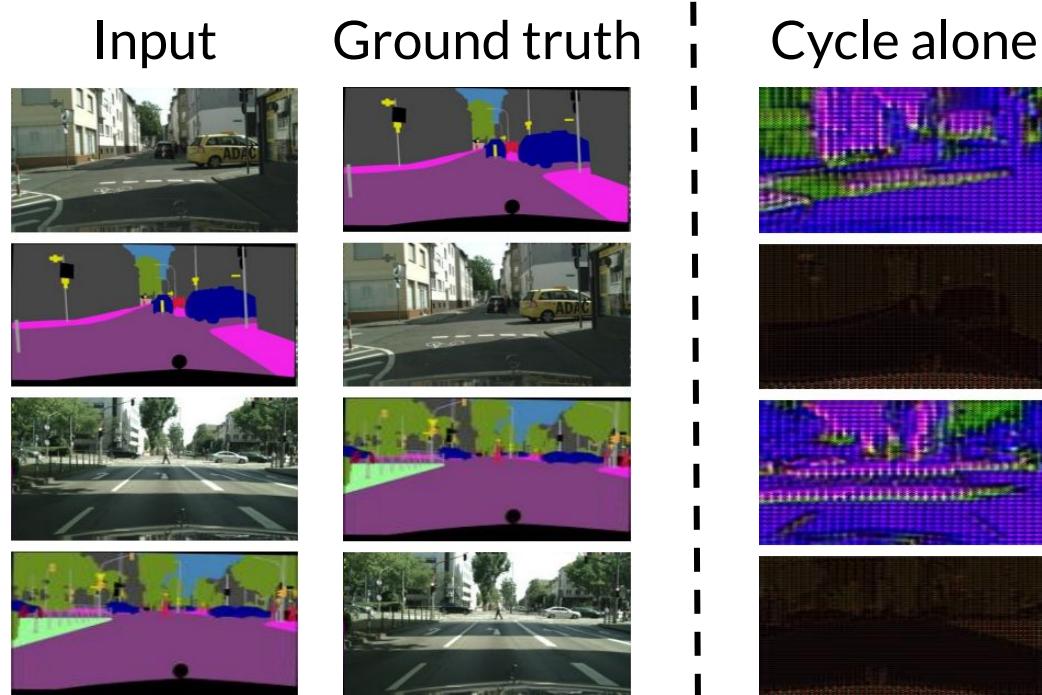
Adversarial Loss +

Cycle Consistency Loss

Cycle Consistency Loss

Adversarial Loss + λ * Cycle Consistency Loss

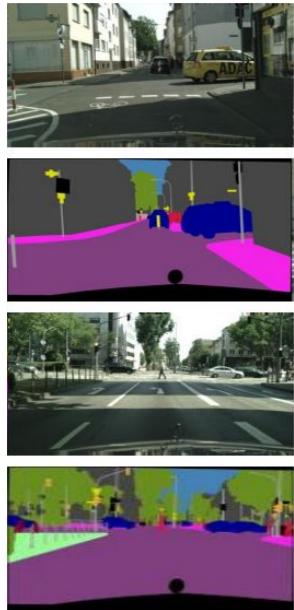
Ablation Studies



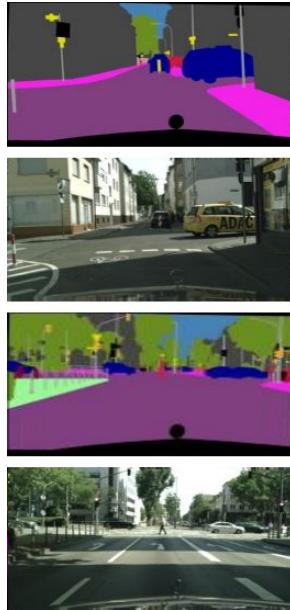
Available from: <https://arxiv.org/abs/1703.10593>

Ablation Studies

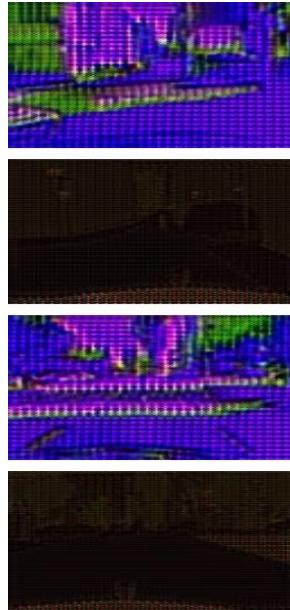
Input



Ground truth



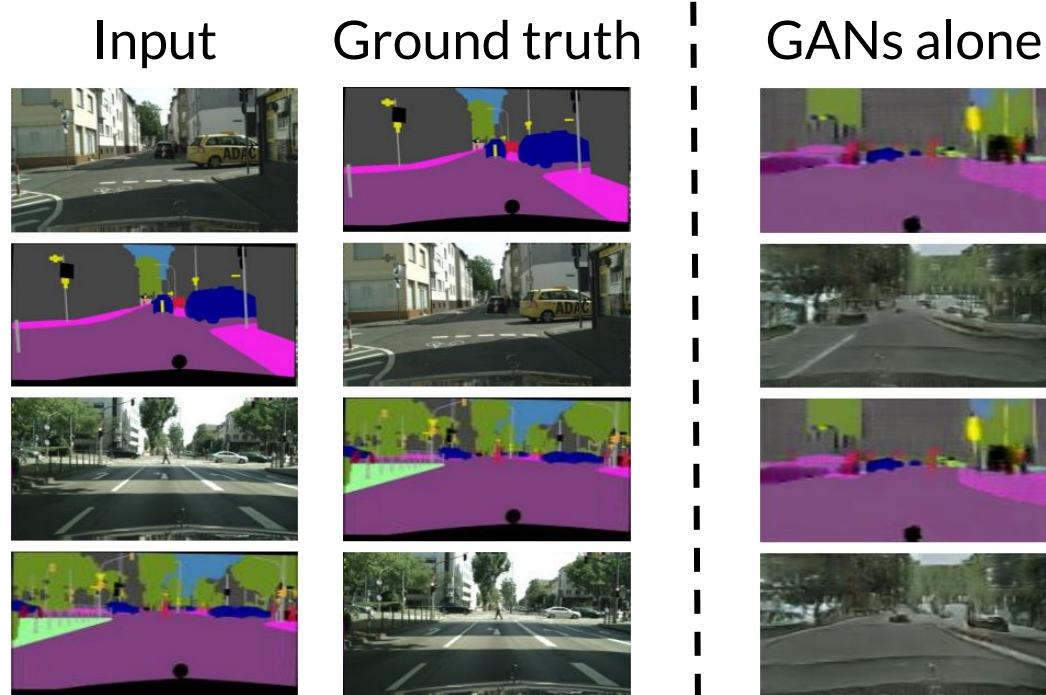
Cycle alone



Without Adversarial
GAN Loss, outputs are
not realistic

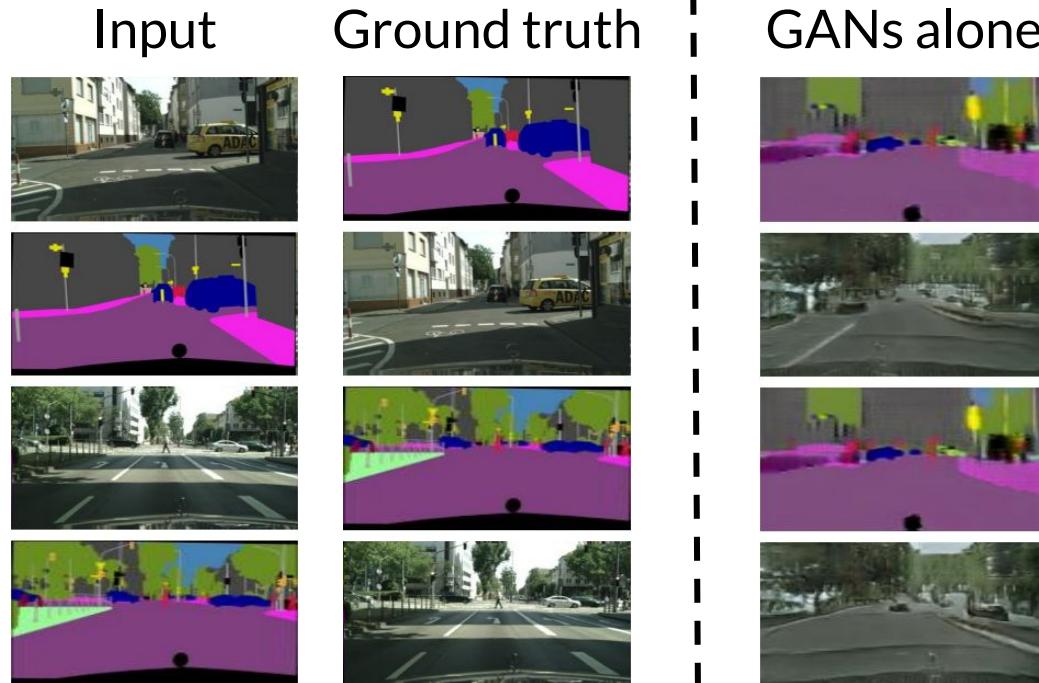
Available from: <https://arxiv.org/abs/1703.10593>

