Machine Learning - Part 5

Apr. 22, 2025

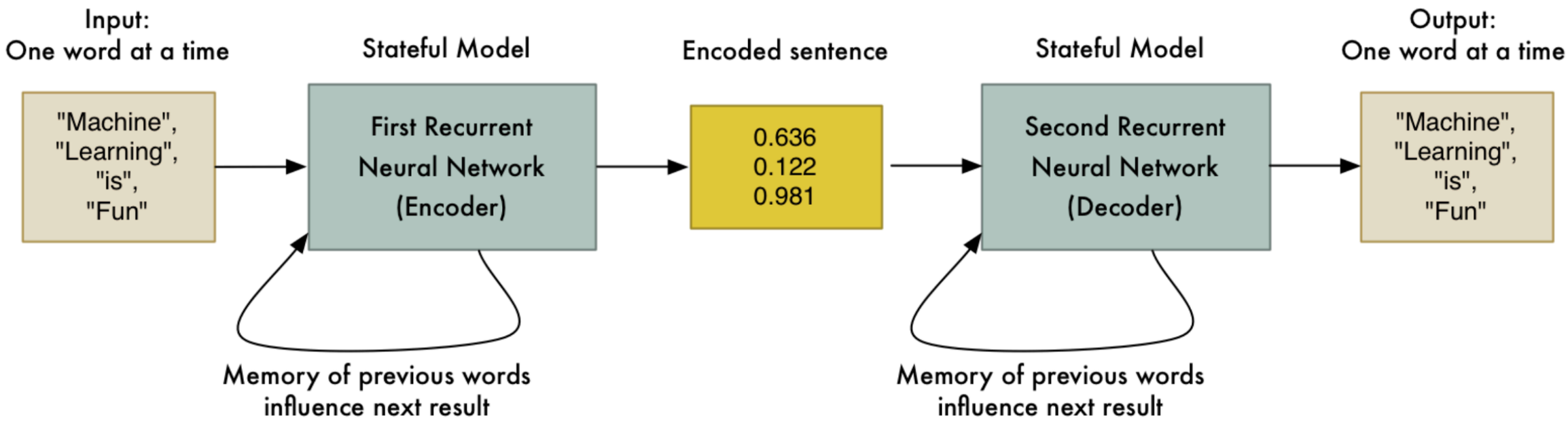
Recap slides:

128 Measurements Generated from Image

0.097496084868908	0.045223236083984	-0.1281466782093	0.032084941864014
0.12529824674129	0.060309179127216	0.17521631717682	0.020976085215807
0.030809439718723	-0.01981477253139	0.10801389068365	-0.00052163278451189
0.036050599068403	0.065554238855839	0.0731306001544	-0.1318951100111
-0.097486883401871	0.1226262897253	-0.029626874253154	-0.0059557510539889
-0.0066401711665094	0.036750309169292	-0.15958009660244	0.043374512344599
-0.14131525158882	0.14114324748516	-0.031351584941149	-0.053343612700701
-0.048540540039539	-0.061901587992907	-0.15042643249035	0.078198105096817
-0.12567175924778	-0.10568545013666	-0.12728653848171	-0.076289616525173
-0.061418771743774	-0.074287034571171	-0.065365232527256	0.12369467318058
0.046741496771574	0.0061761881224811	0.14746543765068	0.056418422609568
-0.12113650143147	-0.21055991947651	0.0041091227903962	0.089727647602558
0.061606746166945	0.11345765739679	0.021352224051952	-0.0085843298584223
0.061989940702915	0.19372203946114	-0.086726233363152	-0.022388197481632
0.10904195904732	0.084853030741215	0.09463594853878	0.020696049556136
-0.019414527341723	0.0064811296761036	0.21180312335491	-0.050584398210049
0.15245945751667	-0.16582328081131	-0.035577941685915	-0.072376452386379
-0.12216668576002	-0.0072777755558491	-0.036901291459799	-0.034365277737379
0.083934605121613	-0.059730969369411	-0.070026844739914	-0.045013956725597
0.087945111095905	0.11478432267904	-0.089621491730213	-0.013955107890069
-0.021407851949334	0.14841195940971	0.078333757817745	-0.17898085713387
-0.018298890441656	0.049525424838066	0.13227833807468	-0.072600327432156
-0.011014151386917	-0.051016297191381	-0.14132921397686	0.0050511928275228
0.0093679334968328	-0.062812767922878	-0.13407498598099	-0.014829395338893
0.058139257133007	0.0048638740554452	-0.039491076022387	-0.043765489012003
-0.024210374802351	-0.11443792283535	0.071997955441475	-0.012062266469002
-0.057223934680223	0.014683869667351	0.05228154733777	0.012774495407939
0.023535015061498	-0.081752359867096	-0.031709920614958	0.069833360612392
-0.0098039731383324	0.037022035568953	0.11009479314089	0.11638788878918
0.020220354199409	0.12788131833076	0.18632389605045	-0.015336792916059
0.0040337680839002	-0.094398014247417	-0.11768248677254	0.10281457751989
0.051597066223621	-0.10034311562777	-0.040977258235216	-0.082041338086128

The idea of turning a face into a list of measurements is an example of an *encoding*. We are taking raw data (a picture of a face) and turning it into a list of measurements that represent it (the encoding).

Recap slides:



We can connect two RNNs end-to-end. The first RNN can generate the encoding that represents a sentence. Then the second RNN can take that encoding and just do the same logic in reverse to decode the original sentence again.

Machine Learning - Part 5

April 22, 2025

By the end of this lecture, you will be able to:

1. Explain what a Generative Adversarial Network is
2. Explain how style transfer works
3. Explain how to trick a neural network used for classification

Generative Adversarial Networks (GAN)

A GAN consists of a pair of neural networks competing to create more realistic outputs.

Generative Adversarial Networks (GAN)

A GAN consists of a pair of neural networks competing to create more realistic outputs.

Example: creating pictures of people who do not exist.

Generative Adversarial Networks (GAN)

A GAN consists of a pair of neural networks competing to create more realistic outputs.

Example: creating pictures of people who do not exist.

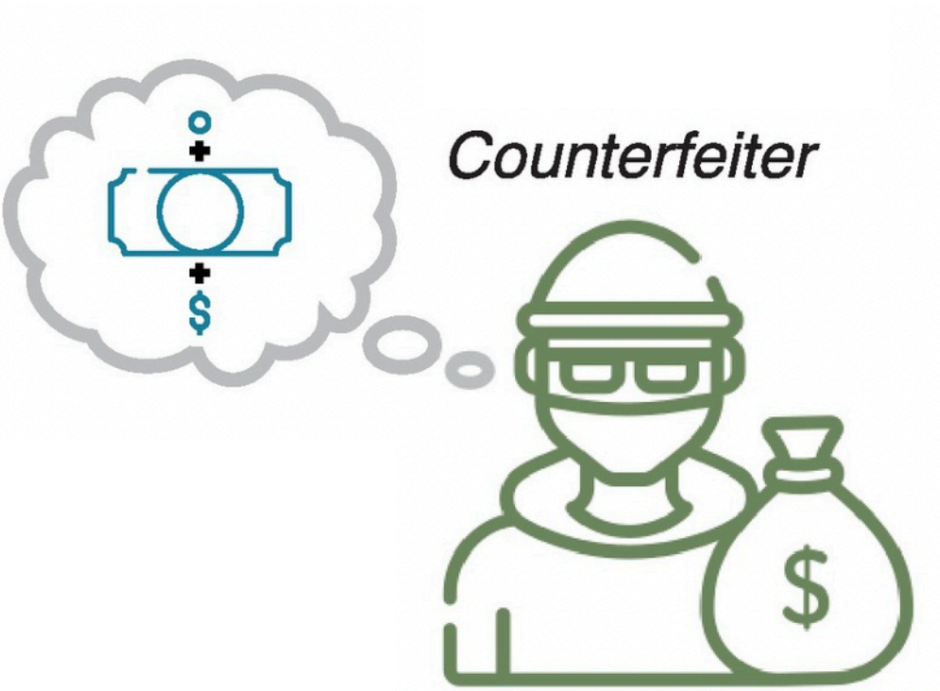
GANs were first invented in 2014 and the techniques have improved substantially since.

Generative Adversarial Networks (GAN)

https://www.whichfaceisreal.com/index.php

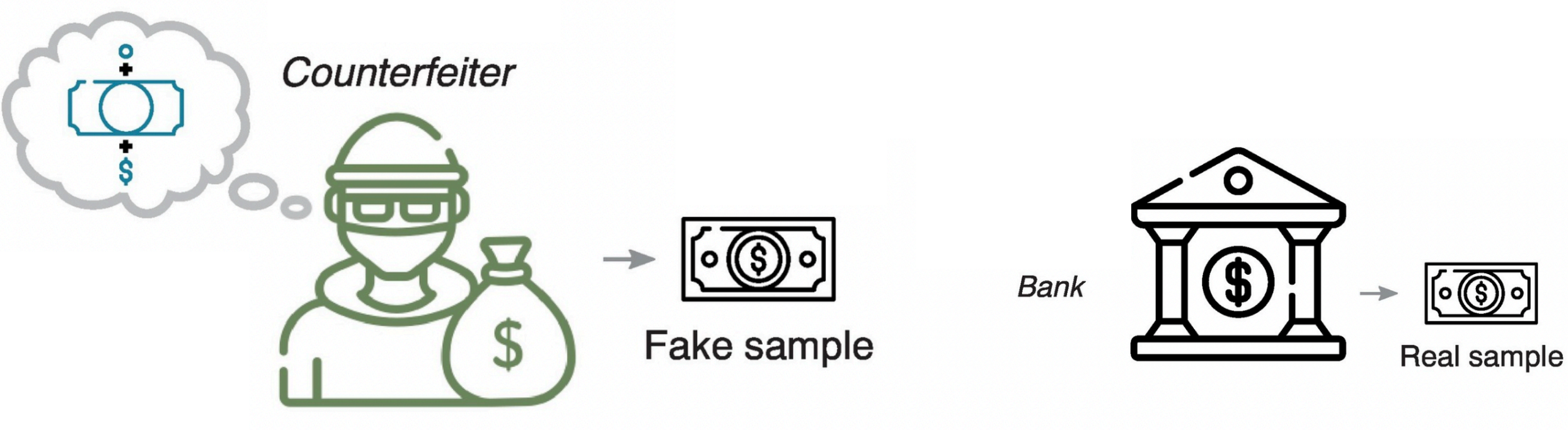
Generative Adversarial Networks (GAN)

Imagine that you are a counterfeiter trying to produce fake dollar bills.



Generative Adversarial Networks (GAN)

Imagine that you are a counterfeiter trying to produce fake dollar bills. At your disposal, you have access to a few real dollar bills and you can use these as templates.

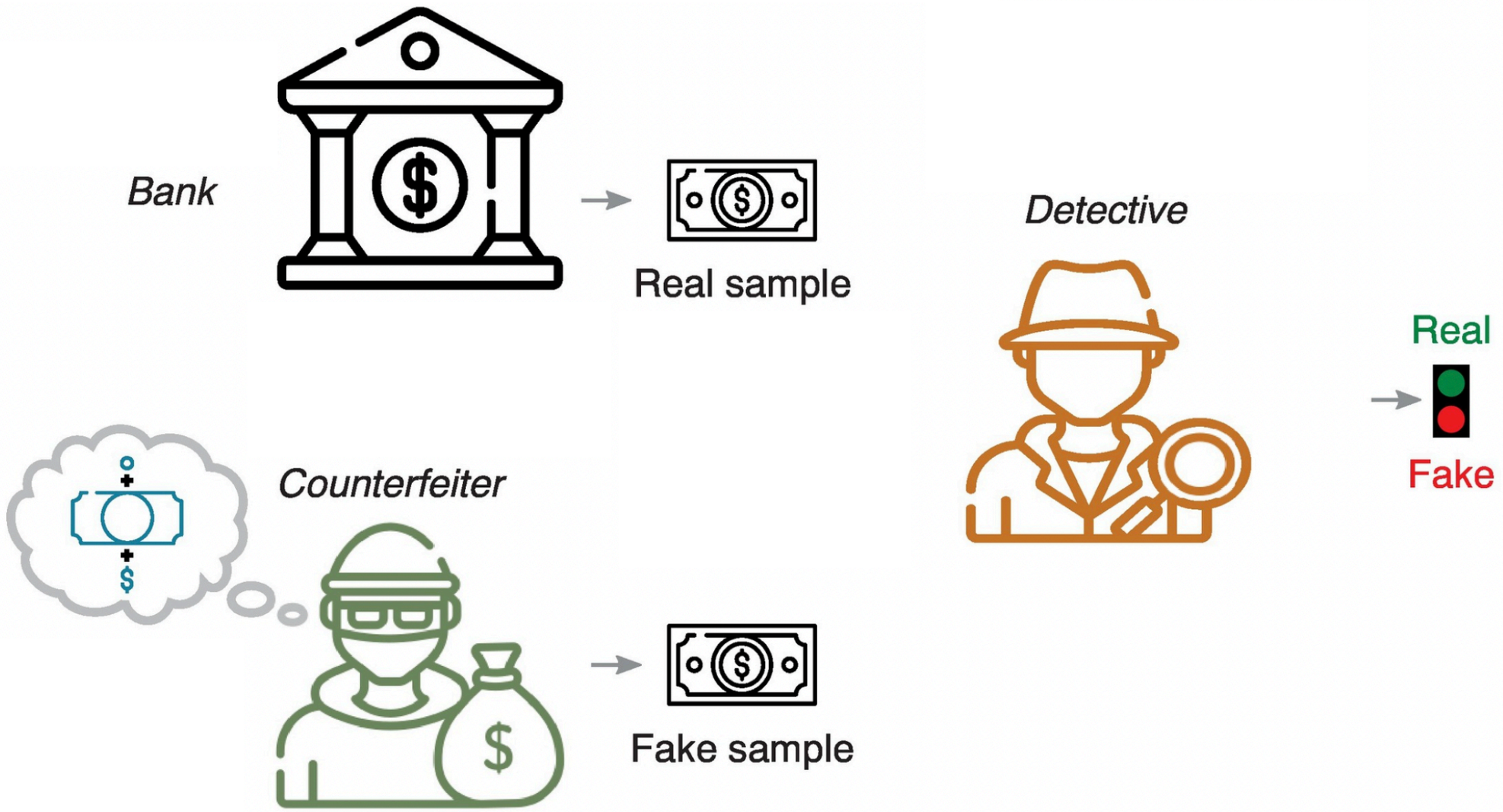


Generative Adversarial Networks (GAN)

Imagine that you are a counterfeiter trying to produce fake dollar bills. At your disposal, you have access to a few real dollar bills and you can use these as templates. However, you don't really know what makes a convincing fake bill (serial numbers, material used, security strips etc) so you eventually get caught.

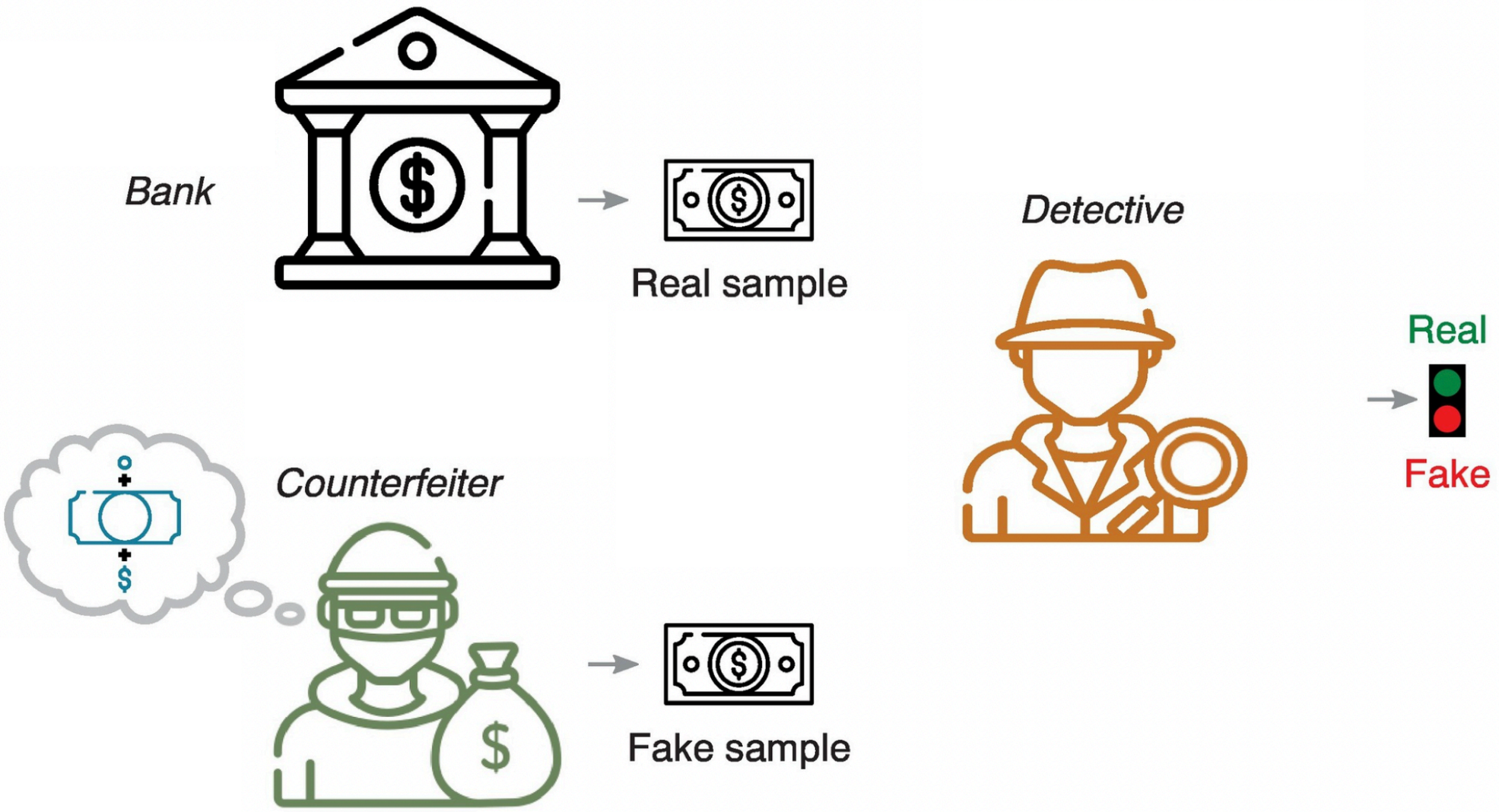
Generative Adversarial Networks (GAN)

Once you are caught because of some detail on your bills, you learn to fix that detail.