Ablation Studies



Available from: <https://arxiv.org/abs/1703.10593>

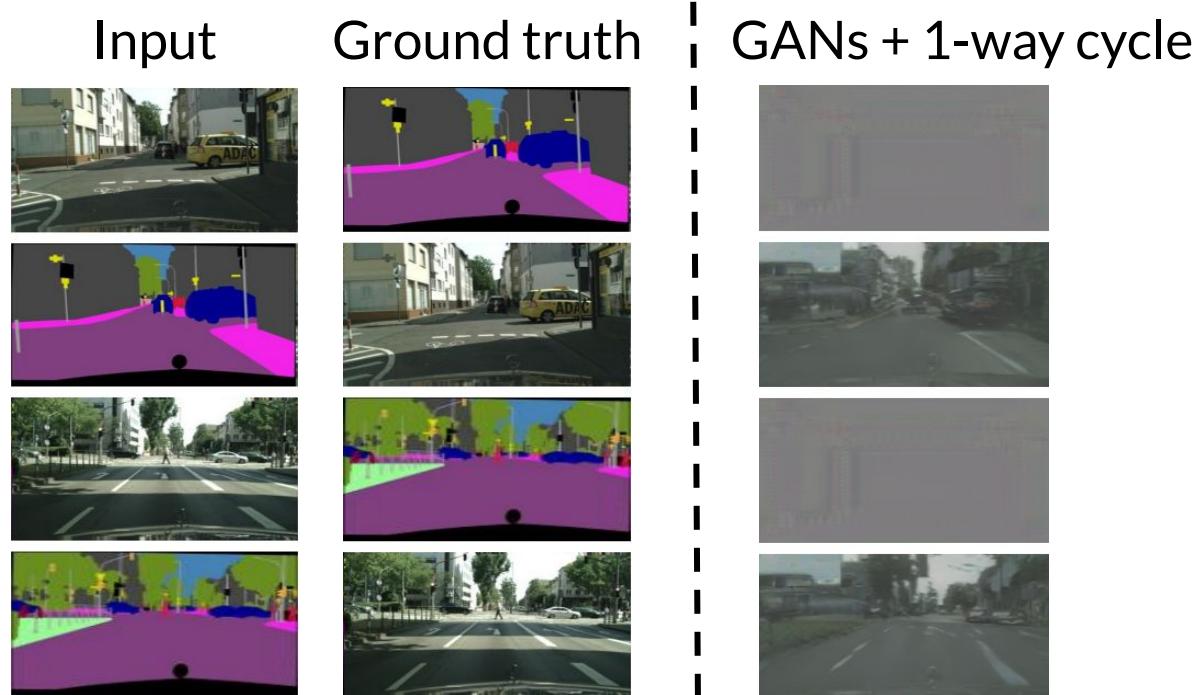
Ablation Studies



Without Cycle Consistency Loss, outputs show signs of mode collapse

Available from: <https://arxiv.org/abs/1703.10593>

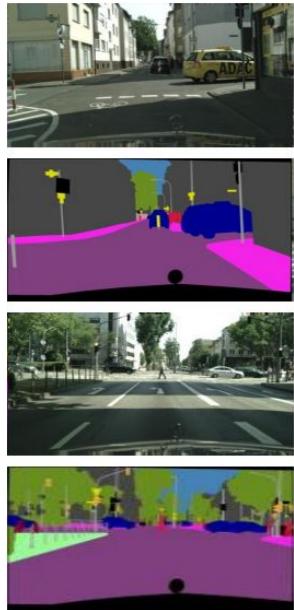
Ablation Studies



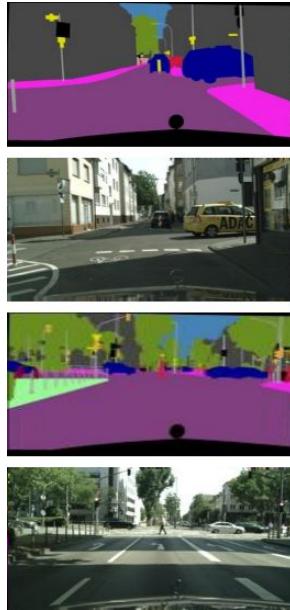
Available from: <https://arxiv.org/abs/1703.10593>

Ablation Studies

Input



Ground truth



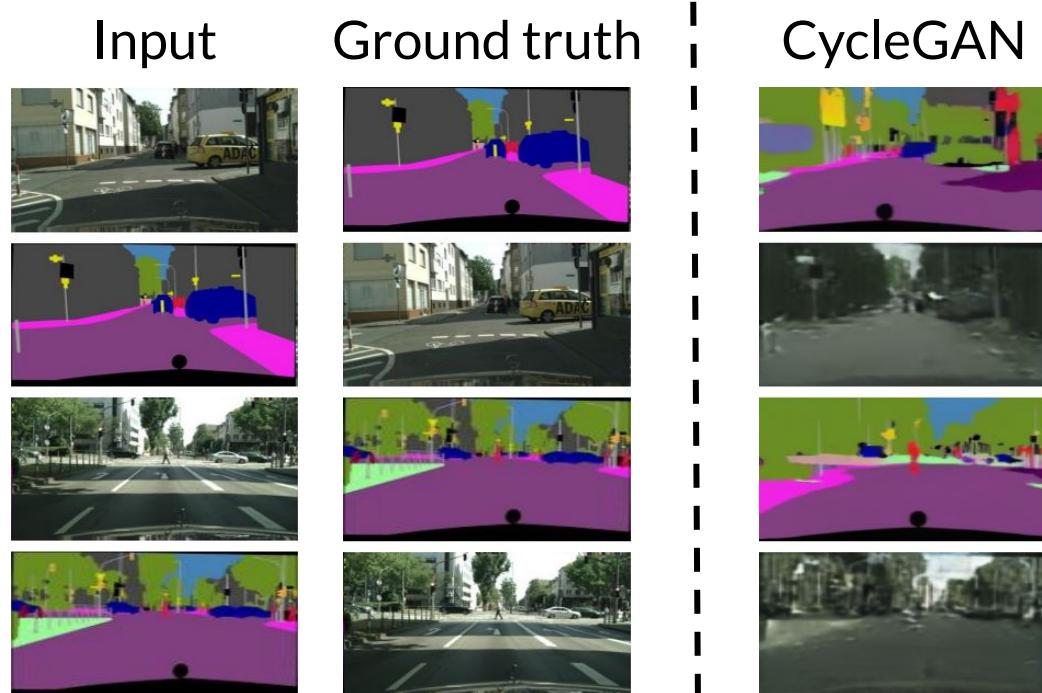
GANs + 1-way cycle



Without **full** Cycle
Consistency Loss, outputs
see mode collapse too

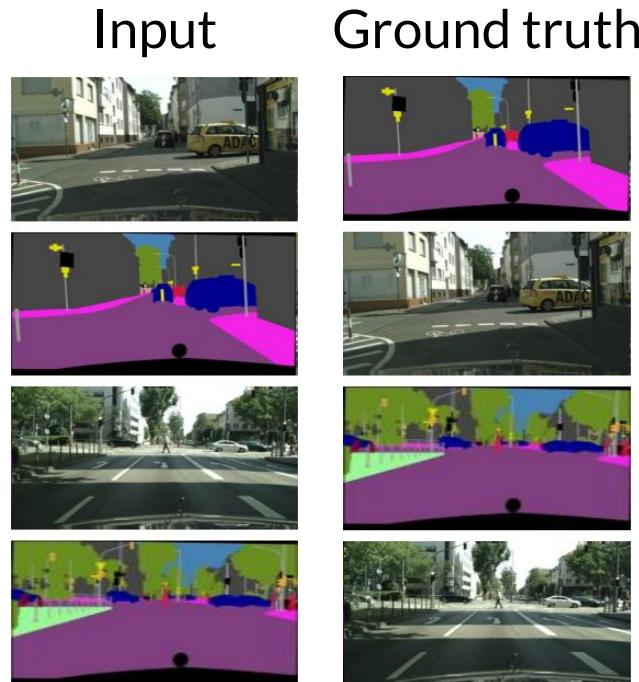
Available from: <https://arxiv.org/abs/1703.10593>

Ablation Studies



Available from: <https://arxiv.org/abs/1703.10593>

Ablation Studies



CycleGAN

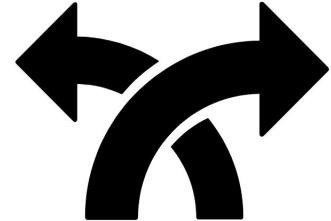


**CycleGAN uses both
Adversarial Loss and
Cycle Consistency Loss**

Available from: <https://arxiv.org/abs/1703.10593>

Summary

- Cycle consistency helps transfer uncommon style elements between the two GANs, while maintaining common content
- Add an extra loss term to each generator to softly encourage cycle consistency
- Cycle consistency is used in both directions



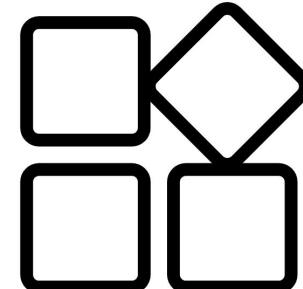


deeplearning.ai

CycleGAN: Least Squares Loss

Outline

- Least squares in statistics
- Least Squares Loss in GANs
 - Discriminator
 - Generator



Least Squares Loss: Another GAN Loss Function

- Came out when training stability was a big problem in GANS
 - Similar time to WGAN-GP

Least Squares Loss: Another GAN Loss Function

- Came out when training stability was a big problem in GANS
 - Similar time to WGAN-GP
- Helps with vanishing gradients and mode collapse



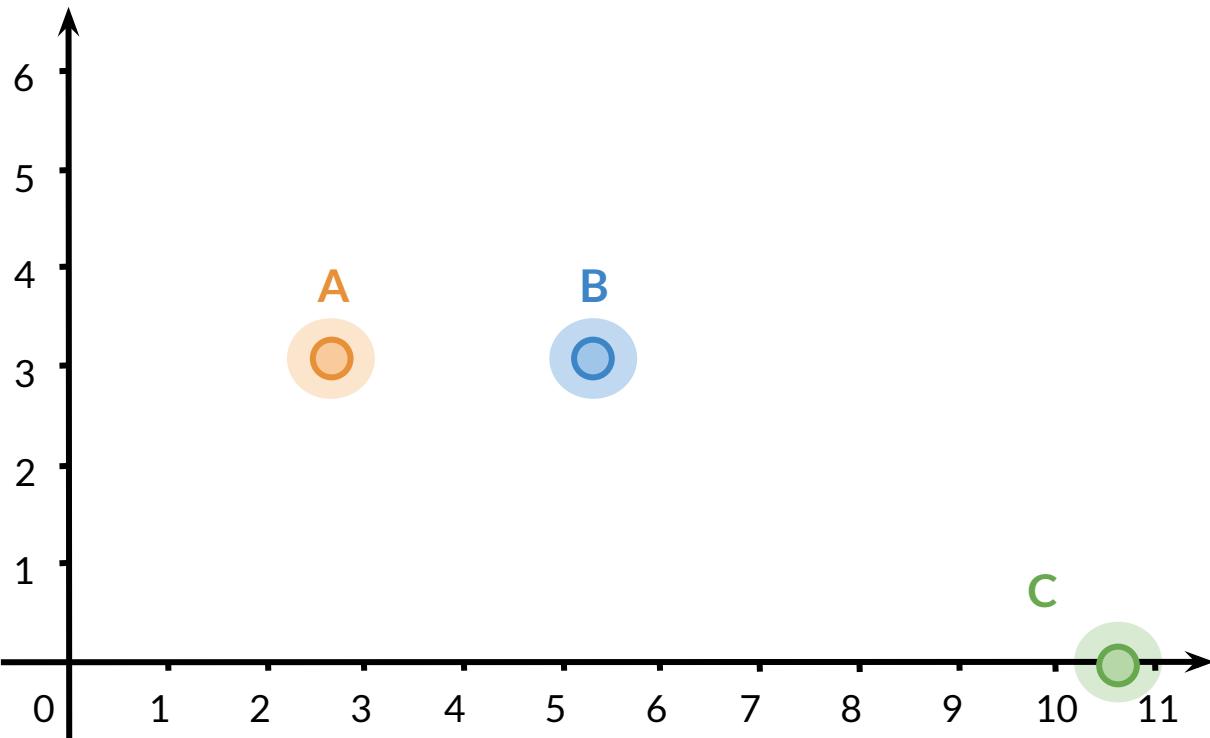
Least Squares Loss: Another GAN Loss Function

- Came out when training stability was a big problem in GANS
 - Similar time to WGAN-GP
- Helps with vanishing gradients and mode collapse