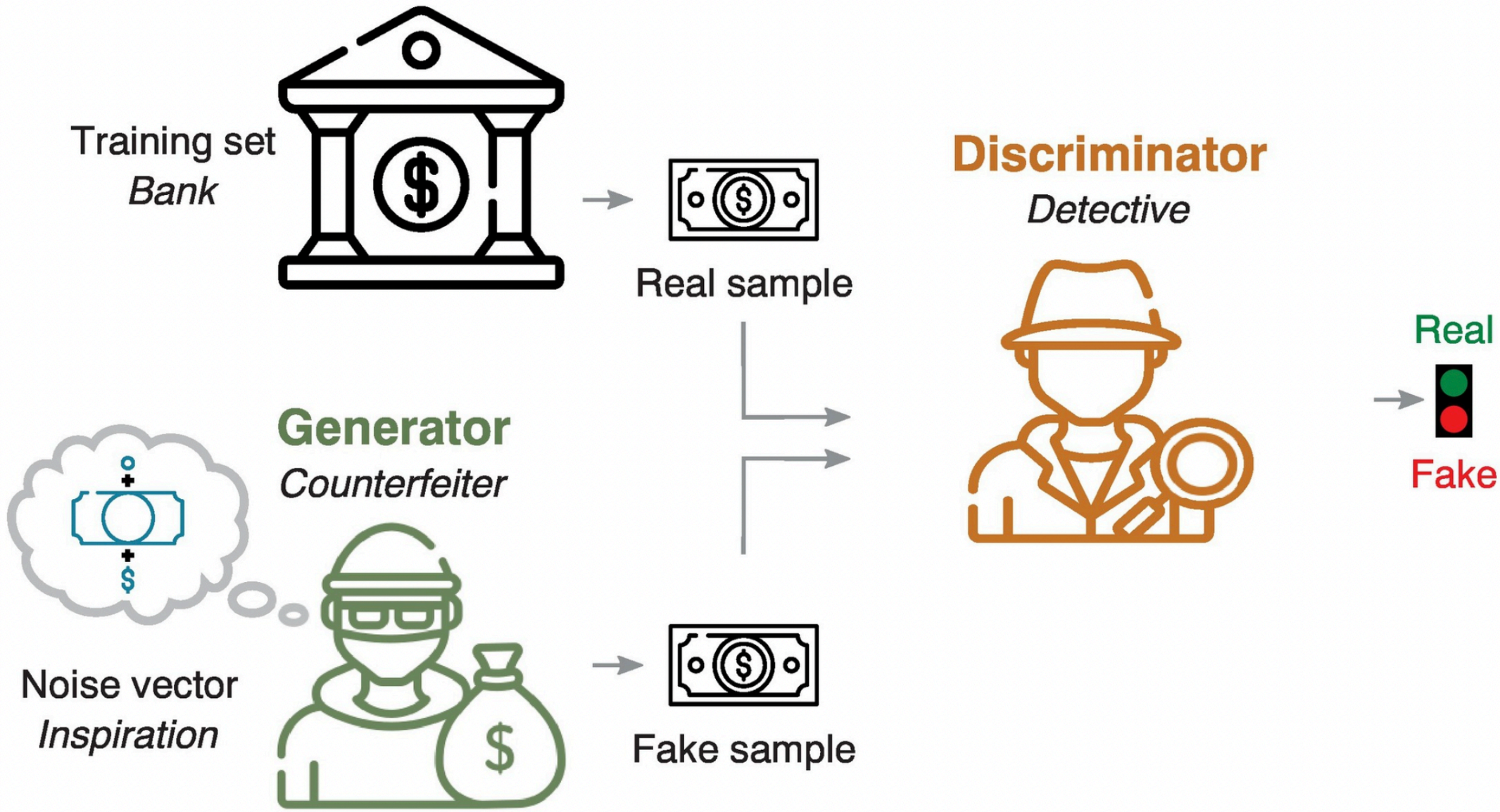
Generative Adversarial Networks (GAN)

Once you are caught because of some detail on your bills, you learn to fix that detail. Through trial and error after being caught many times, your counterfeits start improving.

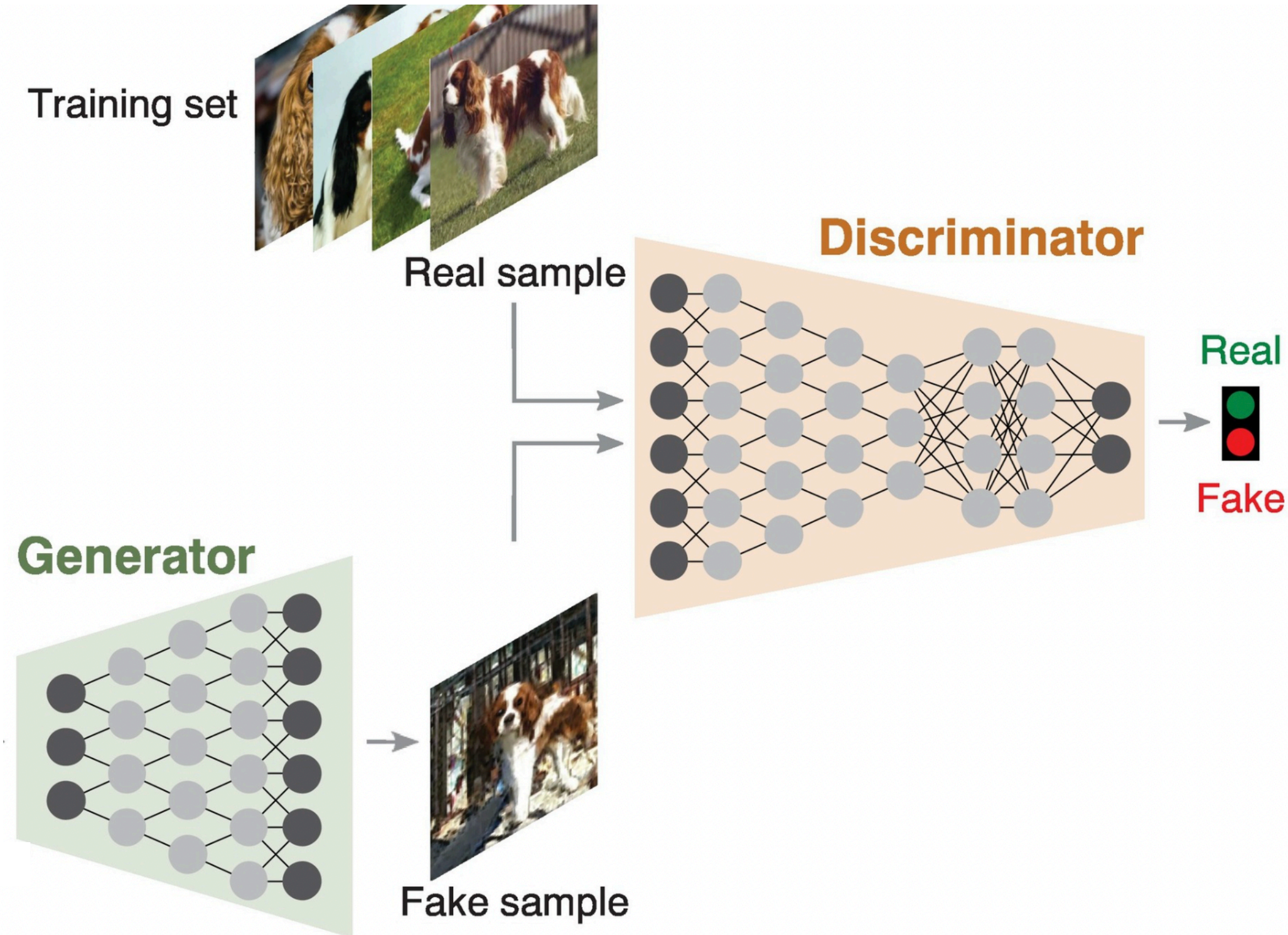


Generative Adversarial Networks (GAN)

Idea: for neural networks, train both the counterfeiter and detective to get better counterfeits!

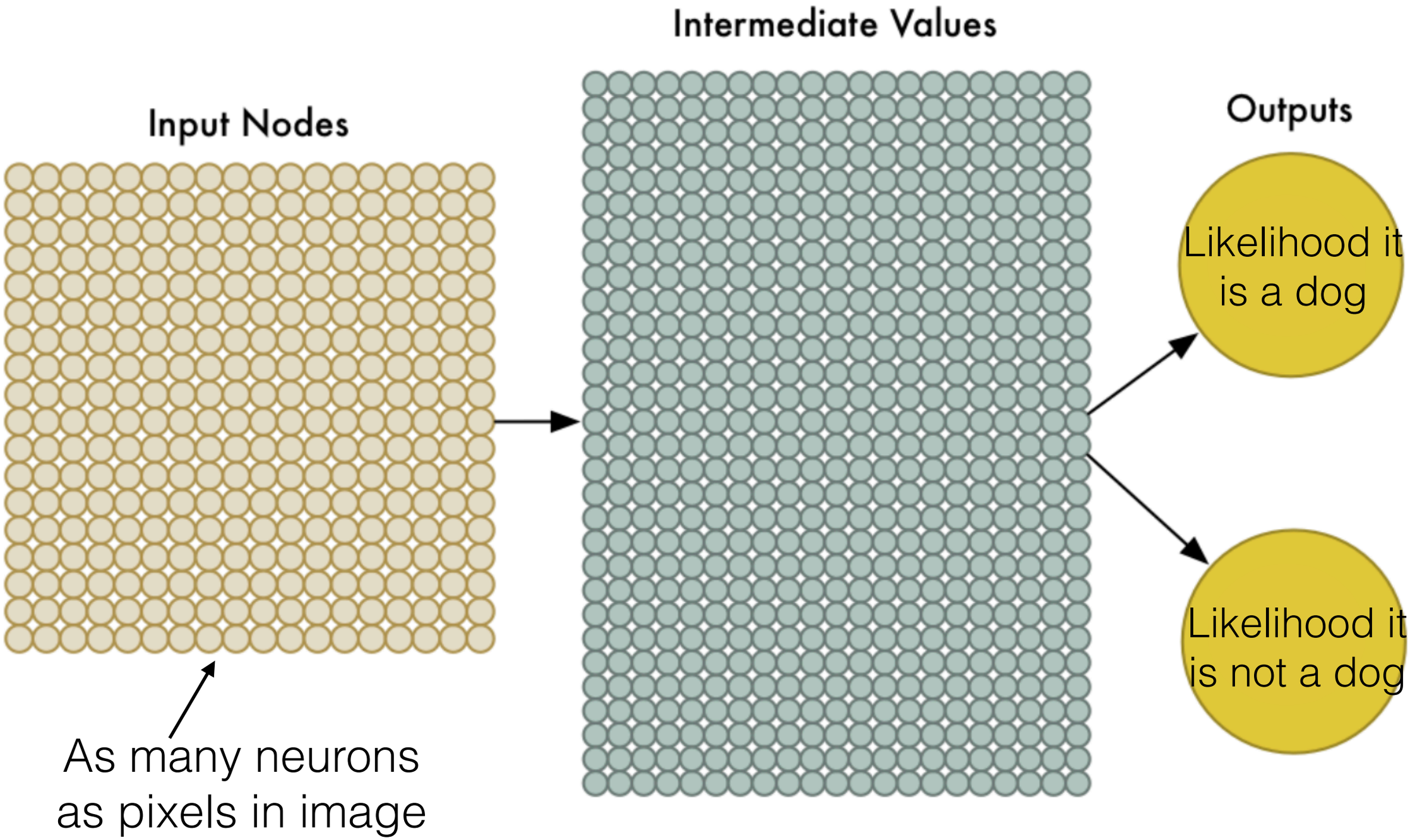


Generative Adversarial Networks (GAN)



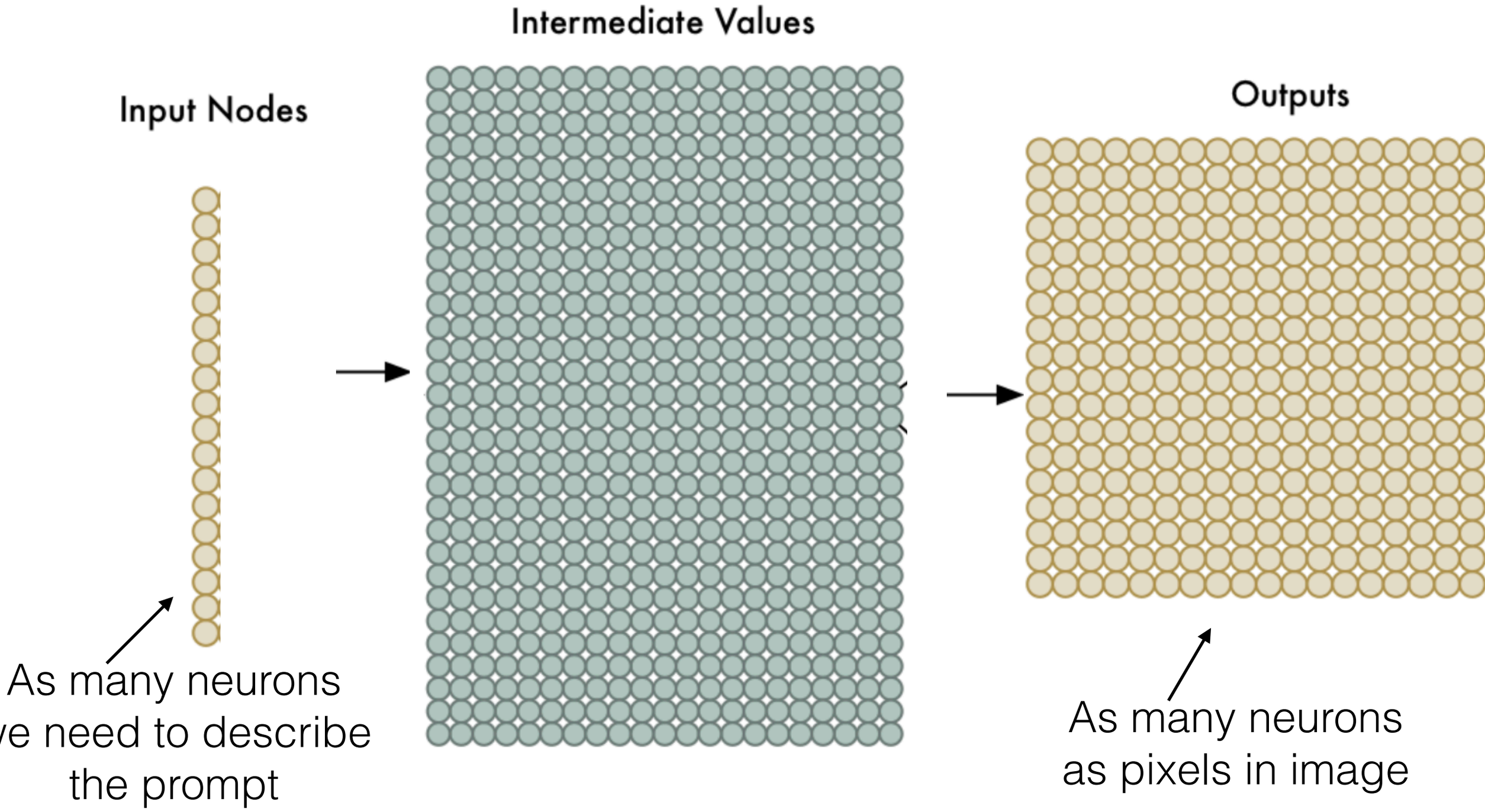
Generative Adversarial Networks (GAN)

Discriminator: a classifier (dog or not dog)



Generative Adversarial Networks (GAN)

Generator: generates a dog image from a prompt (size, breed, …)



Generative Adversarial Networks (GAN)

When training the network, we find the weights of the discriminator to catch as many fakes as possible, and the weights of the generator to create images that fool the discriminator as much as possible.

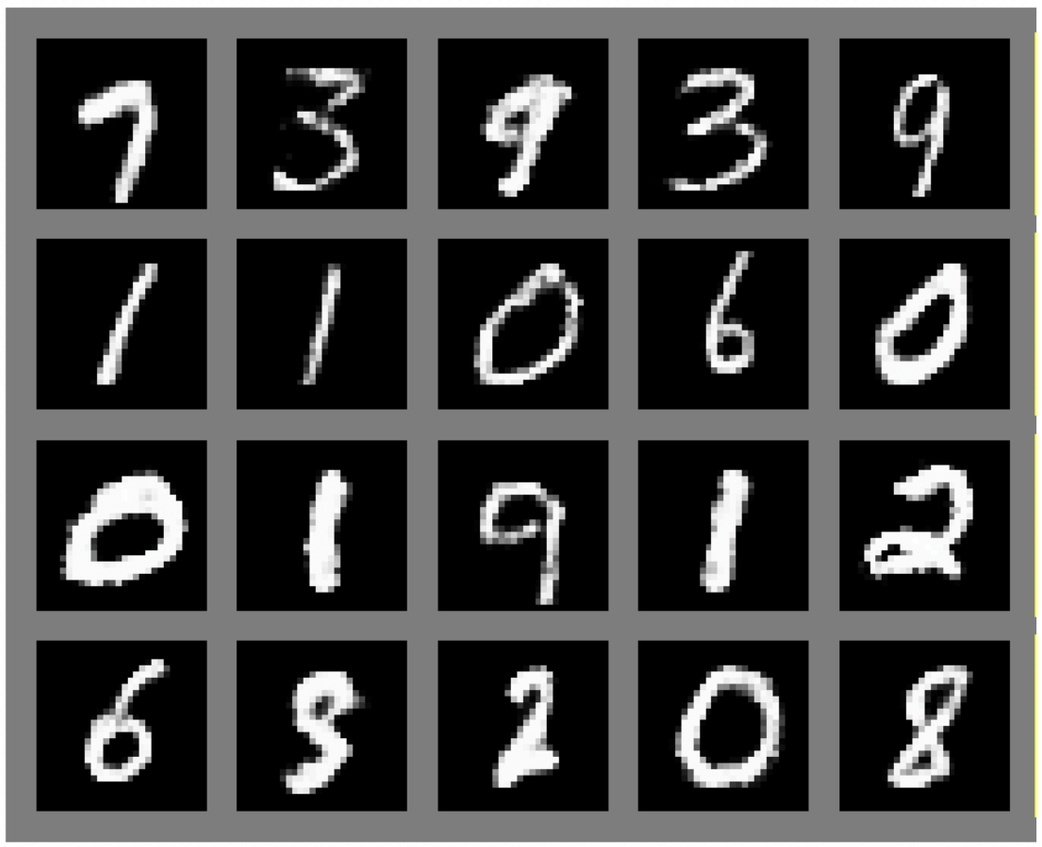
Generative Adversarial Networks (GAN)

When training the network, we find the weights of the discriminator to catch as many fakes as possible, and the weights of the generator to create images that fool the discriminator as much as possible.

If the training goes perfectly, we end up at a Nash equilibrium between the discriminator and generator and can end the training, since neither network then wants to change their weights.

2014

It worked well for some data…

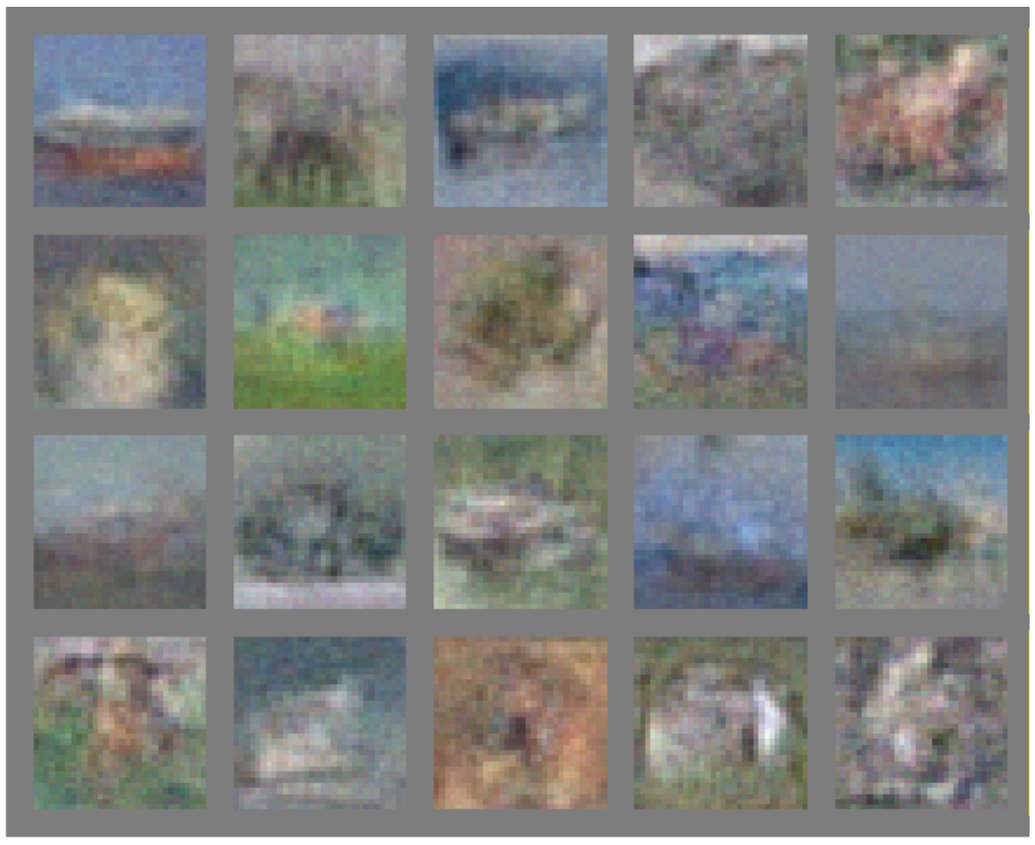


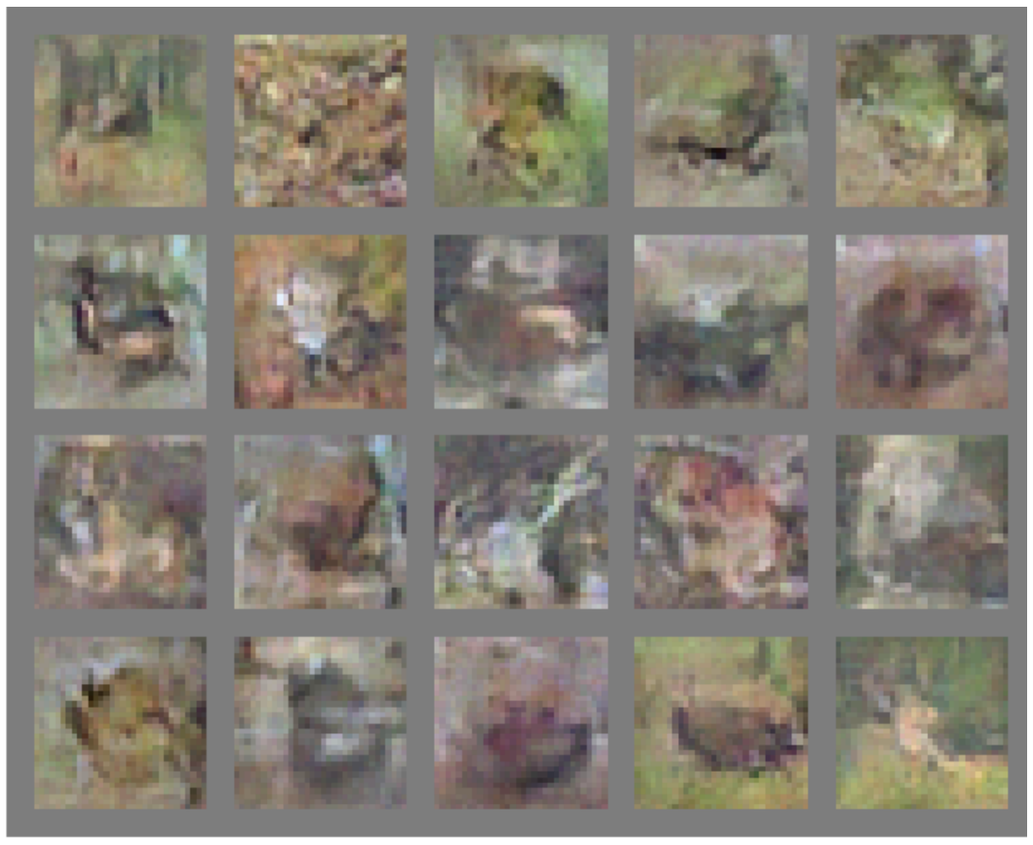
Generate handwritten digits

Generate faces

2014

… but not as well for other data





Generate images of animals and vehicles

Style transfer

We can use a similar idea to create a network that performs style transfer.

Example: transform a photo into a painting in the style of Monet

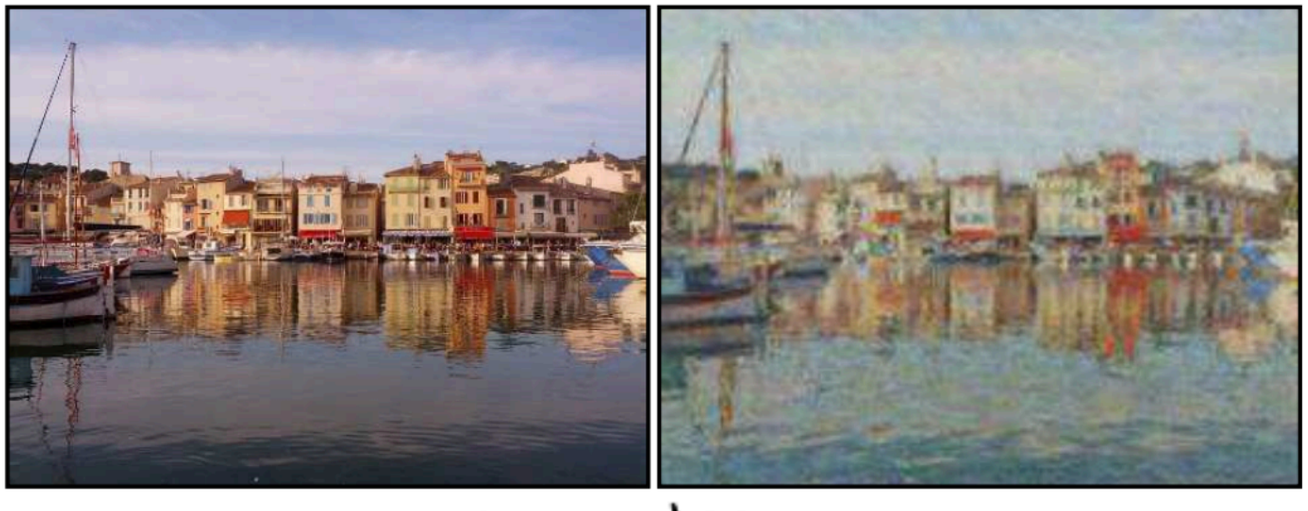


photo → Monet

Style transfer

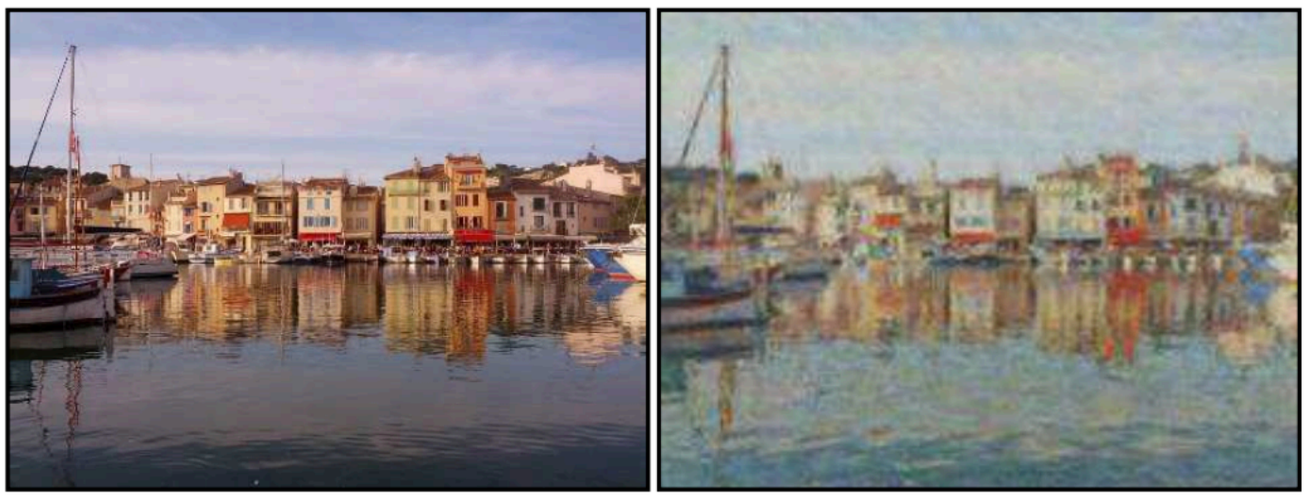
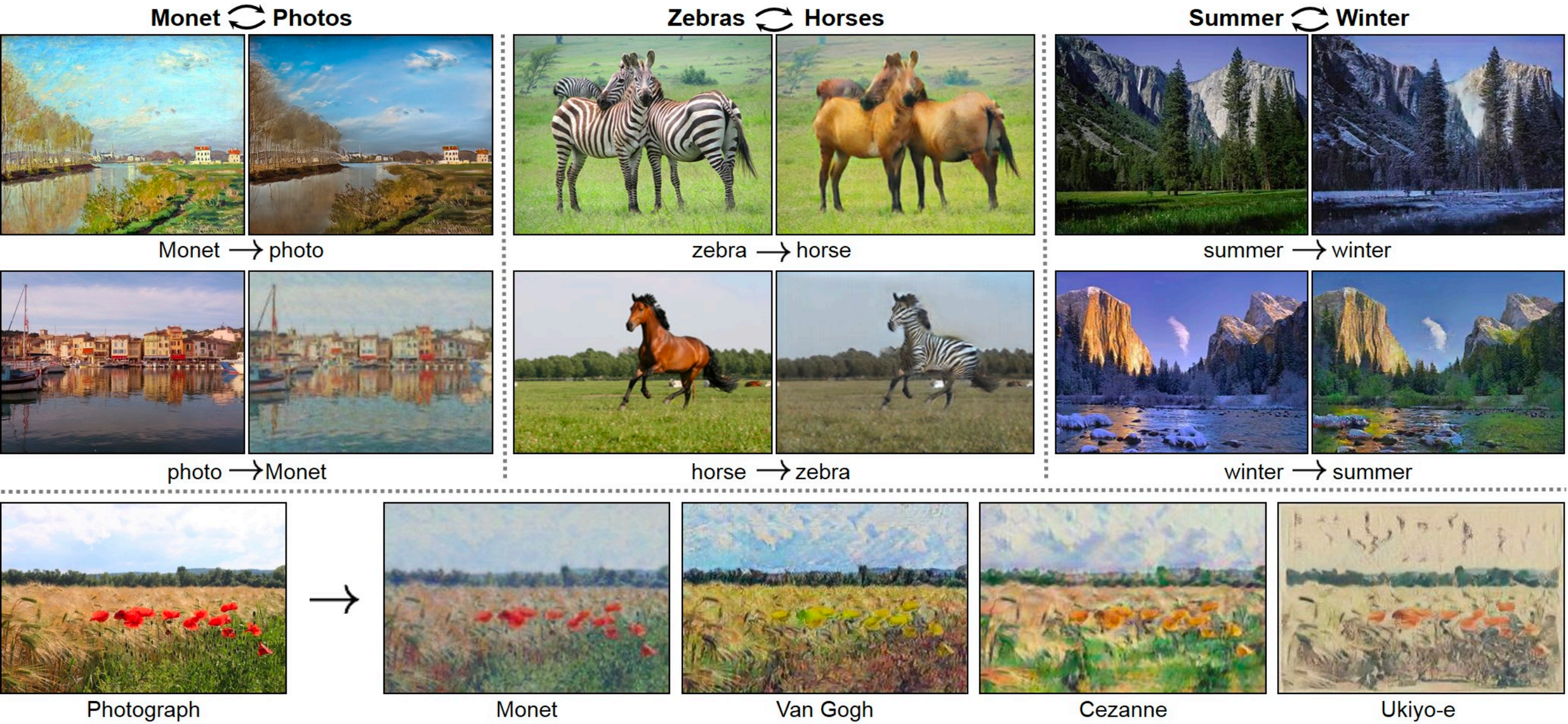


photo → Monet

We use a discriminator that looks at an image and outputs the probability that it was painted in Monet's style.
We use a generator that takes a photo as input and outputs a version changed to look more like Monet.