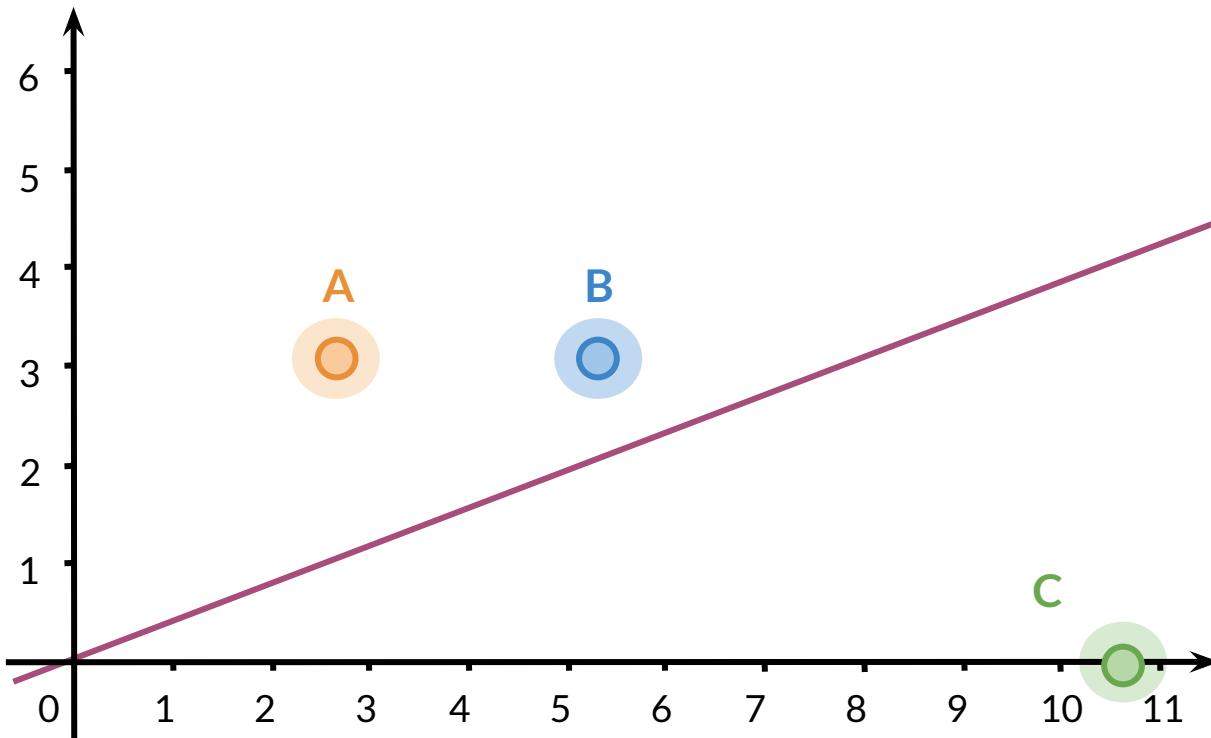


- GAN loss functions are chosen empirically

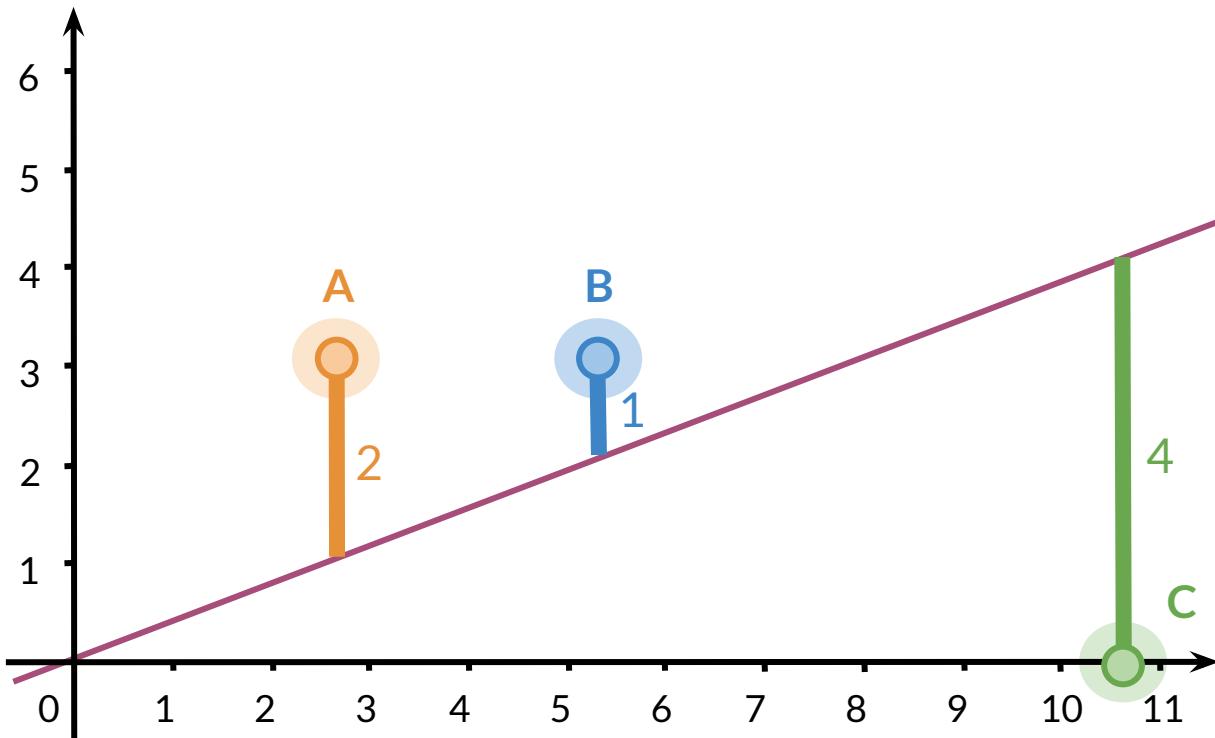
Least Squares



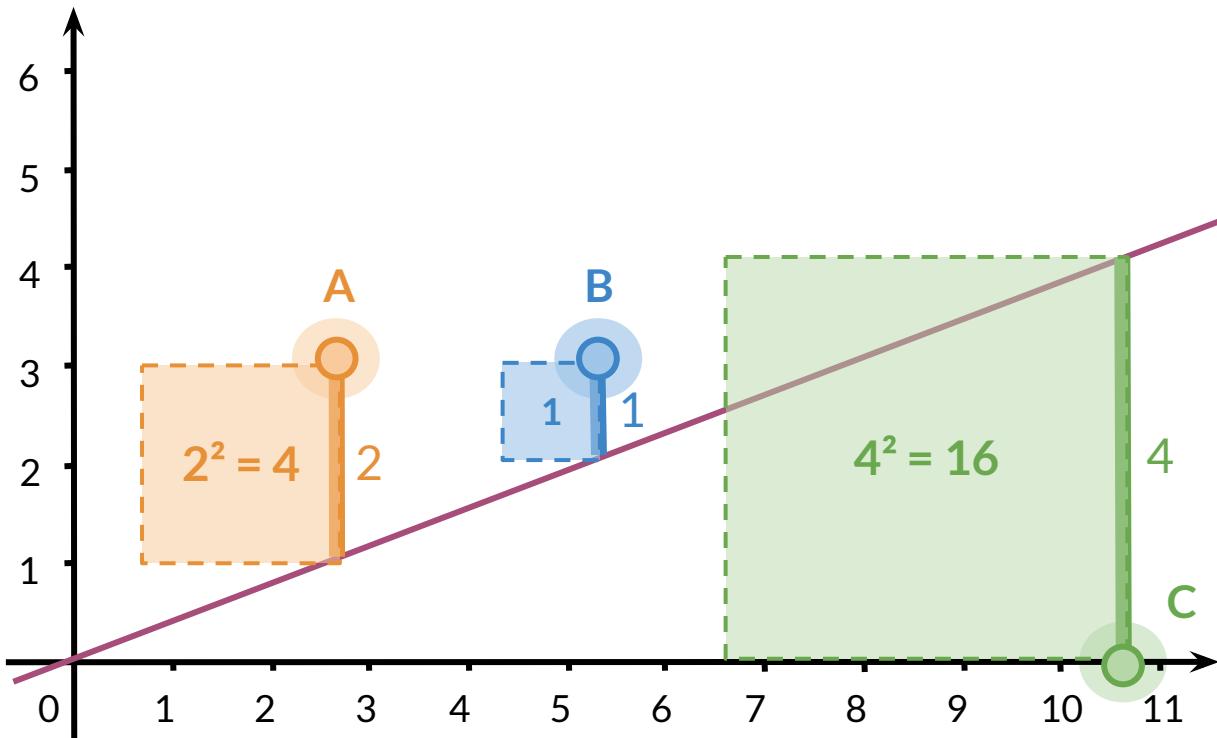
Least Squares



Least Squares

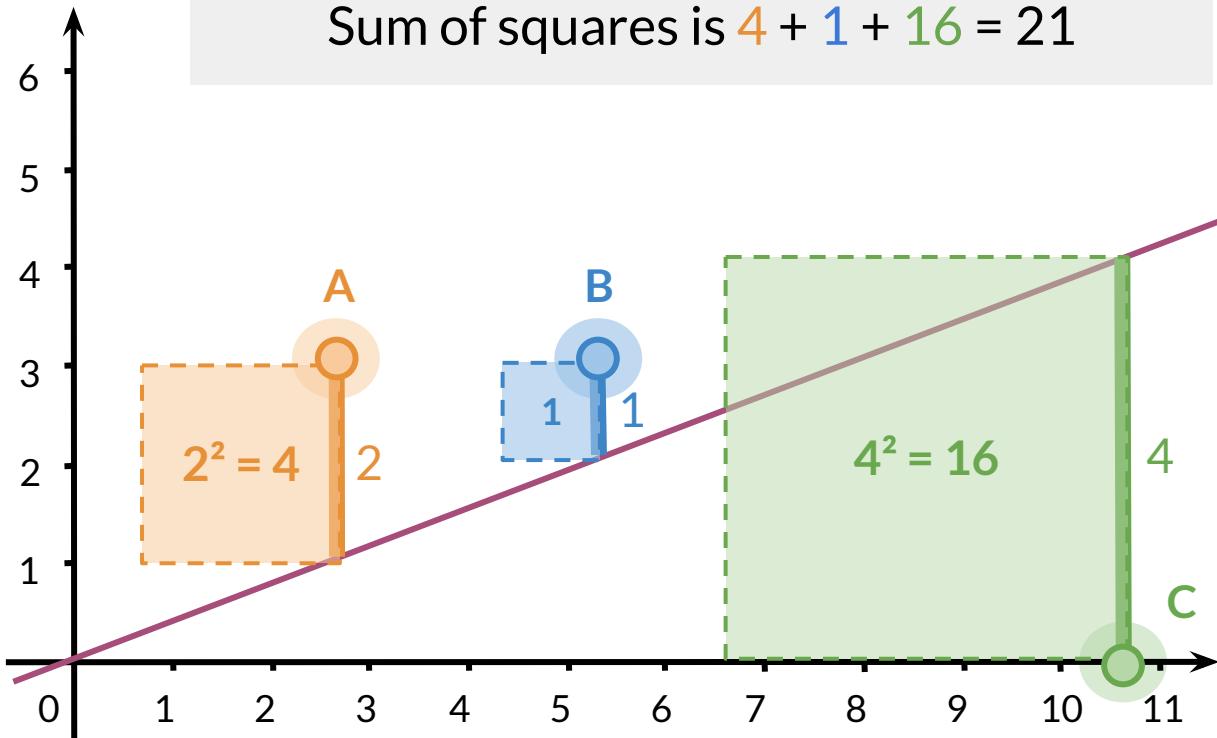


Least Squares



Least Squares

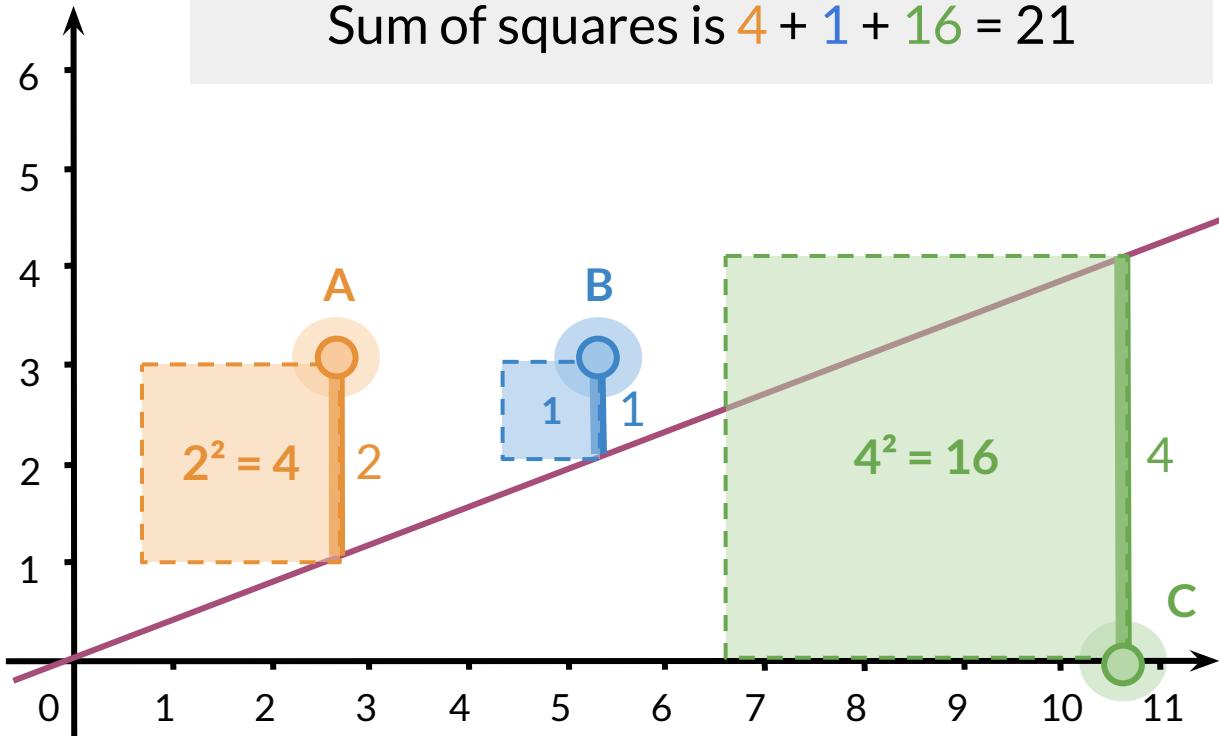
Sum of squares is $4 + 1 + 16 = 21$



Least Squares

Minimize
sum of squares

Sum of squares is $4 + 1 + 16 = 21$



Least Squares Loss: Discriminator

$$(D(\mathbf{x}) - 1)^2$$



Discriminator classification
of real image \mathbf{x}

Least Squares Loss: Discriminator

$$\mathbb{E}_{\mathbf{x}}[(D(\mathbf{x}) - 1)^2]$$

Least Squares Loss: Discriminator

$$\mathbb{E}_{\mathbf{x}} [(D(\mathbf{x}) - 1)^2] + (D(G(\mathbf{z})) - 0)^2$$

Discriminator classification
of fake image $G(\mathbf{z})$

Least Squares Loss: Discriminator

$$\mathbb{E}_{\mathbf{x}}[(D(\mathbf{x}) - 1)^2] + \mathbb{E}_{\mathbf{z}}[(D(G(\mathbf{z})) - 0)^2]$$

Least Squares Loss: Discriminator

$$\mathbb{E}_{\mathbf{x}}[(D(\mathbf{x}) - 1)^2] + \mathbb{E}_{\mathbf{z}}[(D(G(\mathbf{z})))^2]$$

Least Squares Loss: Generator

$$\mathbb{E}_z [(D(G(z)) - 1)^2]$$

Least Squares Loss