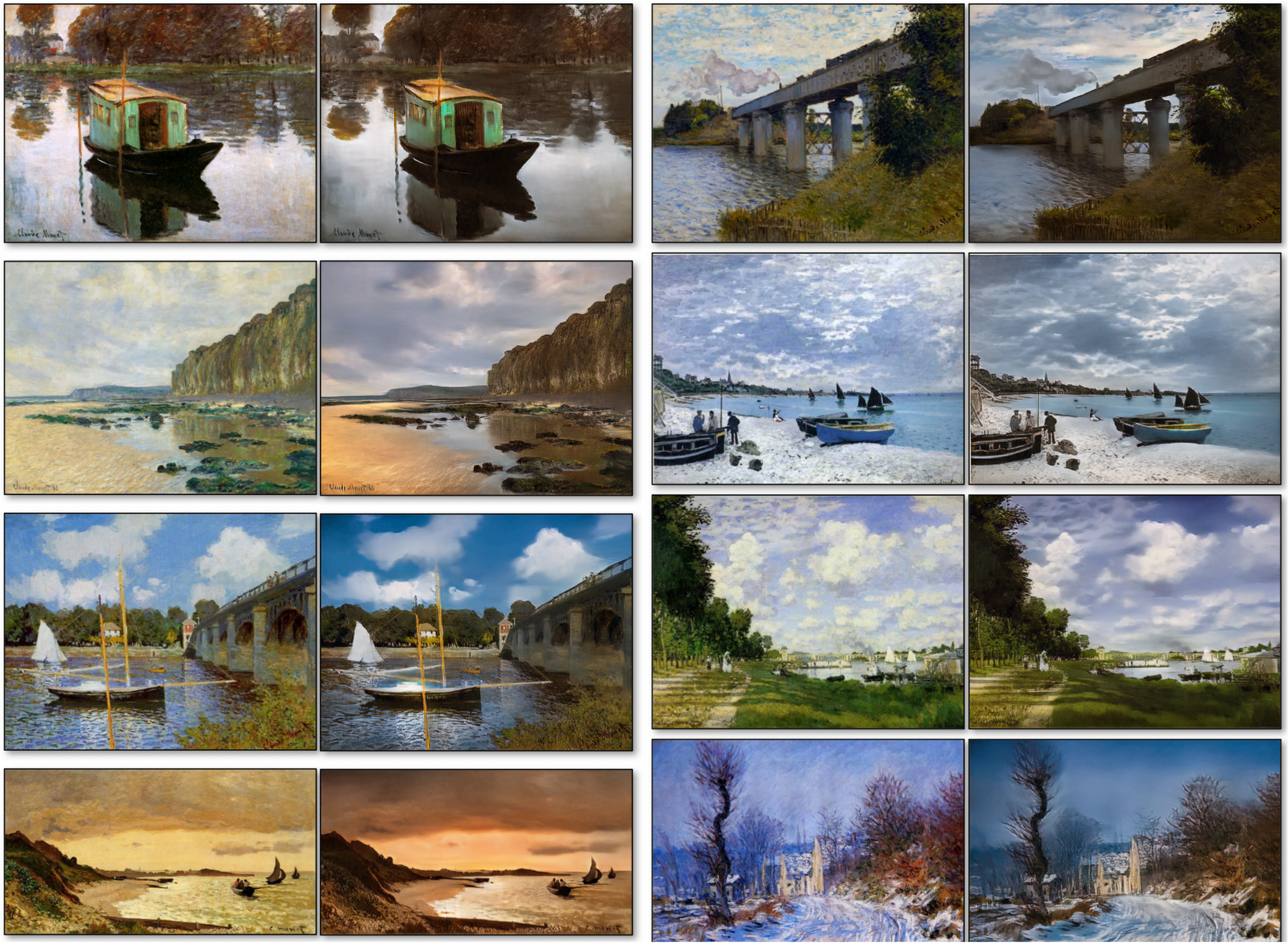
Style transfer

CycleGAN (2017)



Input | Monet | Van Gogh | Cezanne | Ukiyo-e

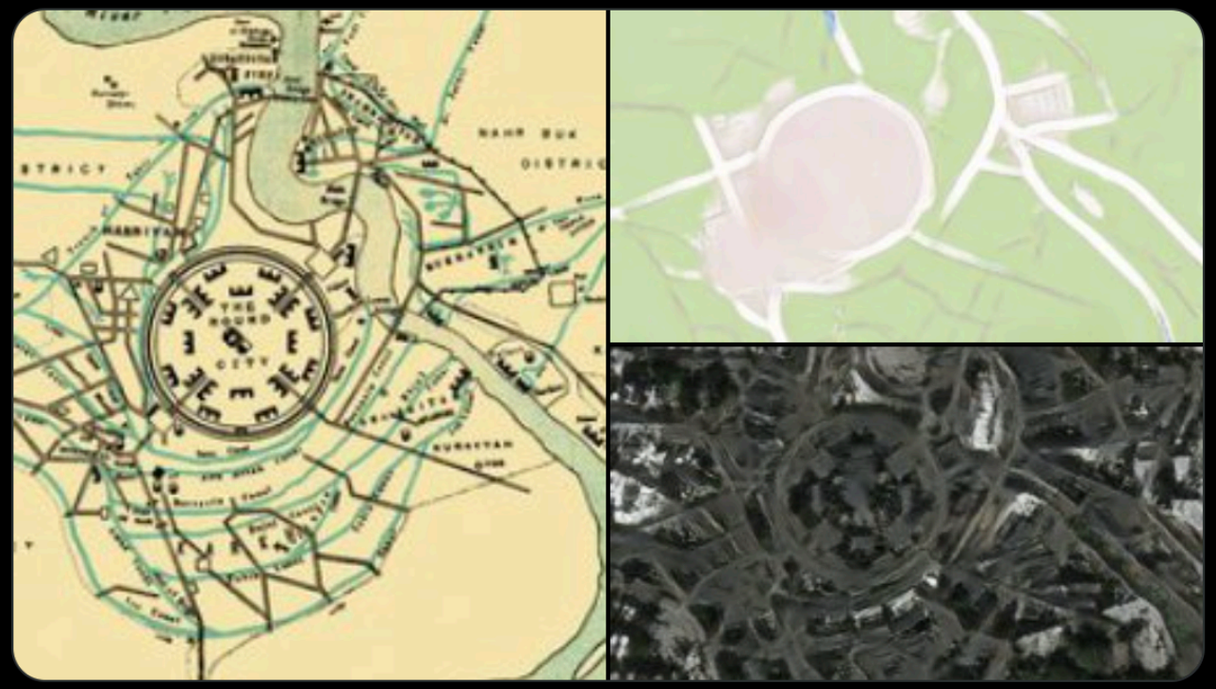
Input Output Input Output



https://junyanz.github.io/CycleGAN/

Jack Clark
@jackclarkSF

An ancient map of Babylon, transmuted via CycleGAN into a modern Google Map & Satellite View

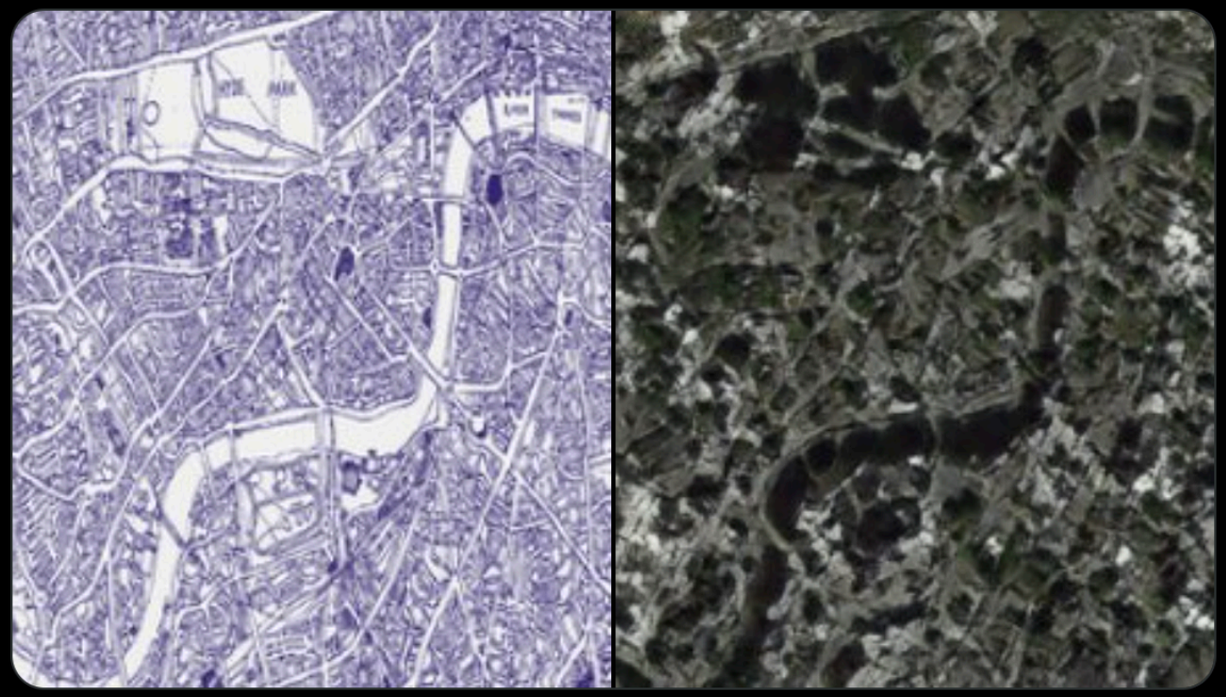


1:22 AM · Jun 1, 2017

110 Retweets **10** Quotes **235** Likes **1** Bookmark

Jack Clark
@jackclarkSF

Having fun with CycleGAN - using it to reconstruct ancient maps as modern satellite photos.

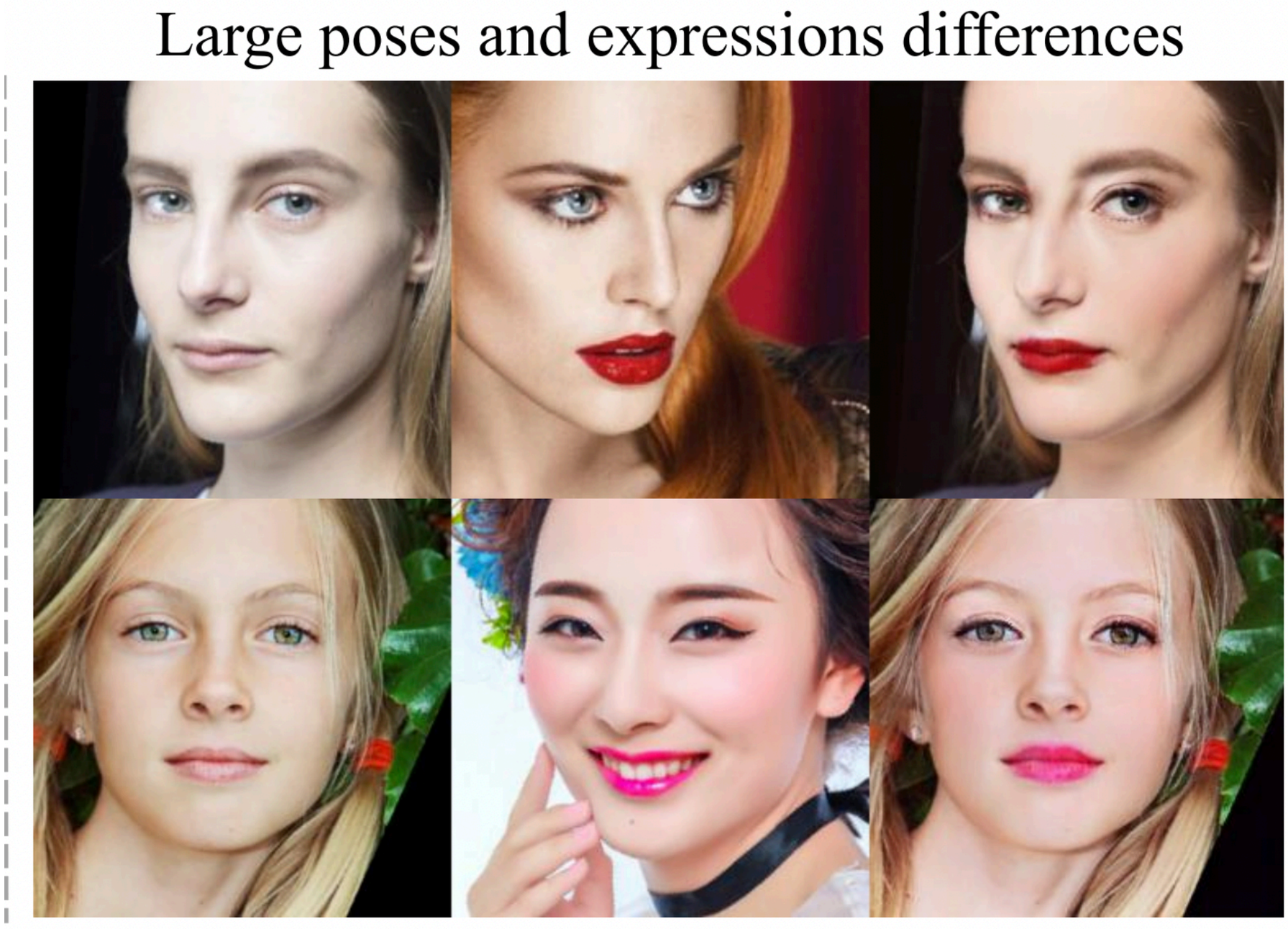


11:12 PM · May 31, 2017

46 Retweets **138** Likes

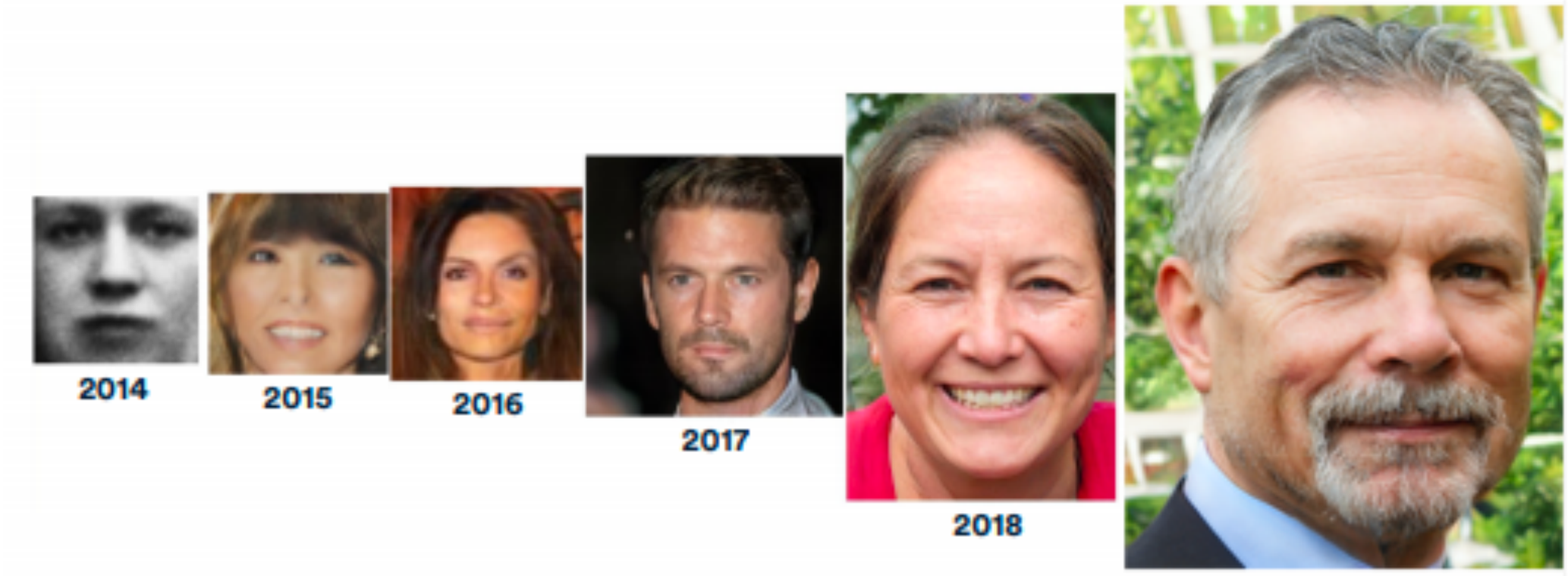
StyleGAN (2018)

PSGAN (2018)


Large poses and expressions differences
Source
Reference
Result

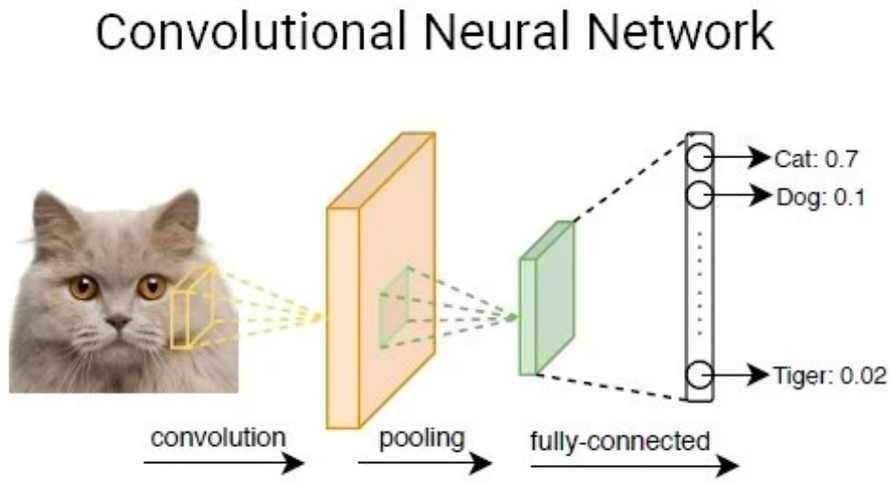
Deepfakes

Quality improvement of GANs over the last decade

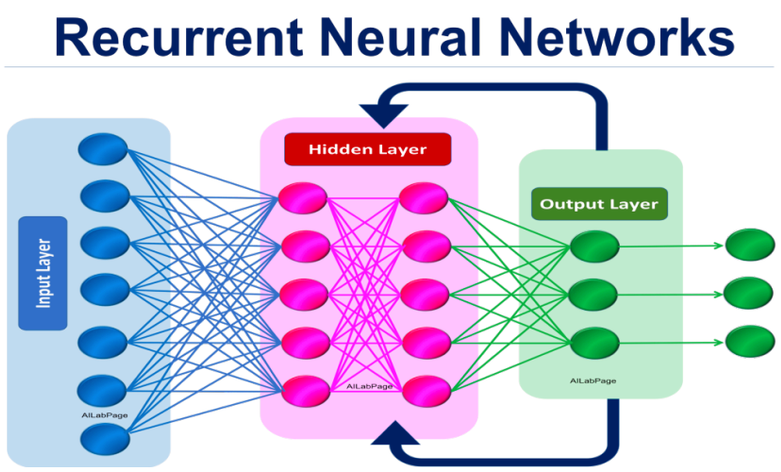


What has caused the improvement?

Image

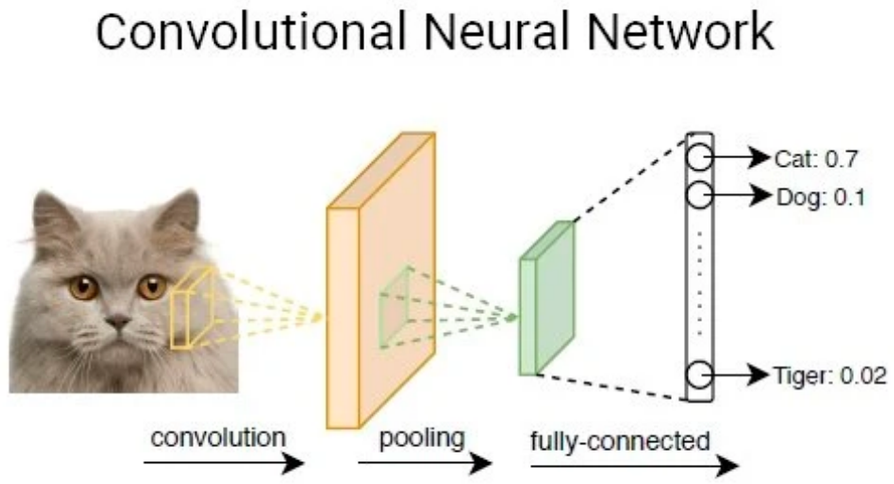


Language

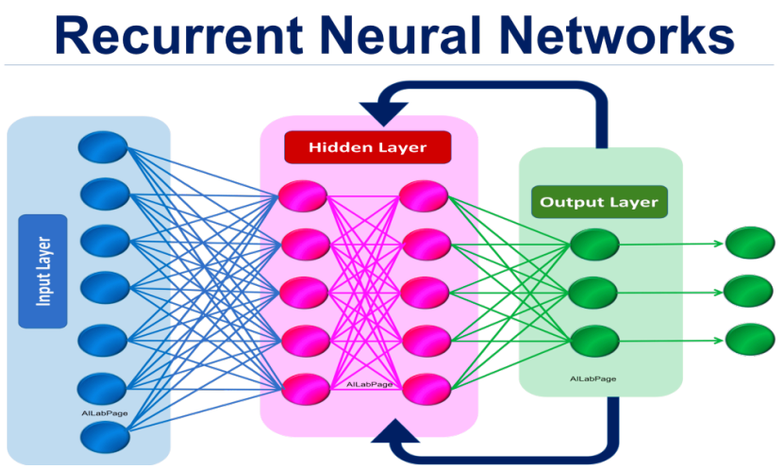


What has caused the improvement?

Image

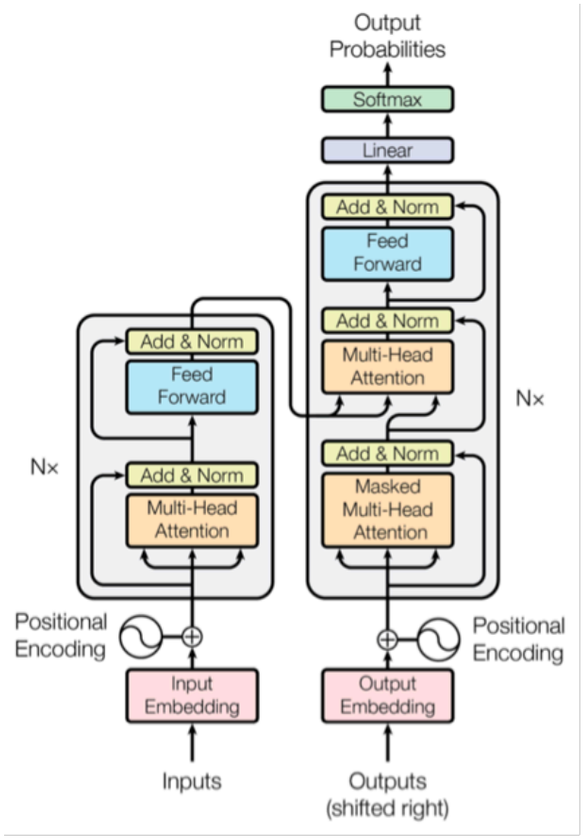


Language



2017

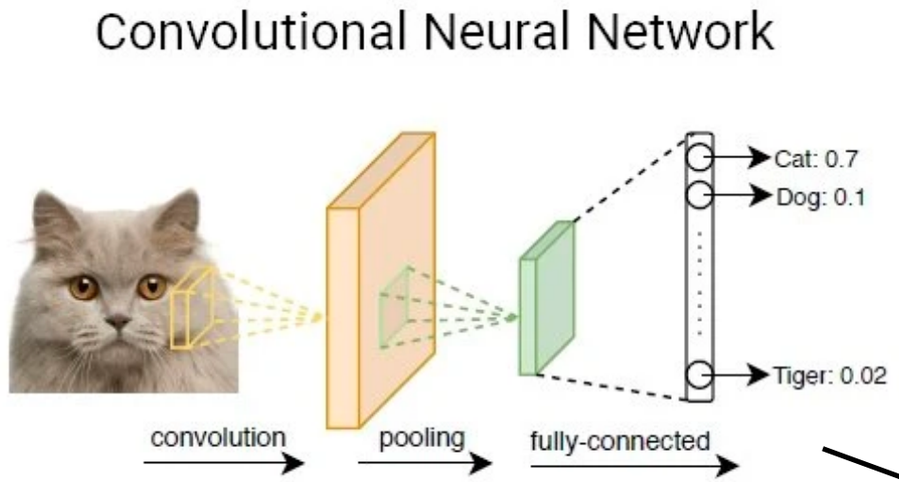
Transformer



Just like recurrent neural networks were replaced by transformers for language and translation…

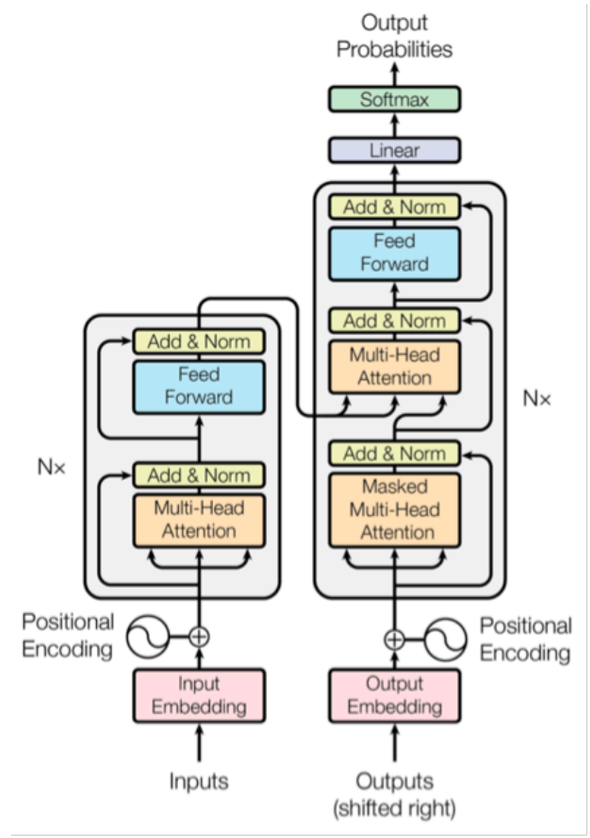
What has caused the improvement?

Image

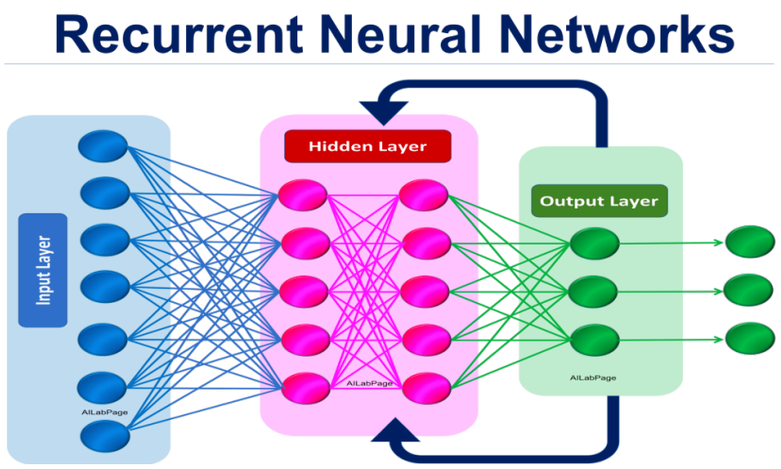


2020

Transformer



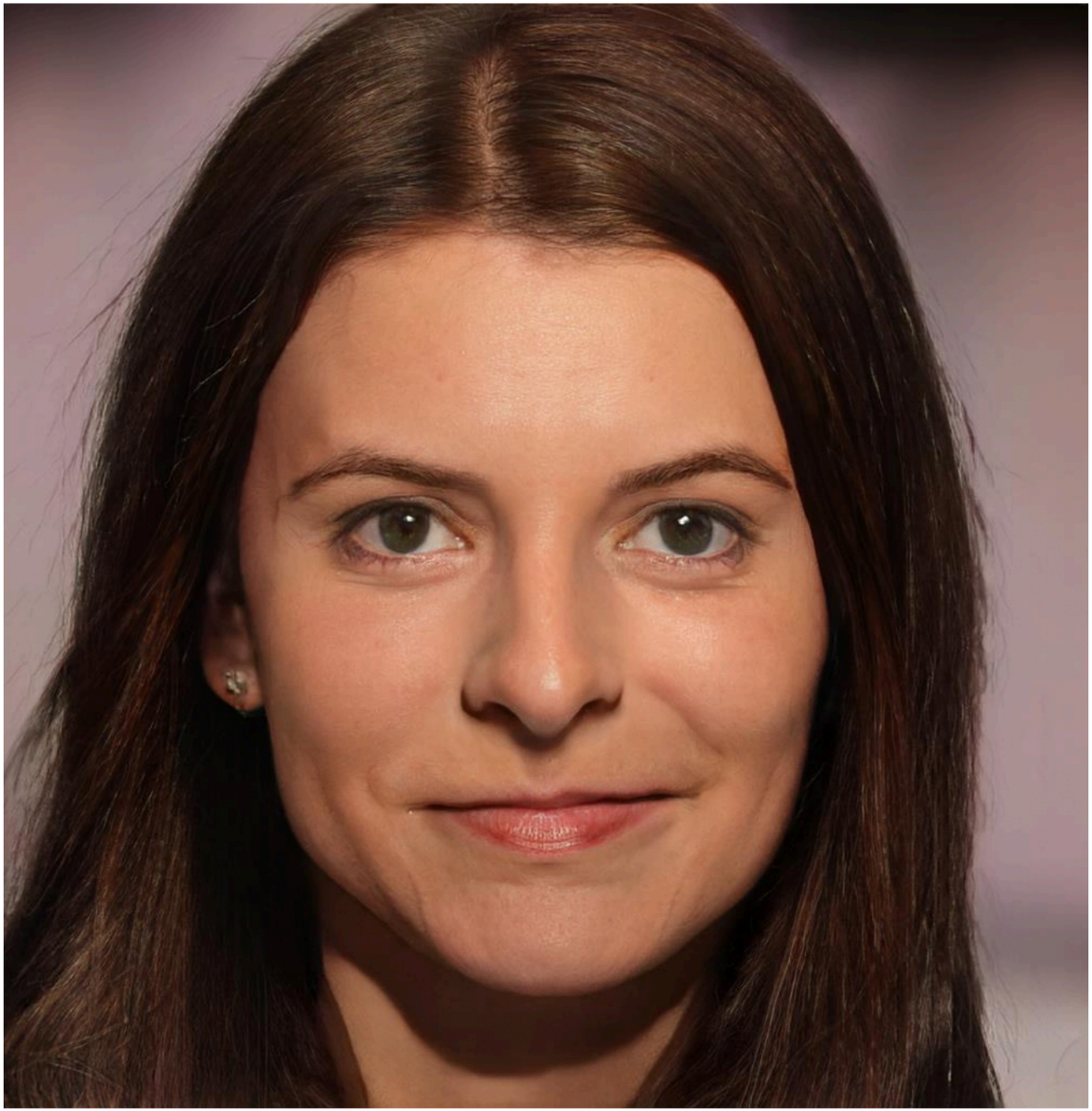
Language

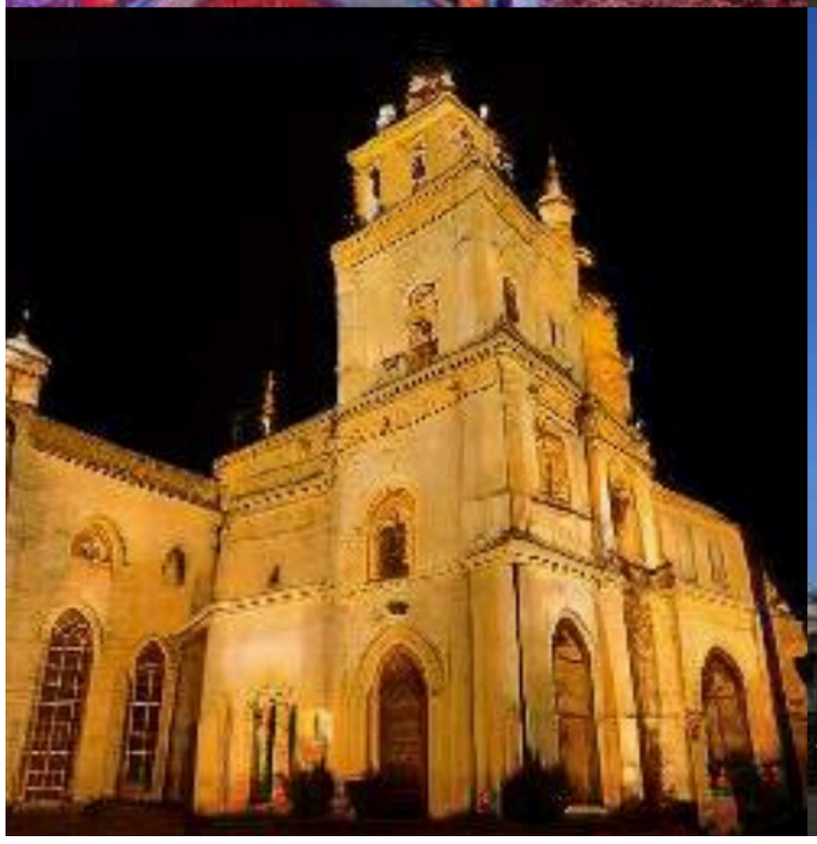


2017

… convolutional neural networks were replaced by transformers for images

2021





Generating images from prompts

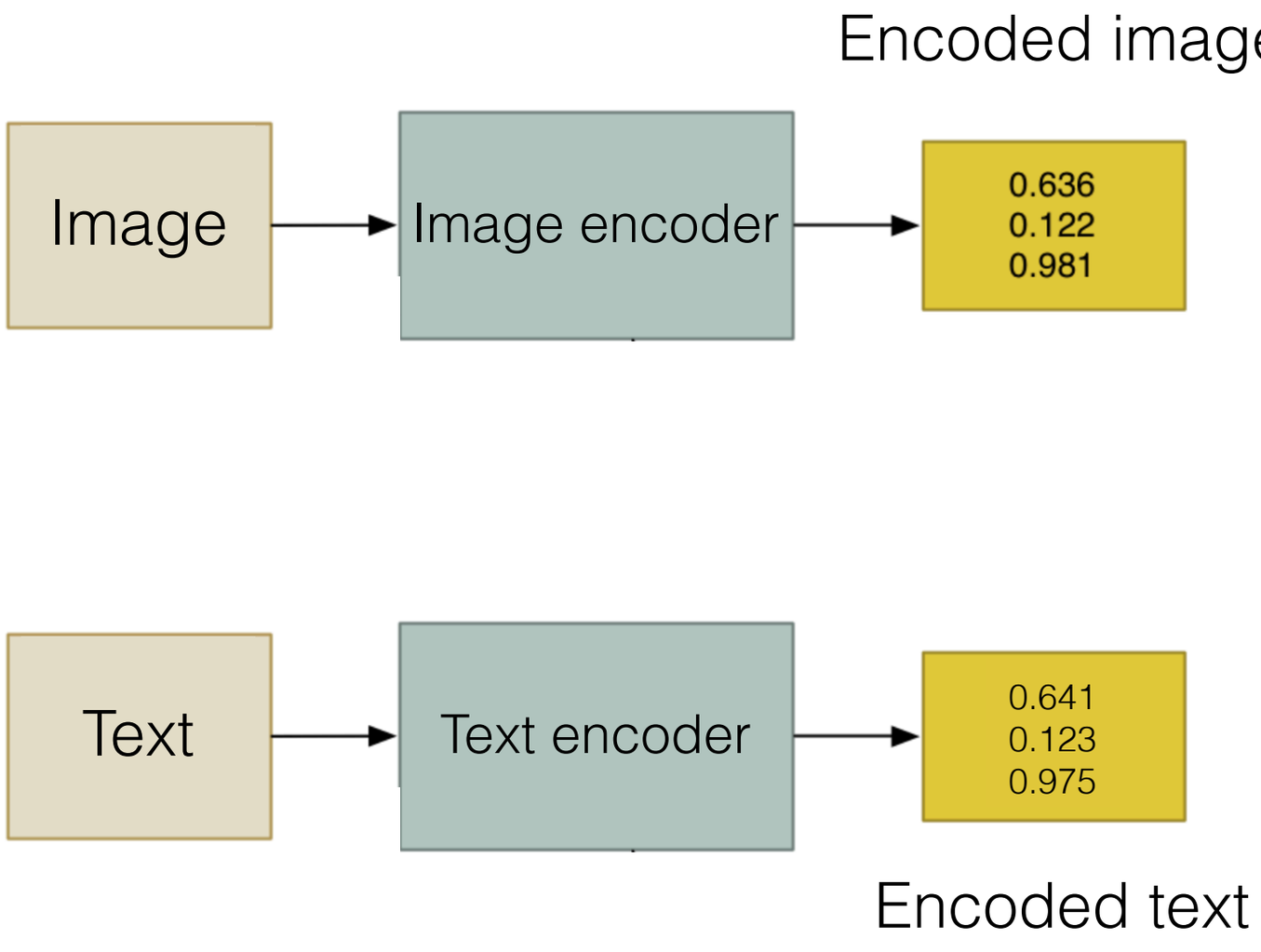
How do DALL-E and similar tools generate images from prompts written in English?