Discriminator
Loss

$$\mathbb{E}_{\mathbf{x}}[(D(\mathbf{x}) - 1)^2] + \mathbb{E}_{\mathbf{z}}[(D(G(\mathbf{z})))^2]$$

Least Squares Loss

Discriminator
Loss

$$\mathbb{E}_{\mathbf{x}} [(D(\mathbf{x}) - 1)^2] + \mathbb{E}_{\mathbf{z}} [(D(G(\mathbf{z})))^2]$$

Generator
Loss

$$\mathbb{E}_{\mathbf{z}} [(D(G(\mathbf{z}))) - 1)^2]$$

Least Squares Loss

Discriminator
Loss

$$\mathbb{E}_{\mathbf{x}} [(D(\mathbf{x}) - 1)^2] + \mathbb{E}_{\mathbf{z}} [(D(G(\mathbf{z})))^2]$$

Generator
Loss

$$\mathbb{E}_{\mathbf{z}} [(D(G(\mathbf{z}))) - 1)^2]$$

Reduces vanishing gradient problem

Least Squares Loss

Discriminator
Loss

$$\mathbb{E}_{\mathbf{x}} [(D(\mathbf{x}) - 1)^2] + \mathbb{E}_{\mathbf{z}} [(D(G(\mathbf{z})))^2]$$

Generator
Loss

$$\mathbb{E}_{\mathbf{z}} [(D(G(\mathbf{z}))) - 1)^2]$$

Also known as Mean Squared Error!

Context of Least Squares Loss

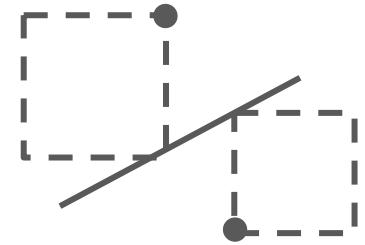
Adversarial Loss + λ * Cycle Consistency Loss



Least Squares Loss

Summary

- Least squares fits a line from several points
- Least Squares Loss is used as the Adversarial Loss function in CycleGAN
- More stable than BCELoss, since the gradient is only flat when prediction is exactly correct



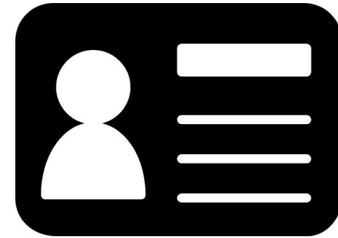


deeplearning.ai

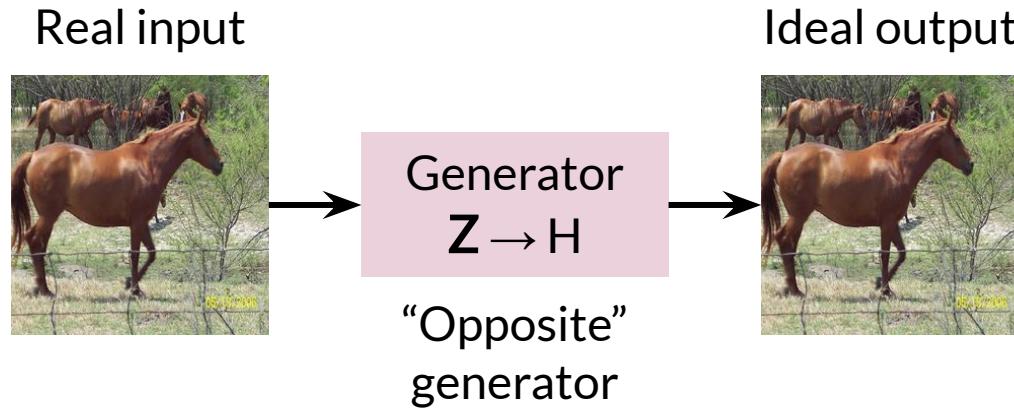
CycleGAN: Identity Loss

Outline

- Identity Loss
 - How it works
 - Impact on outputs

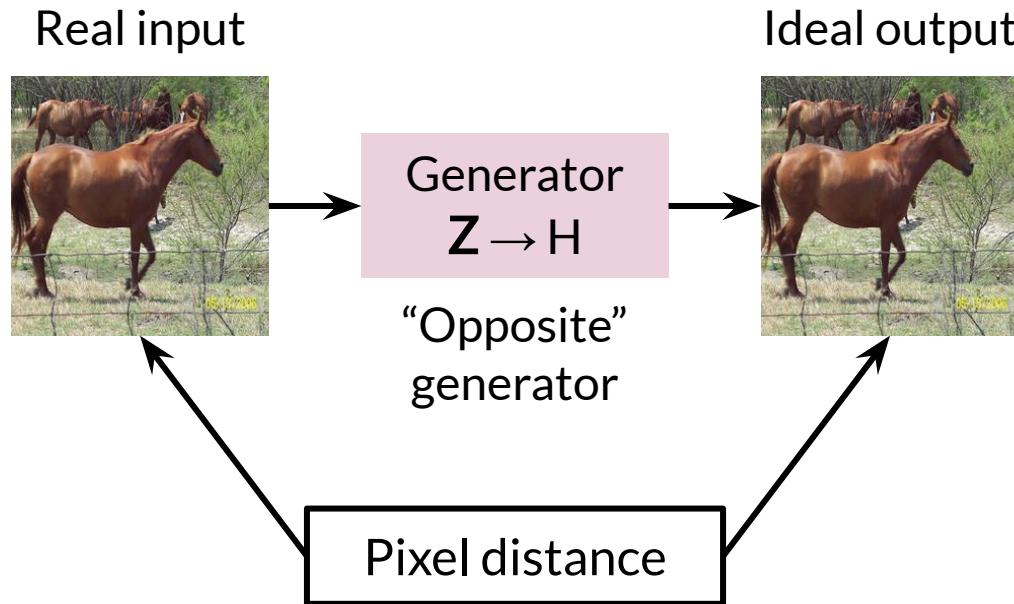


Identity Loss



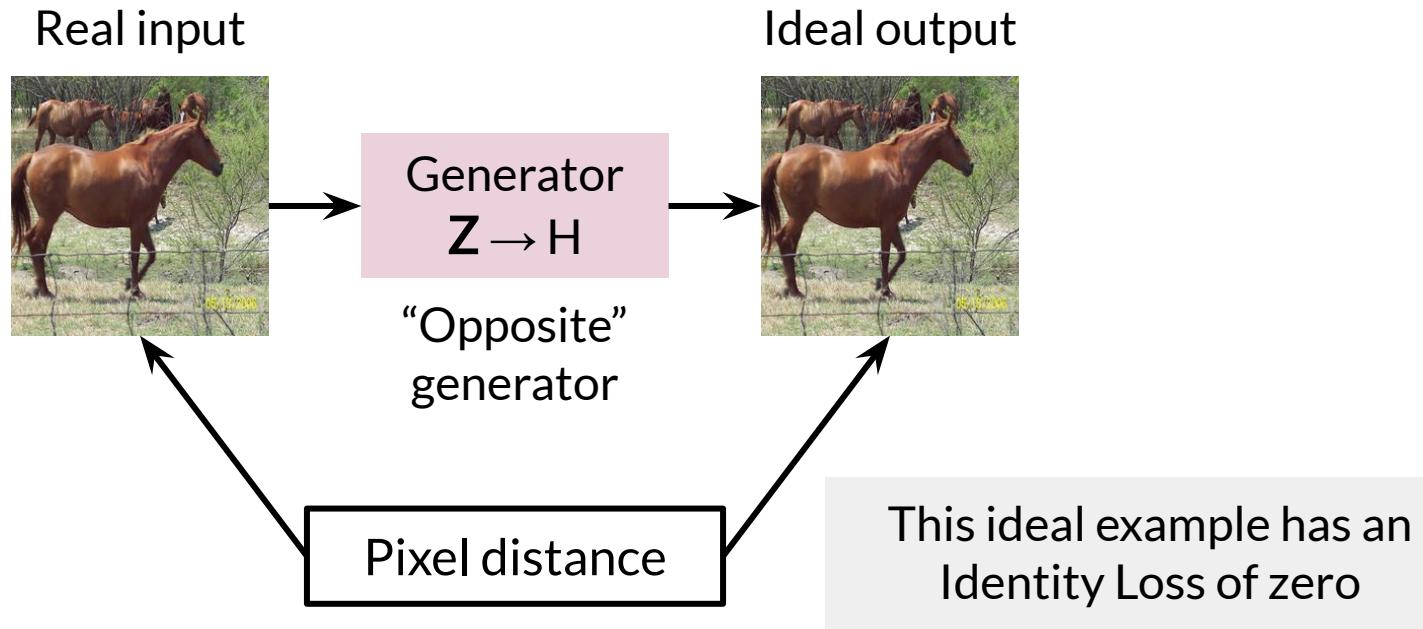
Images available from: <https://github.com/togheppi/CycleGAN>

Identity Loss



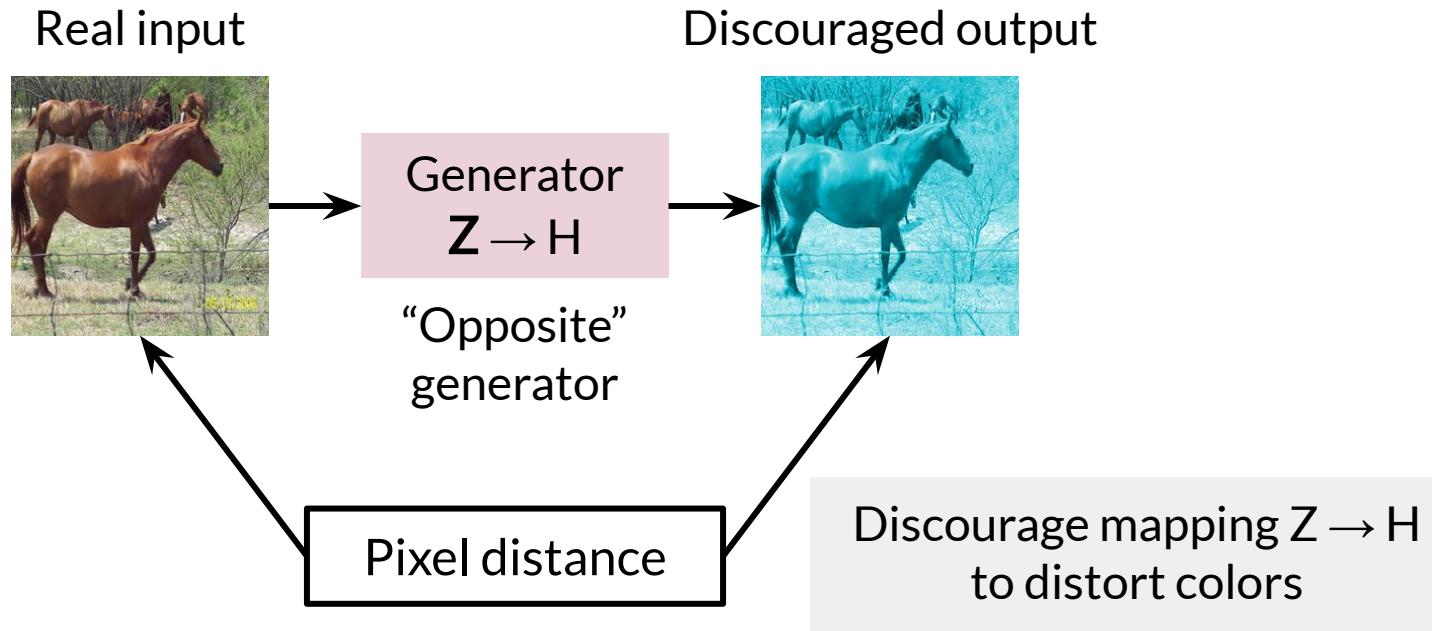
Images available from: <https://github.com/togheppi/CycleGAN>

Identity Loss



Images available from: <https://github.com/togheppi/CycleGAN>

Identity Loss



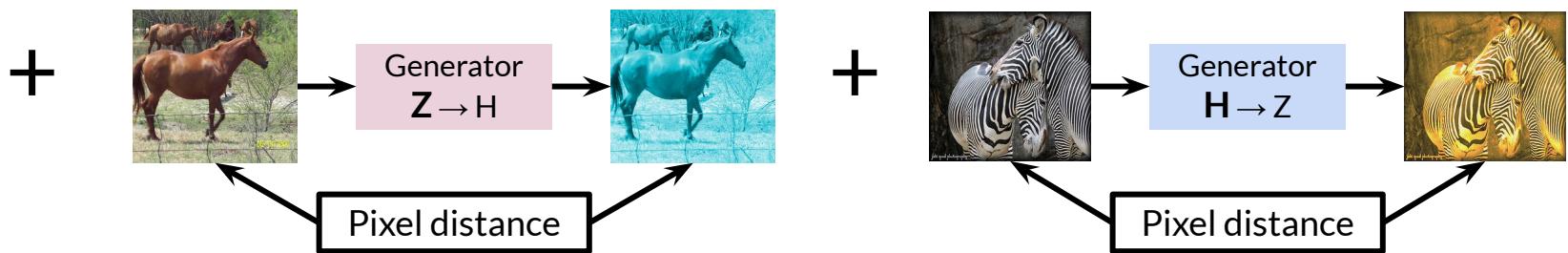
Images available from: <https://github.com/togheppi/CycleGAN>

Context of Identity Loss

Adversarial Loss + λ * Cycle Consistency Loss

Context of Identity Loss

Adversarial Loss + λ * Cycle Consistency Loss



Images available from: <https://github.com/togheppi/CycleGAN>

Context of Identity Loss

Adversarial Loss + λ * Cycle Consistency Loss

+ Identity Loss

Context of Identity Loss

Adversarial Loss + λ_1^* Cycle Consistency Loss

+ λ_2^* Identity Loss

Identity Loss Example: Photo → Monet



Identity Loss helps preserve
original photo color

Available from: <https://arxiv.org/abs/1703.10593>

Summary

- Identity Loss takes a real image in domain B and inputs it into Generator: $A \rightarrow B$, expecting an identity mapping
 - An identity mapping means the output is the same as the input
- Pixel distance is used
 - Ideally, no difference between input and output!
- Identity Loss is optionally added to help with color preservation



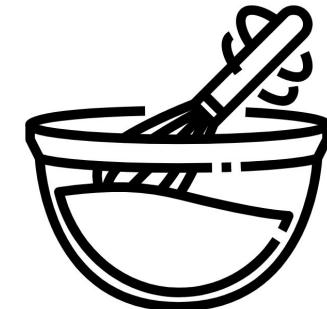


deeplearning.ai

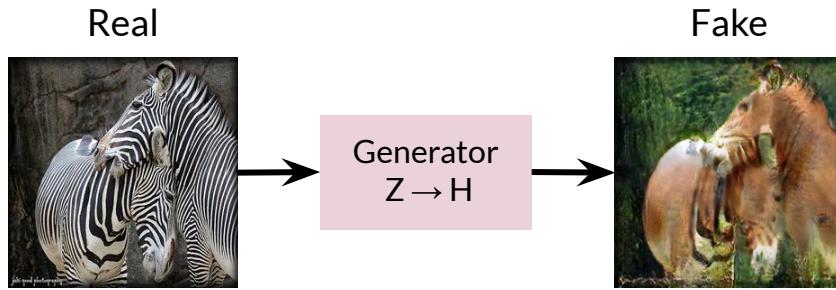
CycleGAN: Putting It All Together

Outline

- Putting CycleGAN together!
 - Two GANs
 - Cycle Consistency Loss
 - Least Squares Adversarial Loss
 - Identity Loss (optional)