DALL·E 1

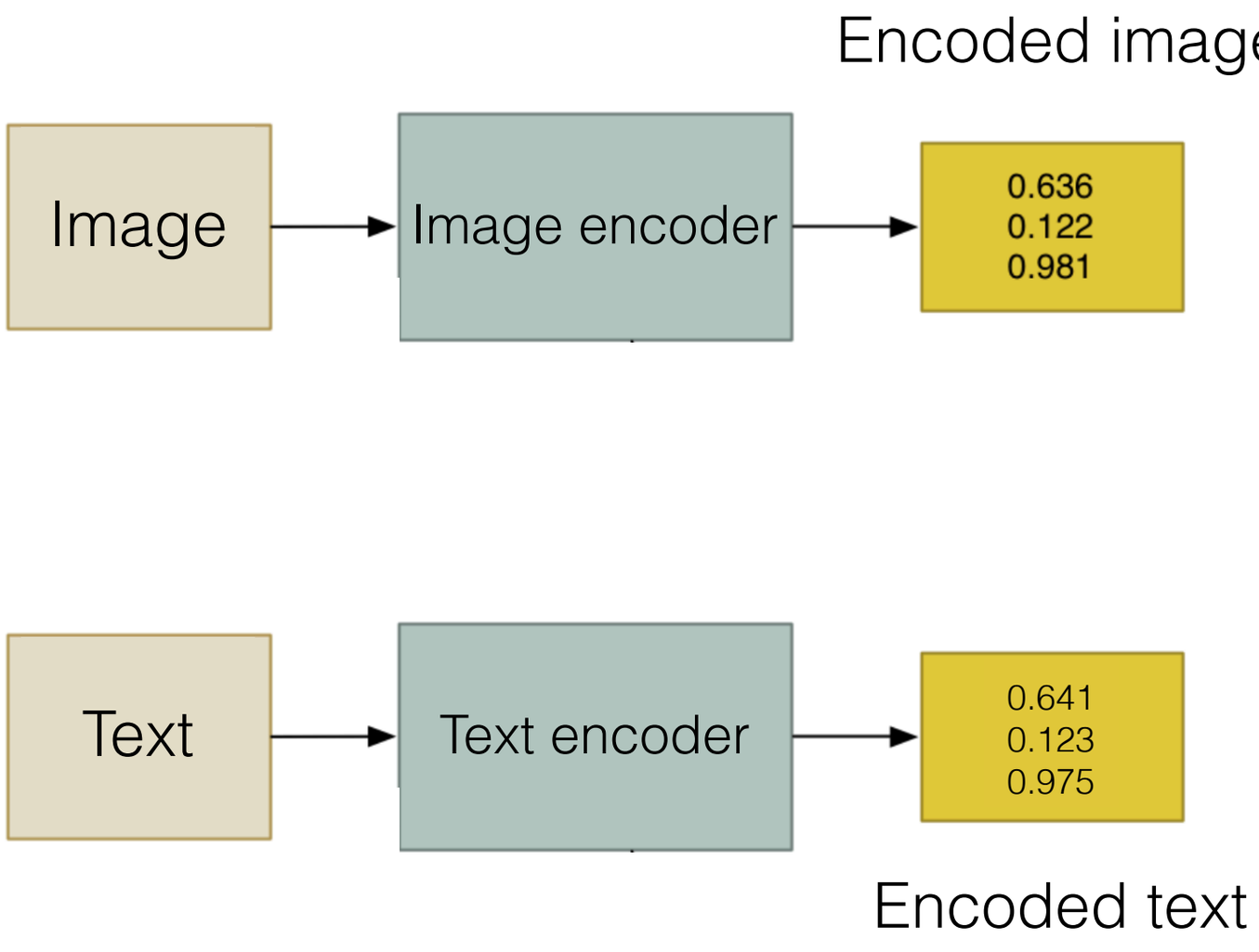
DALL·E 2

"a painting of a fox sitting in a field at sunrise in the style of Claude Monet"

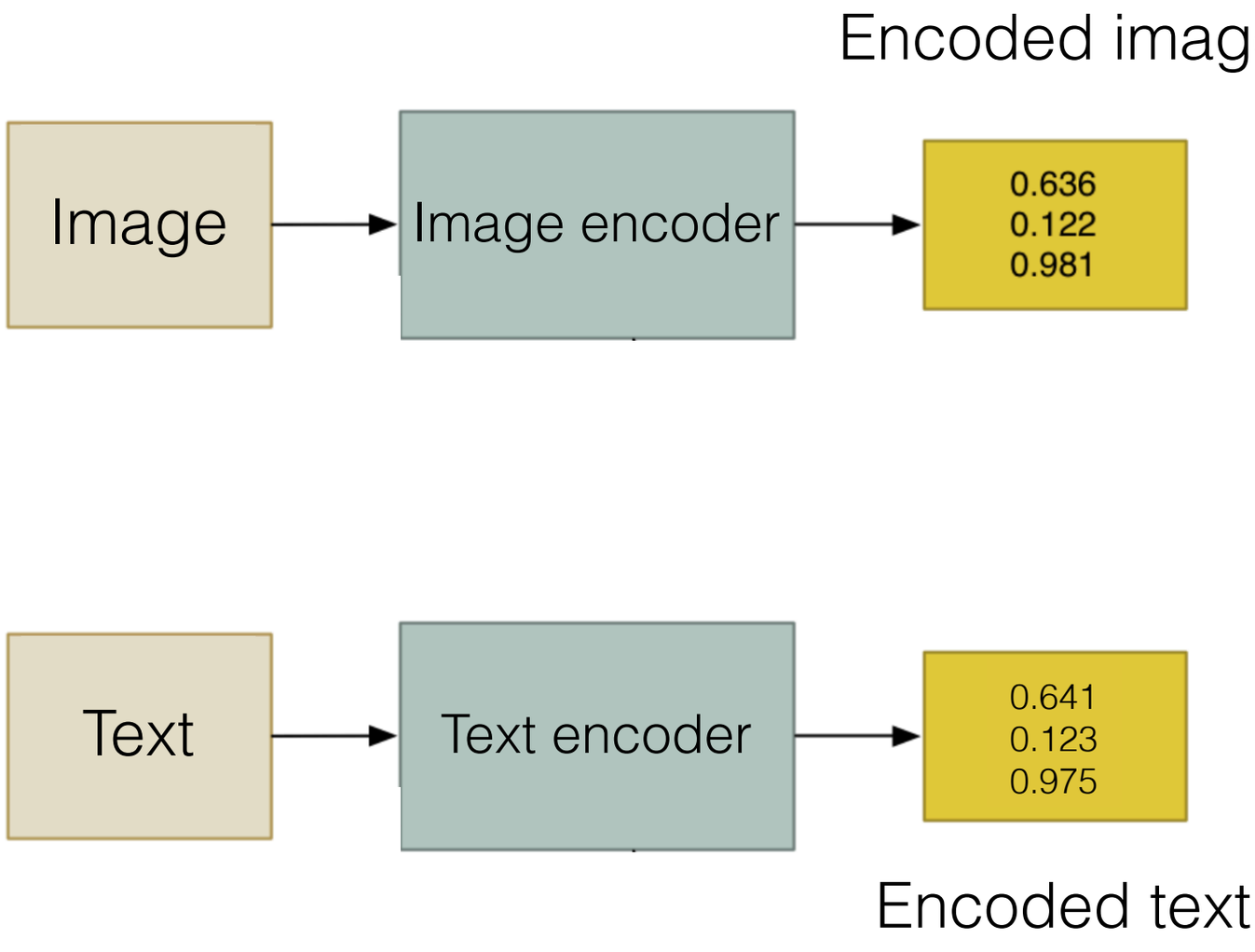
For training, we use a large number of images with text descriptions. We train a text encoder and an image encoder simultaneously, to get encodings that are similar for both the text and images.



When we get a new text prompt, we can compute its encoding with the text encoder, and decode the result with the image decoder, to get an image that closely resembles the text prompt.



Vice versa to get a text description from an image.



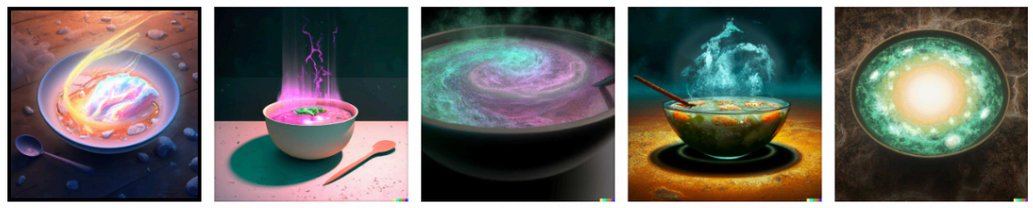
TEXT DESCRIPTION

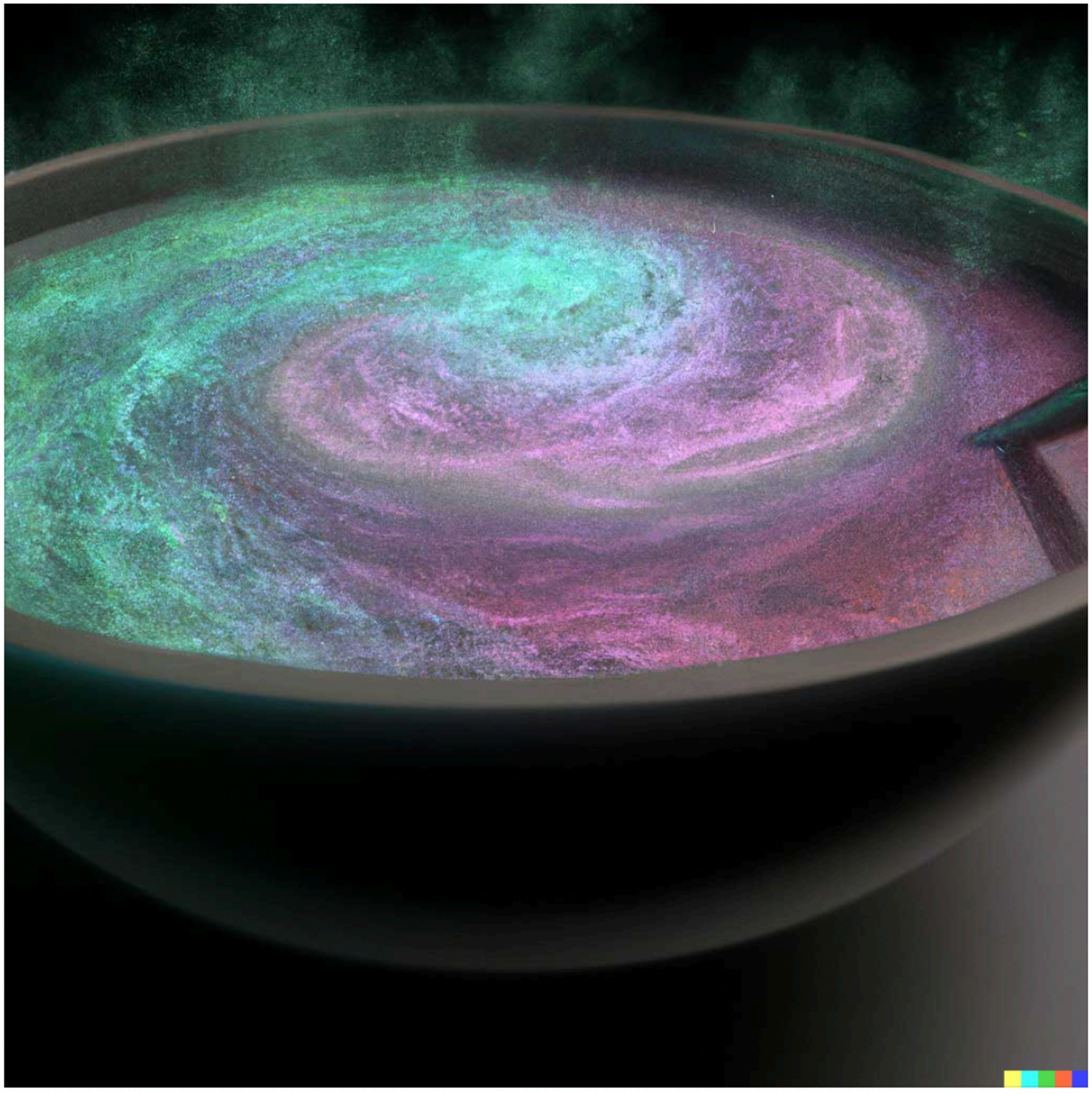
An astronaut Teddy bears A bowl of soup

that is a portal to another dimension that looks like a monster as a planet in the universe

as digital art in the style of Basquiat drawn on a cave wall

DALL·E 2





TEXT DESCRIPTION

An astronaut Teddy bears A bowl of soup

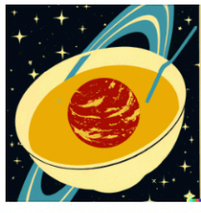
that is a portal to another dimension that looks like a monster as a planet in the universe