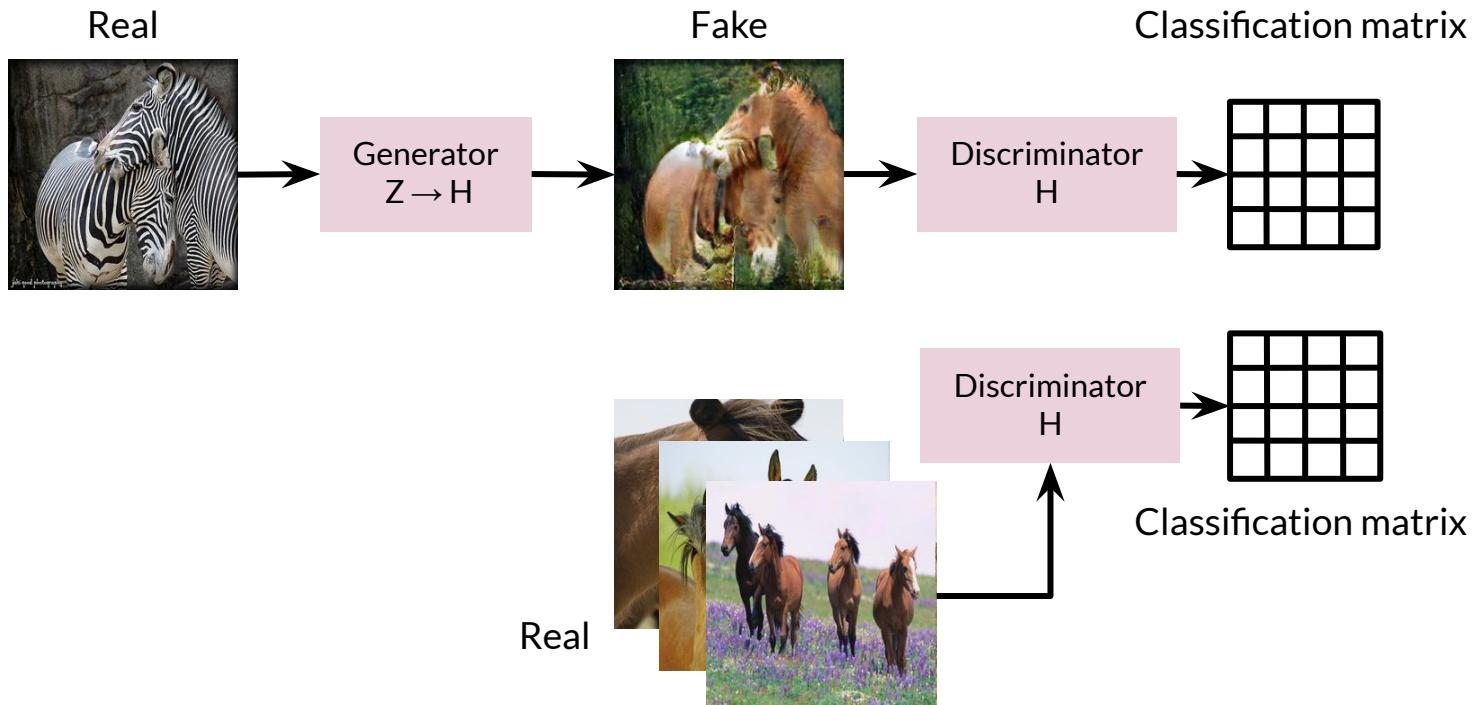


CycleGAN



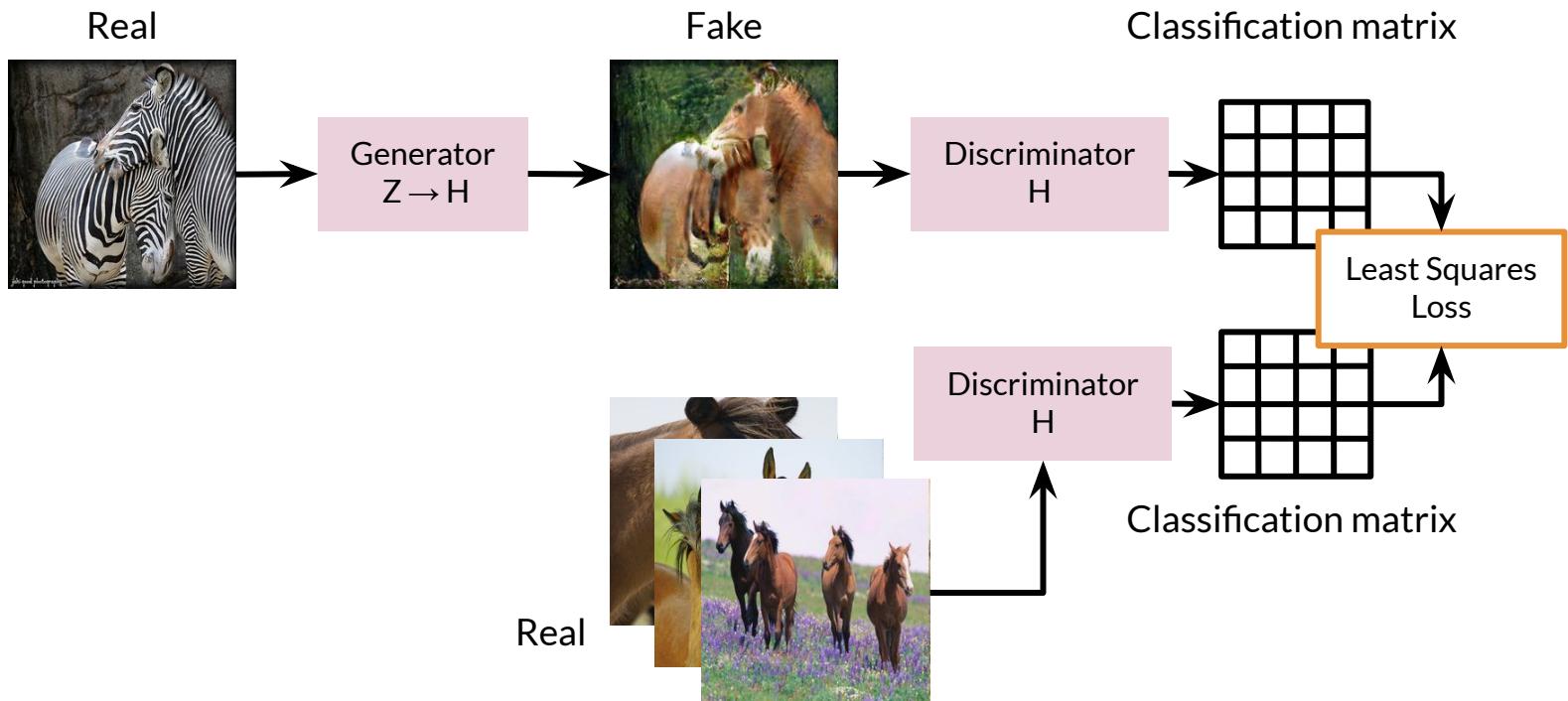
Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN



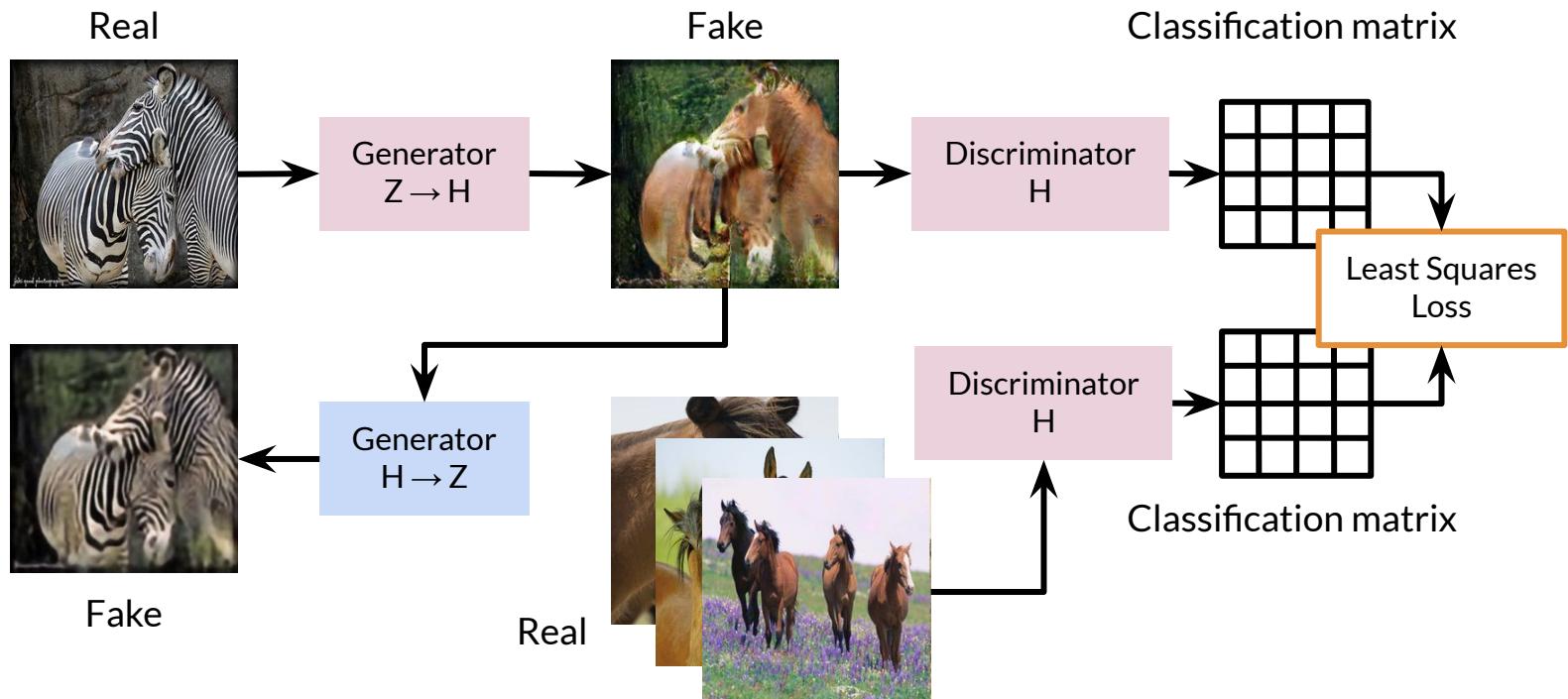
Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN



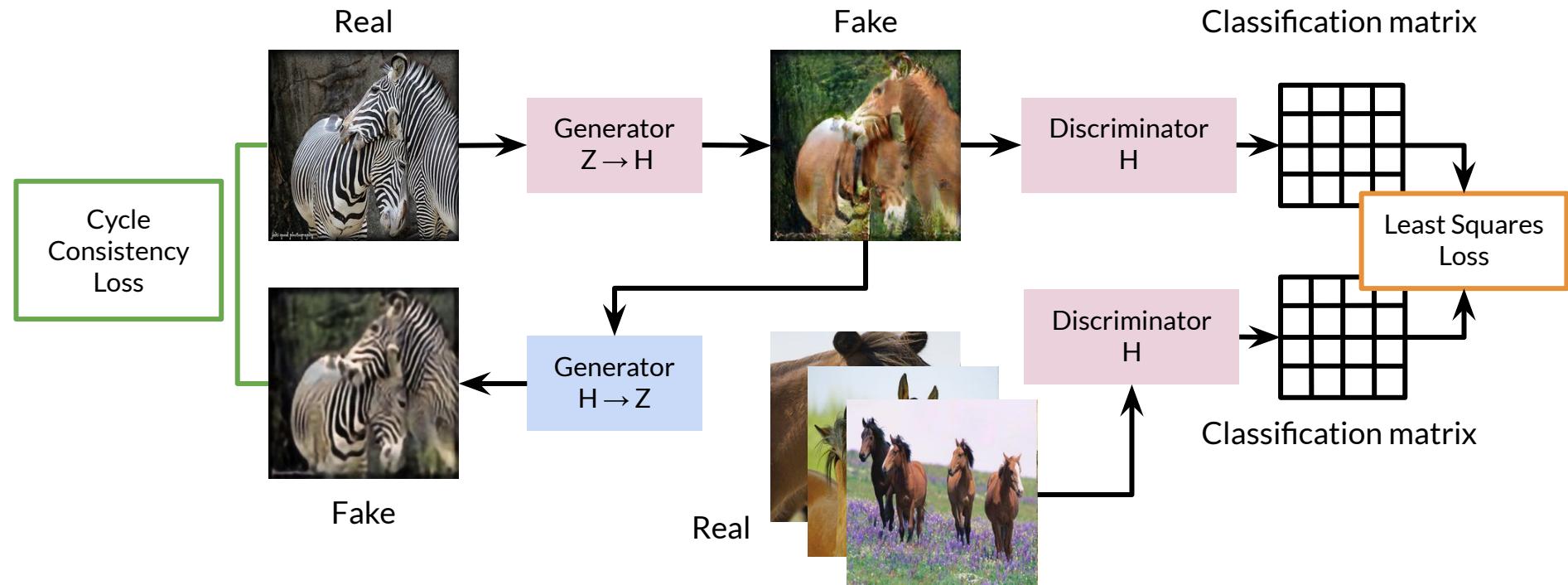
Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN



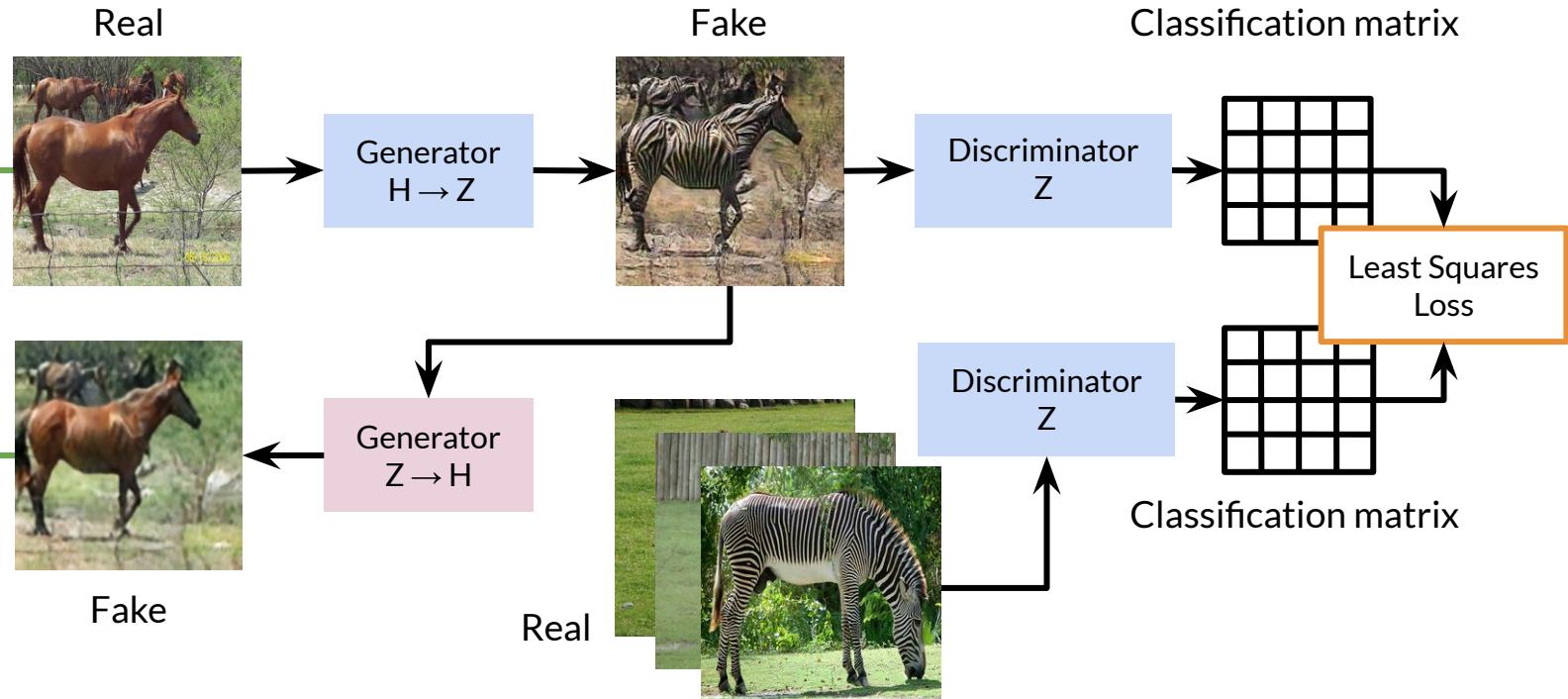
Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN



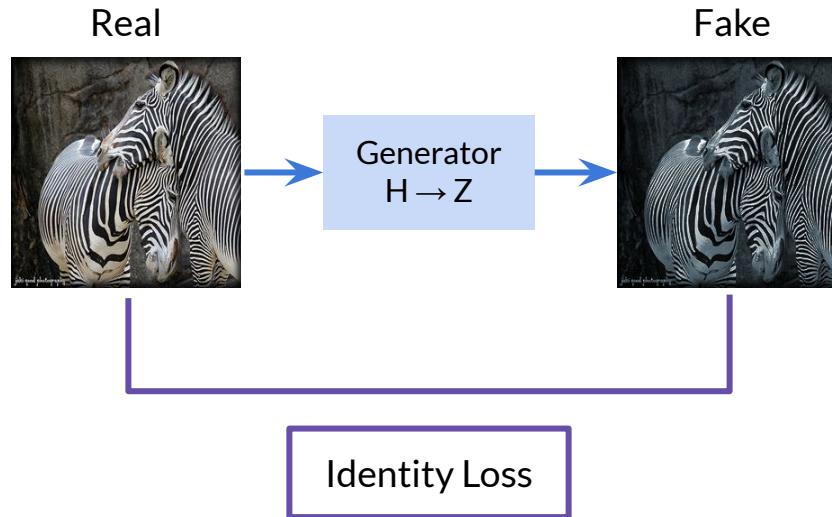
Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN



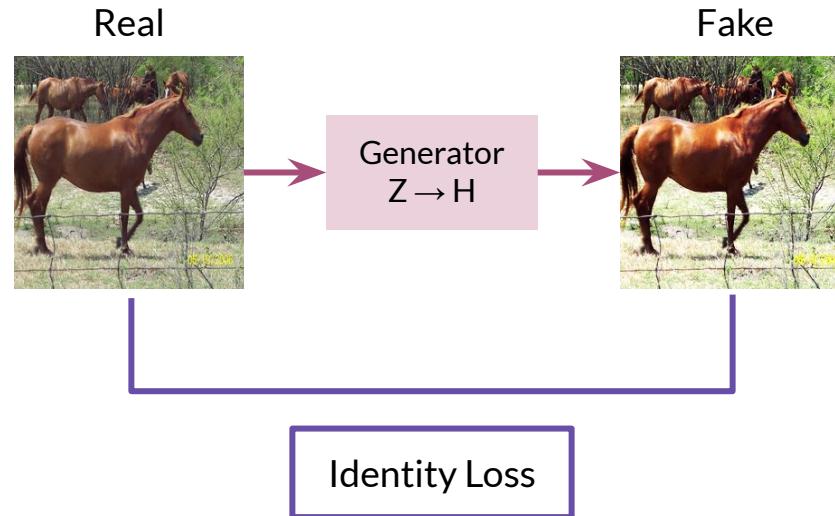
Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN



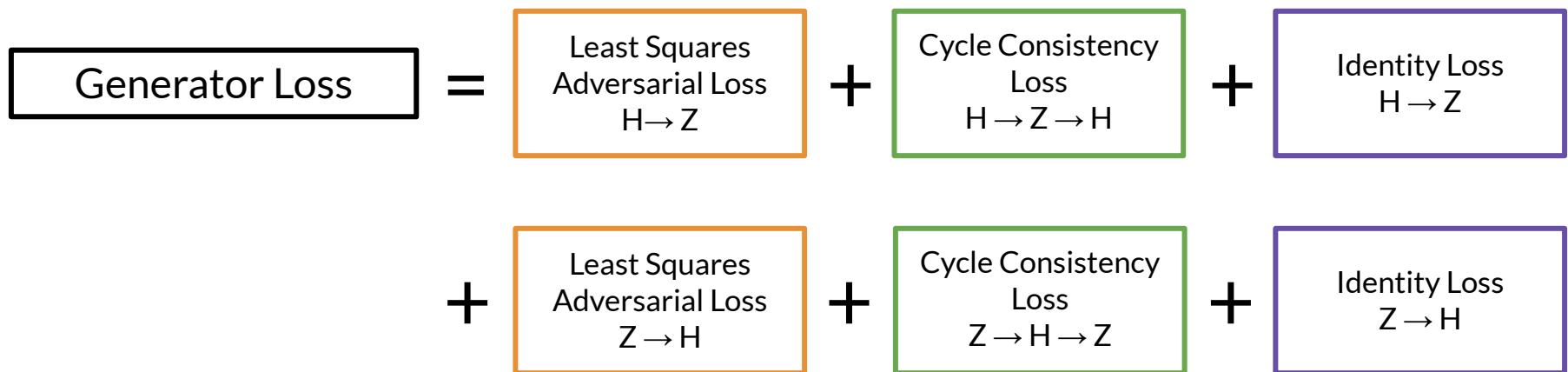
Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN



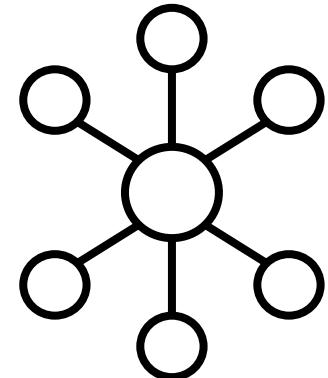
Images available from: <https://github.com/togheppi/CycleGAN>