as a 1960s poster as mixed media with needlework as digital art

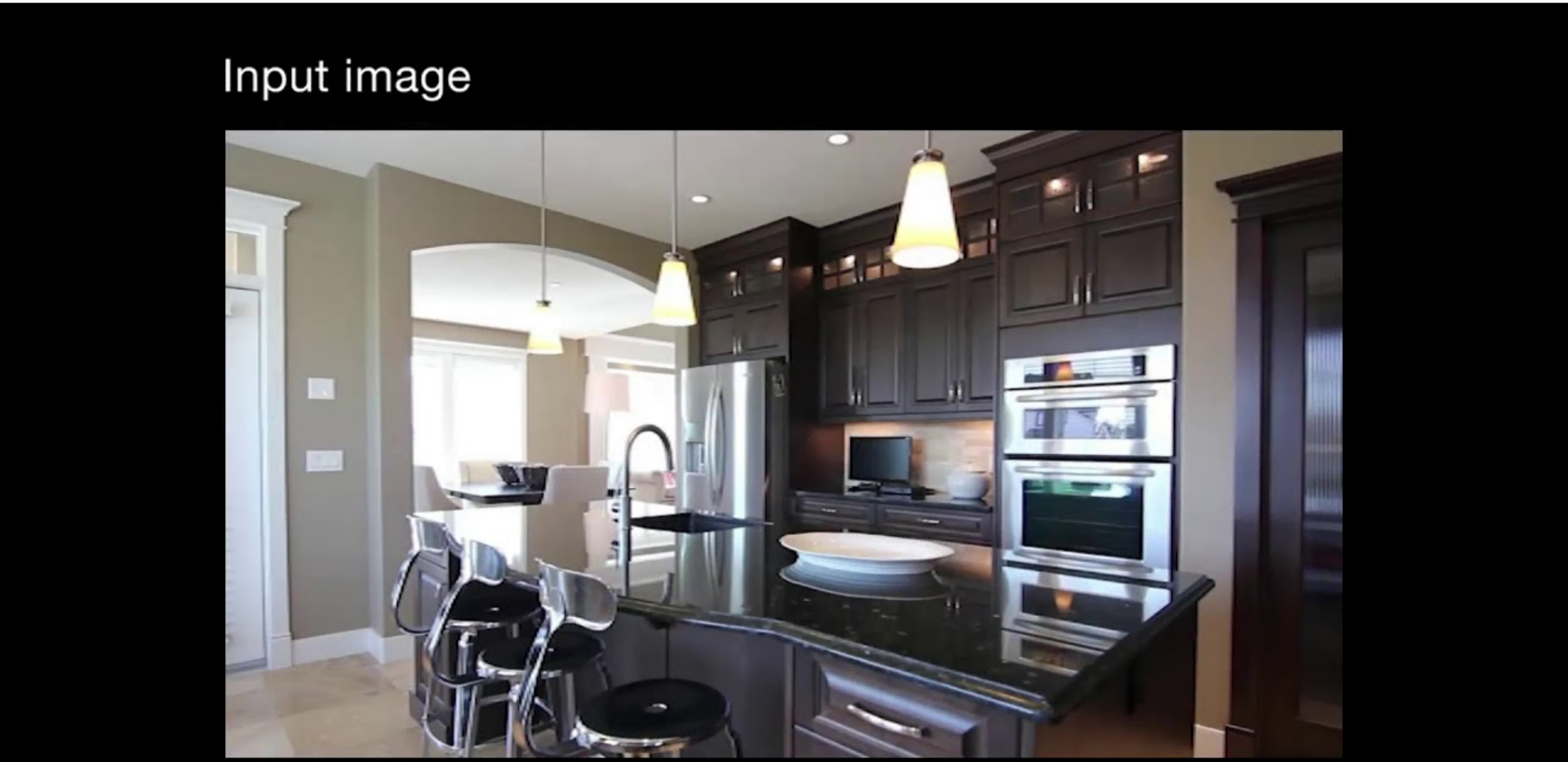
DALL·E 2



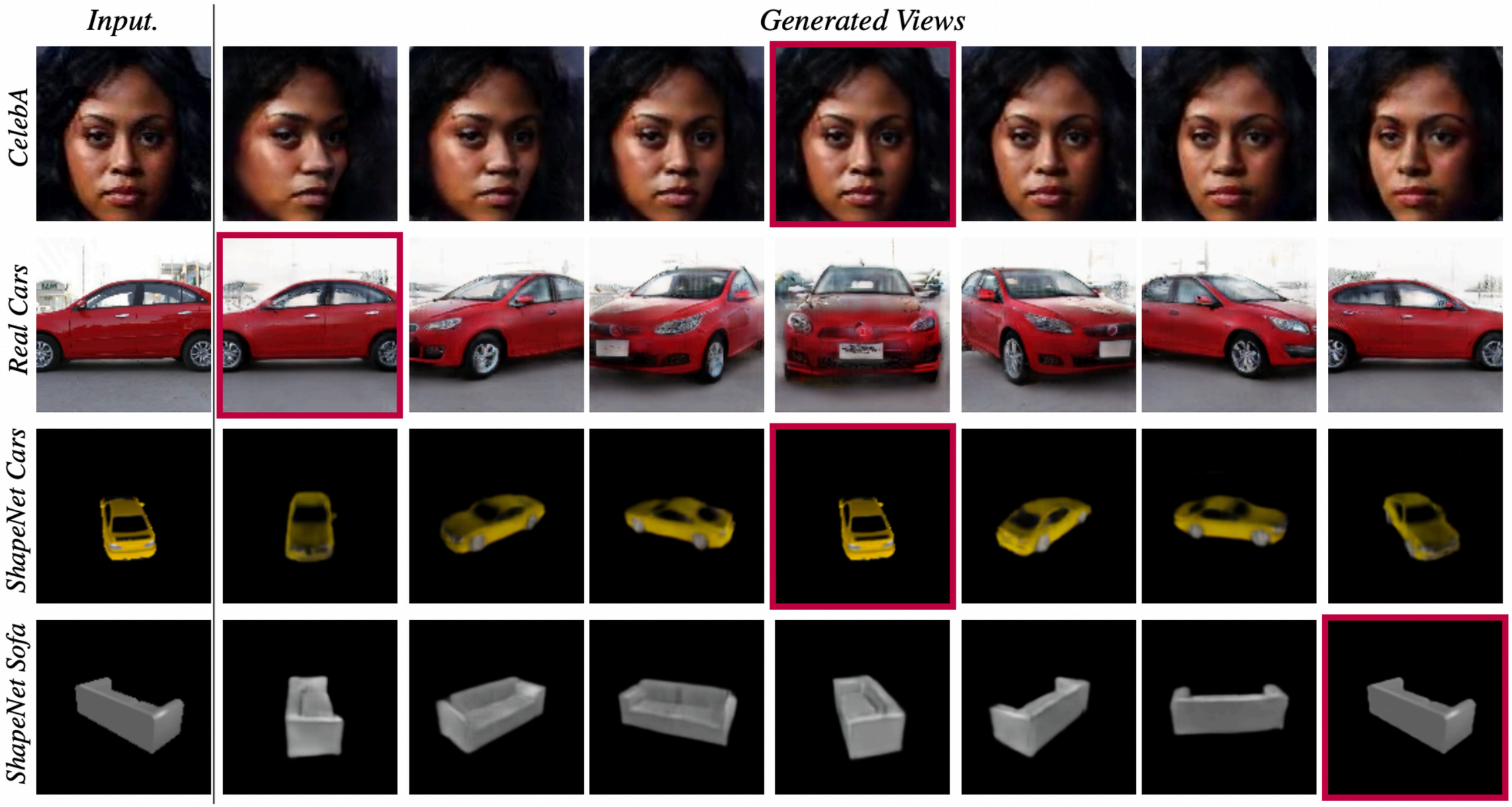




Single-View View Synthesis With Multiplane Images (2020)



Unsupervised Novel View Synthesis from a Single Image (2021)



Music instead of images

Neural network trained to compose like Mozart, or to change a classical piece by Chopin into the style of Bon Jovi.

https://openai.com/research/musenet

Neural nets as security guards

Imagine that we run an auction website like Ebay. On our website, we want to prevent people from selling prohibited items — things like live animals. We can use deep learning to automatically check auction photos for prohibited items and flag the ones that violate the rules.

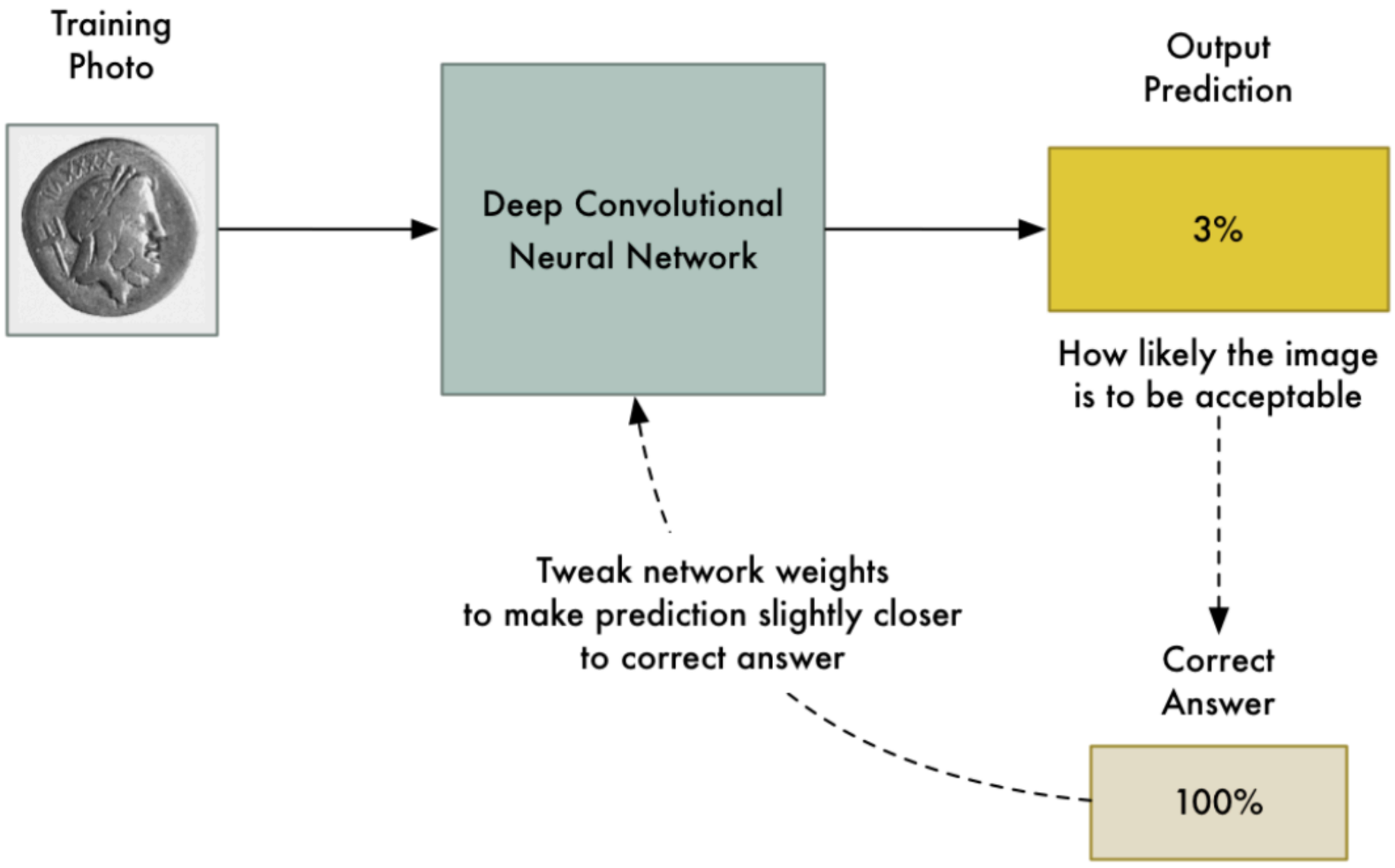
This is a typical image classification problem. To build this, we'll train a deep convolutional neural network to tell prohibited items apart from allowed items and then we'll run all the photos on our site through it.

First, we need a dataset of thousands of images from past auction listings. We need images of both allowed and prohibited items so that we can train the neural network to tell them apart.

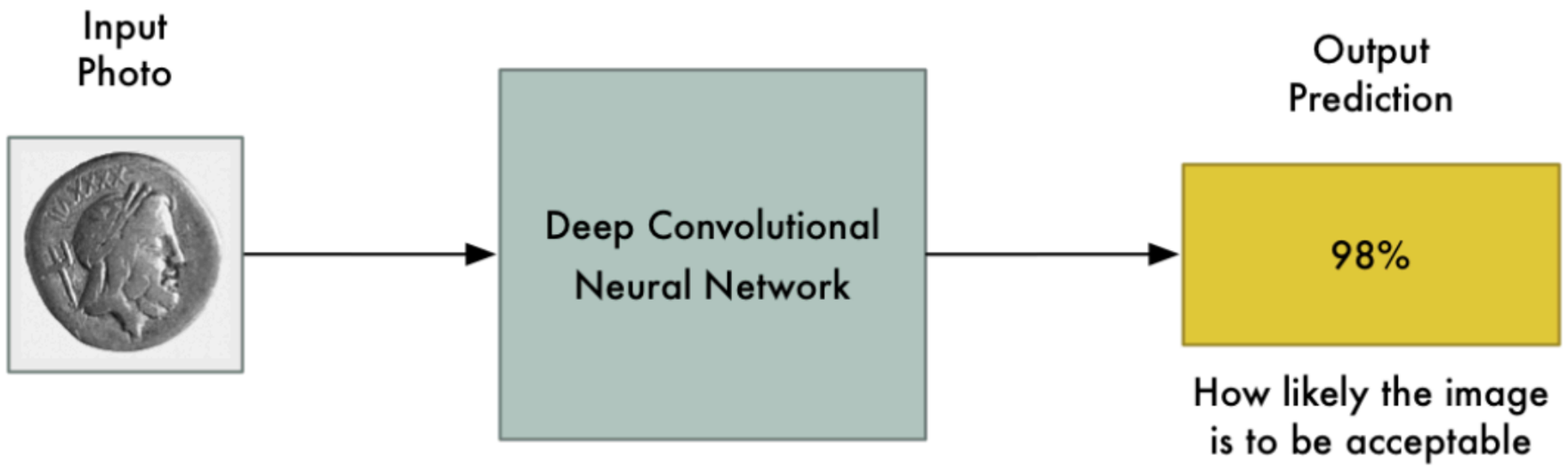
Photos of
Allowed Items

Photos of
Prohibited Items

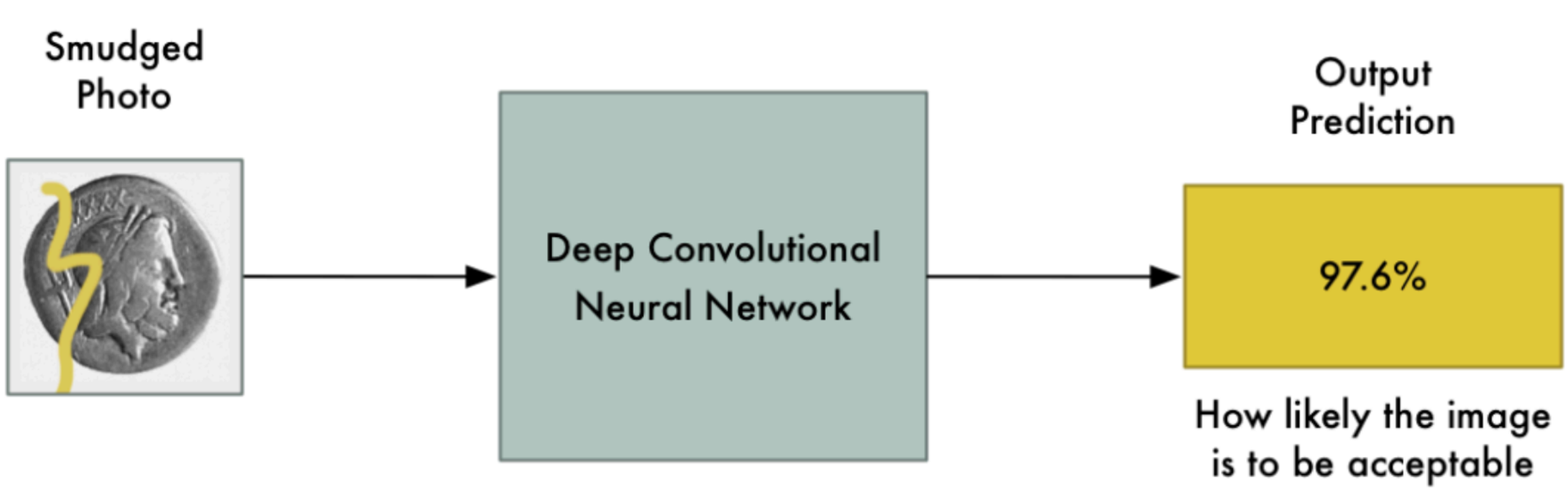
Training the Neural Network



We repeat this thousands of times with thousands of photos until the model reliably produces the correct results with an acceptable accuracy. The end result is a neural network that can reliably classify images.

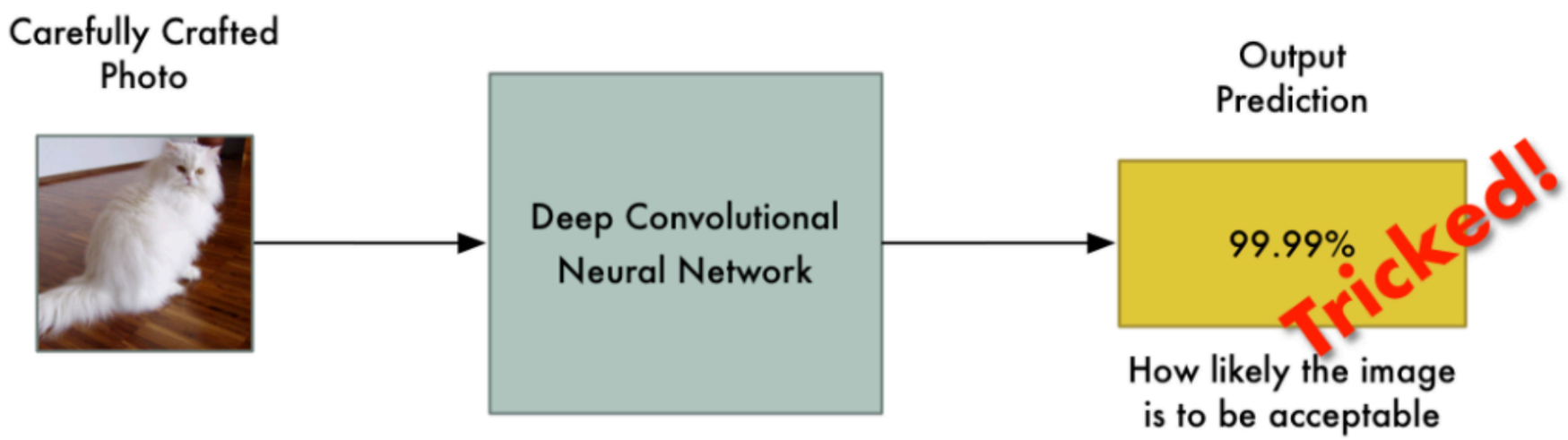


With a fancy model like this, we expect that small changes to the input photo should only cause small changes to the final prediction.