CycleGAN Loss



Summary

- CycleGAN is composed of two GANs
- Generators have 6 loss terms in total, 3 each:
 - Least Squares Adversarial Loss
 - Cycle Consistency Loss
 - Identity Loss
- Discriminator is simpler, with BCELoss using PatchGAN





deeplearning.ai

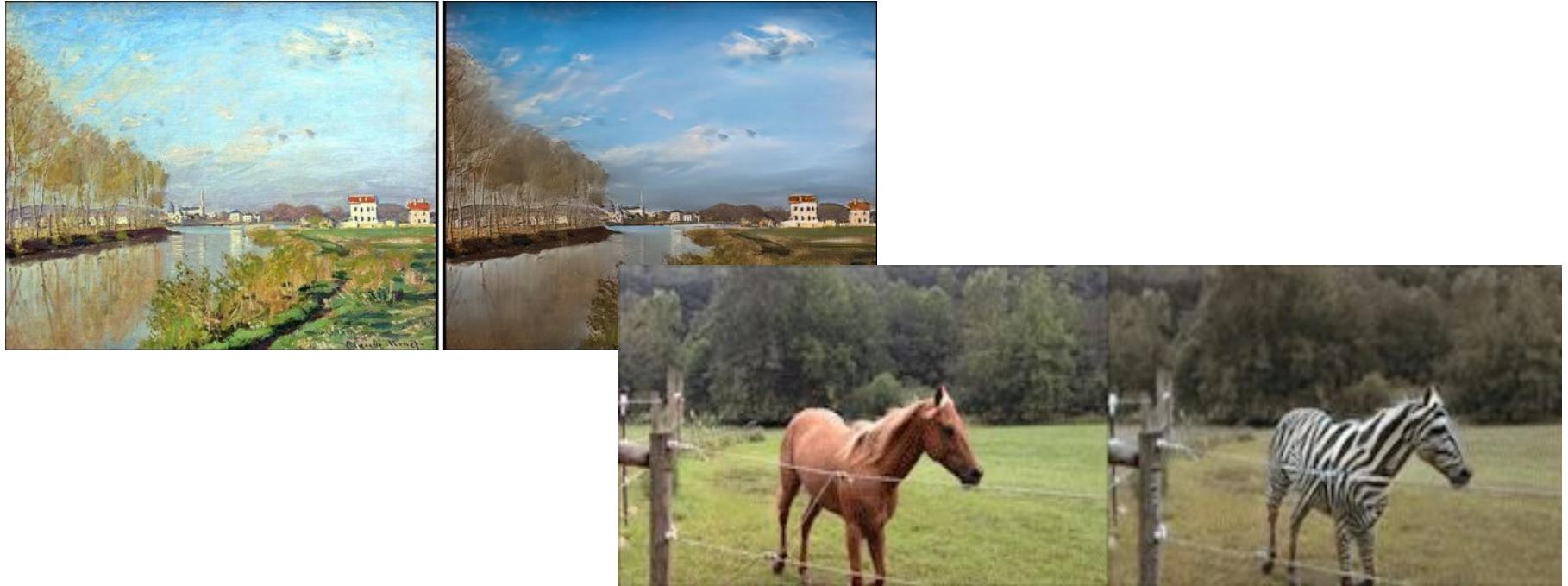
CycleGAN Applications & Variants

Outline

- Overview of some CycleGAN applications
- Some variants of unpaired image-to-image translation

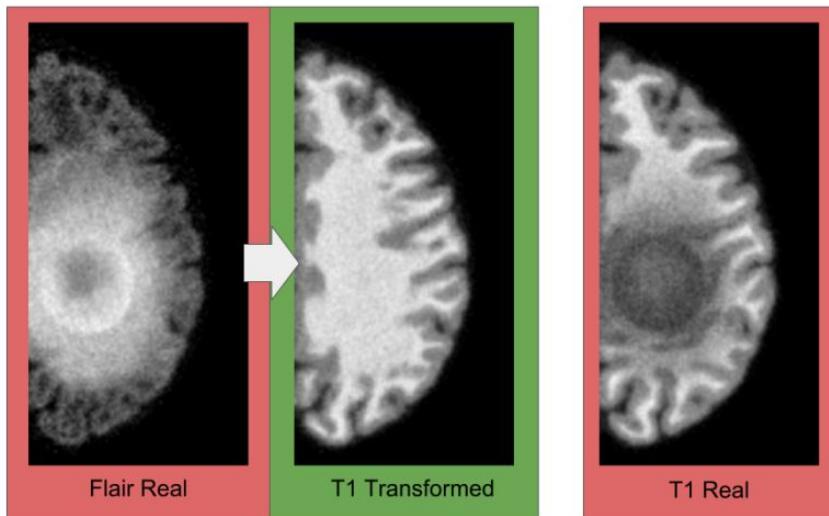


Applications

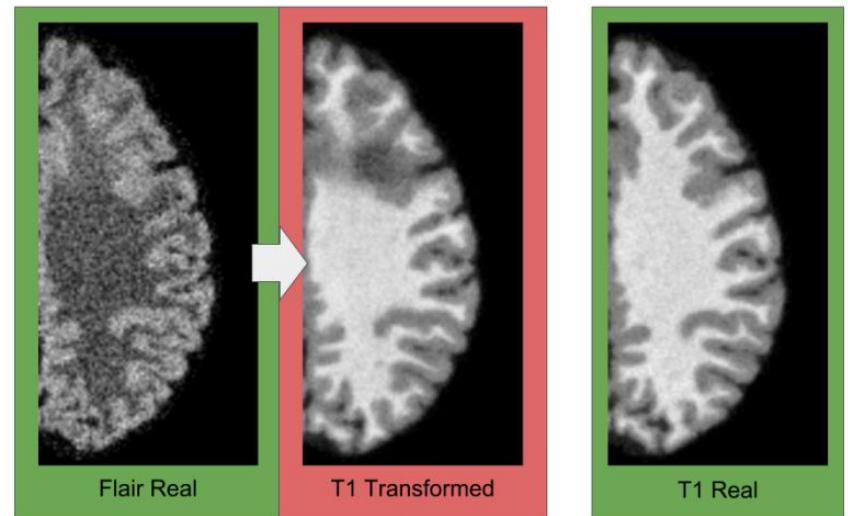


Available from: <https://arxiv.org/abs/1611.07004>

Applications



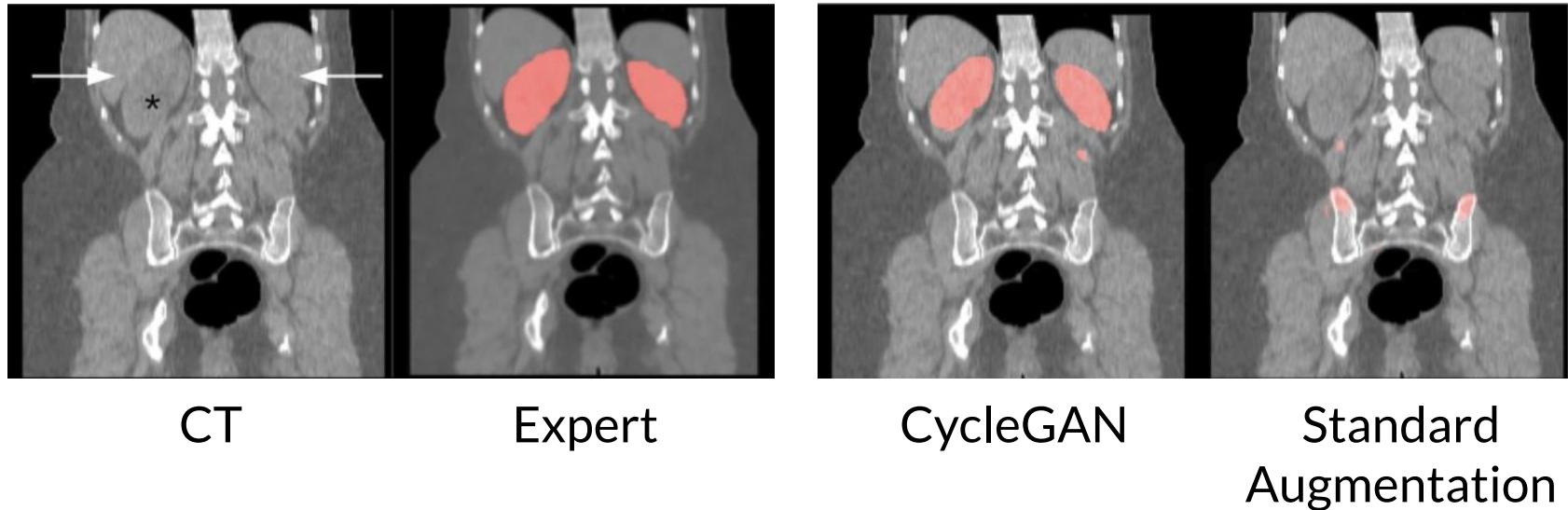
(a) A translation removing tumors



(b) A translation adding tumors

Available from: <https://arxiv.org/abs/1805.08841>

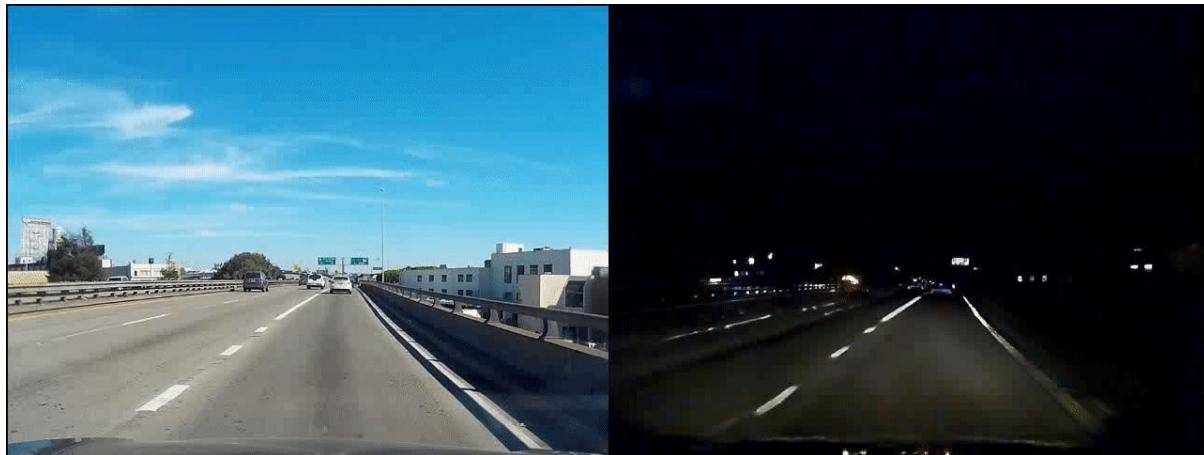
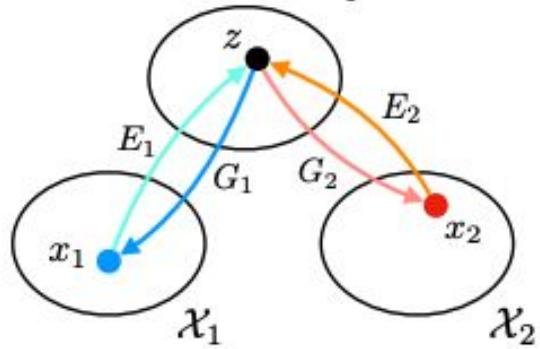
Applications



Available from: <https://www.nature.com/articles/s41598-019-52737-x.pdf>

Variant: UNIT

\mathcal{Z} : shared latent space



Available from: <https://github.com/mingyuliutw/UNIT>

Variant: Multimodal UNIT (MUNIT)



Available from: <https://github.com/NVlabs/MUNIT>

Variant: Multimodal UNIT (MUNIT)



Available from: <https://github.com/NVlabs/MUNIT>

Summary

- Various applications of CycleGAN including:
 - Democratized art and style transfer
 - Medical data augmentation
 - Creating paired data
- UNIT and MUNIT are other models for unpaired (unsupervised) image-to-image translation