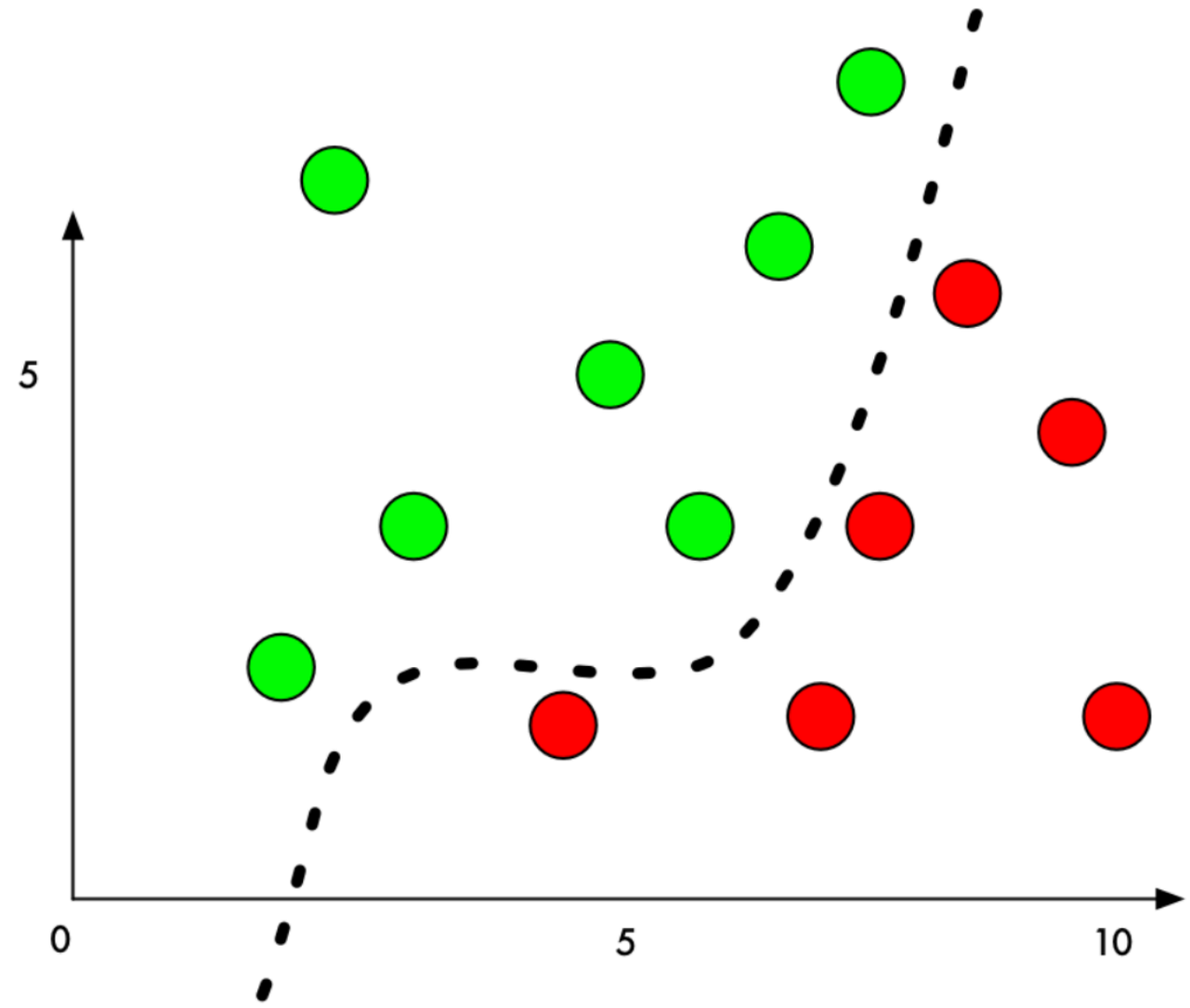


However, this is not always true. If you know *exactly which pixels to change* and *exactly how much to change them*, you can intentionally force the neural network to predict the wrong output for a given picture without changing the appearance of the picture very much.

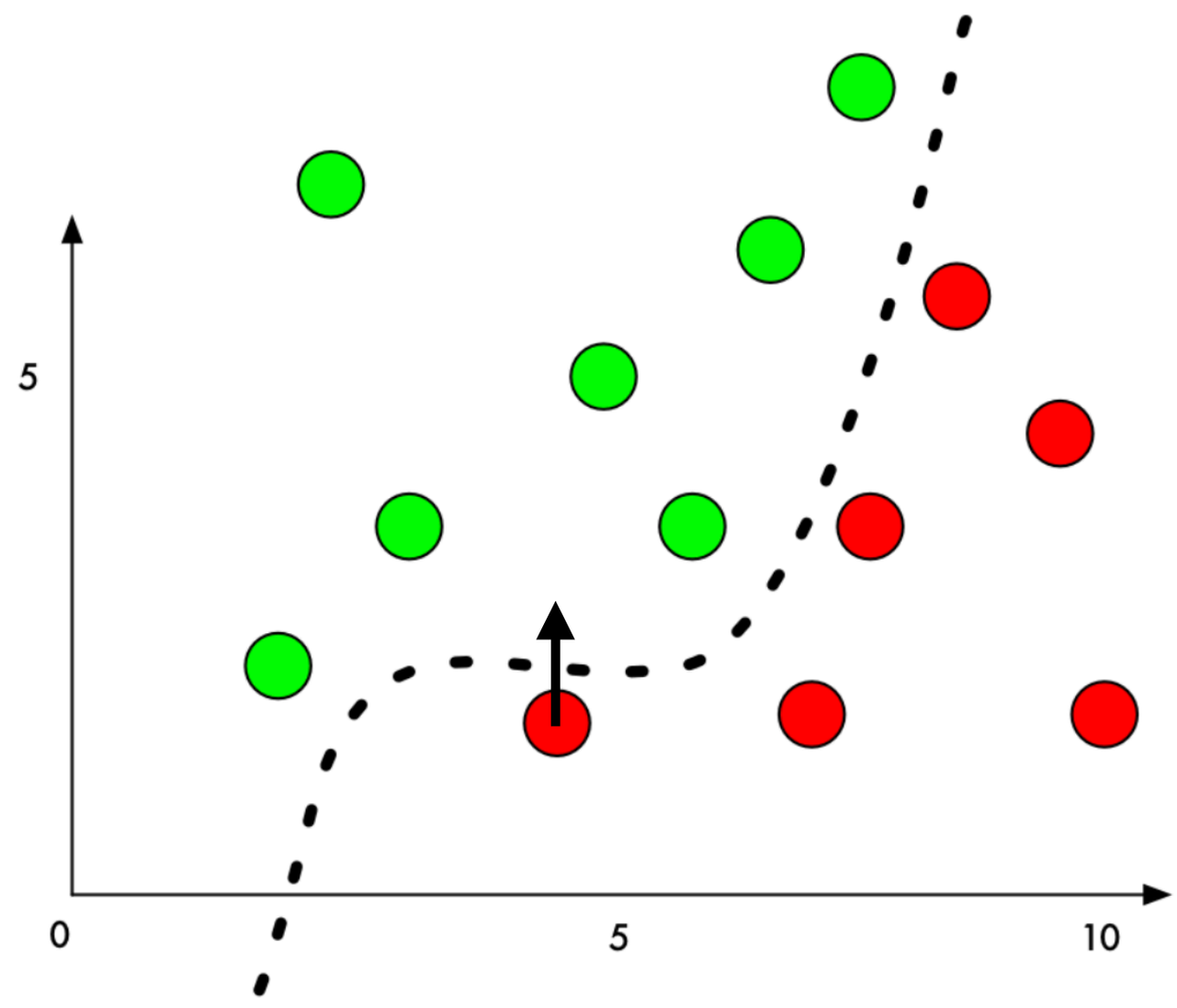
That means we can intentionally craft a picture that is clearly a prohibited item but which completely fools our neural network.



A machine learning classifier works by finding a dividing line between the things it's trying to tell apart.



If we add a small amount to the Y value of a red point right beside the boundary, we can just barely push it over into green territory.



In image classification with deep neural networks, each “point” we are classifying is an entire image made up of thousands of pixels. That gives us *thousands* of possible values that we can tweak to push the point over the decision line. And if we make sure that we tweak the pixels in the image in a way that isn’t too obvious to a human, we can fool the classifier without making the image look manipulated.

Original Image

Persian cat 87%
lynx 0%
Angora 0%
dishwasher 0%
Pomeranian 0%

Hacked Image

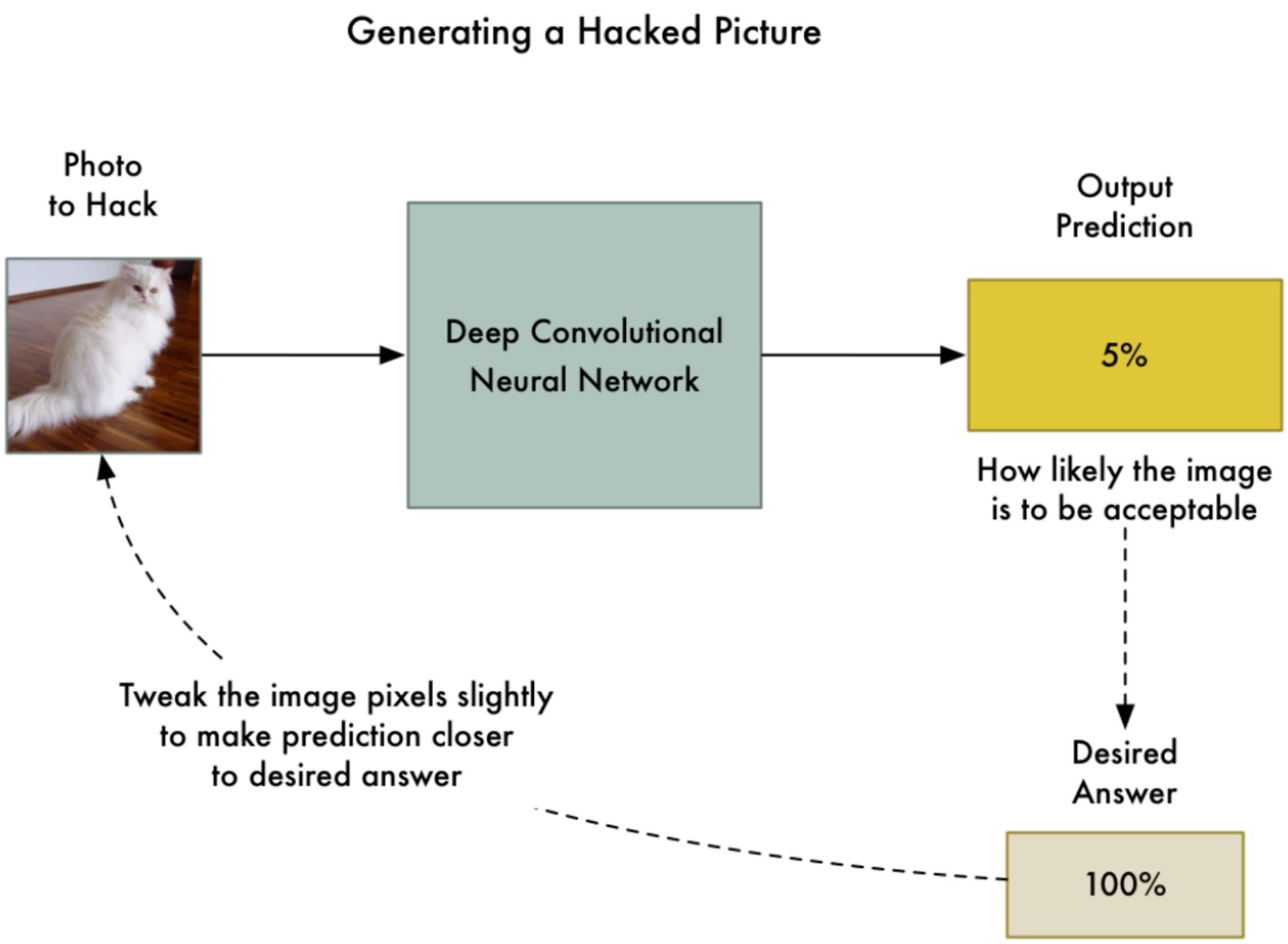
Original Image

Persian cat 87%
lynx 0%
Angora 0%
dishwasher 0%
Pomeranian 0%

Hacked Image

toaster 98%
Crock Pot 1%
Siamese cat 0%
wallaby 0%
carton 0%

How to trick a neural network



The only problem is that by allowing any single pixel to be adjusted without any limitations, the changes to the image can be drastic enough that you'll see them. To prevent these obvious distortions, we can add a simple constraint to our algorithm. We'll say that no single pixel in the hacked image can ever be changed by more than a tiny amount from the original image — let's say something like 0.01%. That forces our algorithm to tweak the image in a way that still fools the neural network without it looking too different from the original image.

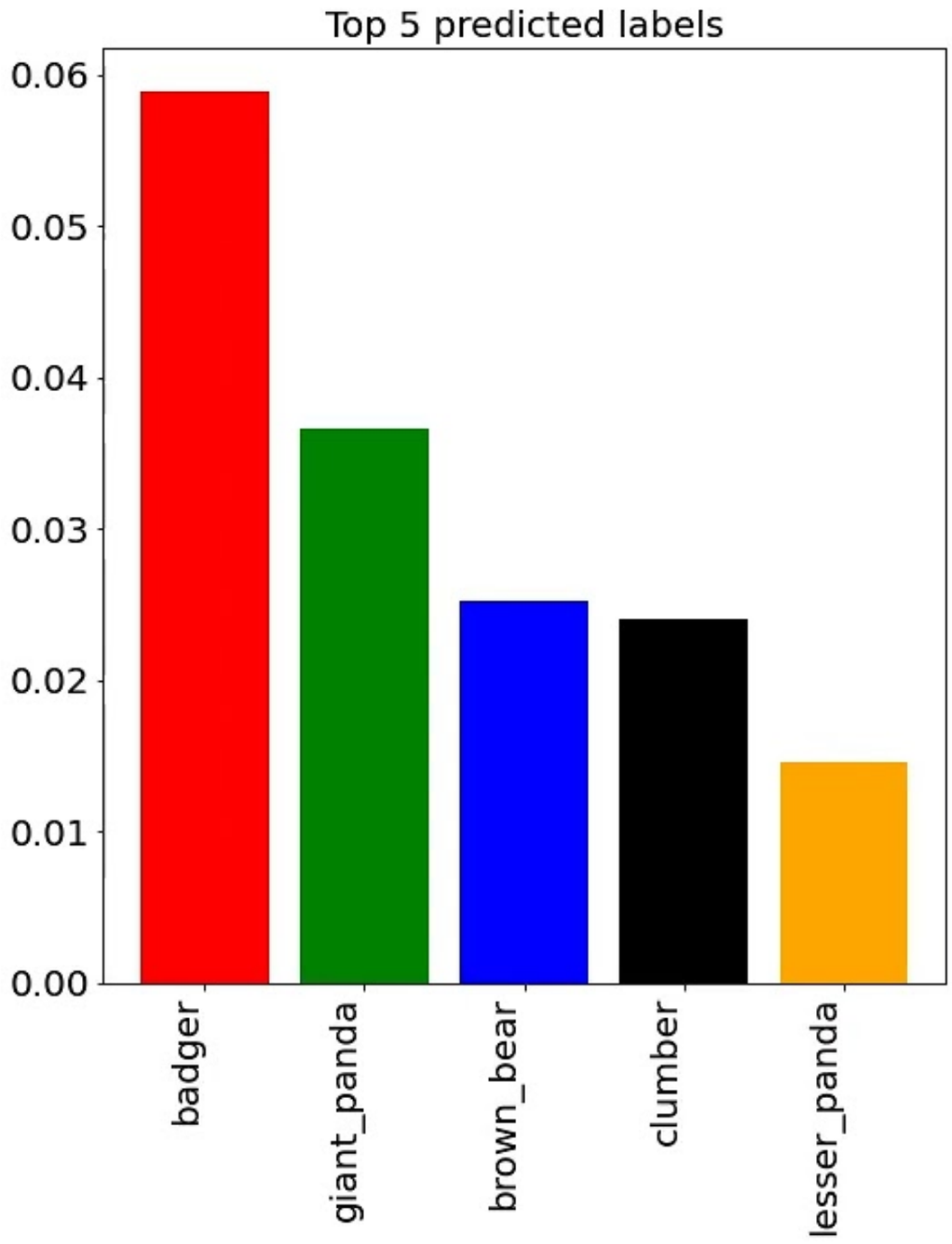
There is still a big limitation with how we create these images — our attack requires direct access to the neural network itself. Because we are actually "training" against the neural network to fool it, we need a copy of it. In the real world, no company is going to let you download their trained neural network's code, so that means we can't attack them… Right?

It turns out that you can train your own substitute neural network to mirror another neural network by probing it to see how it behaves. Then you can use your substitute neural network to generate hacked images that still often fool the original network! This is called a *black-box attack*.

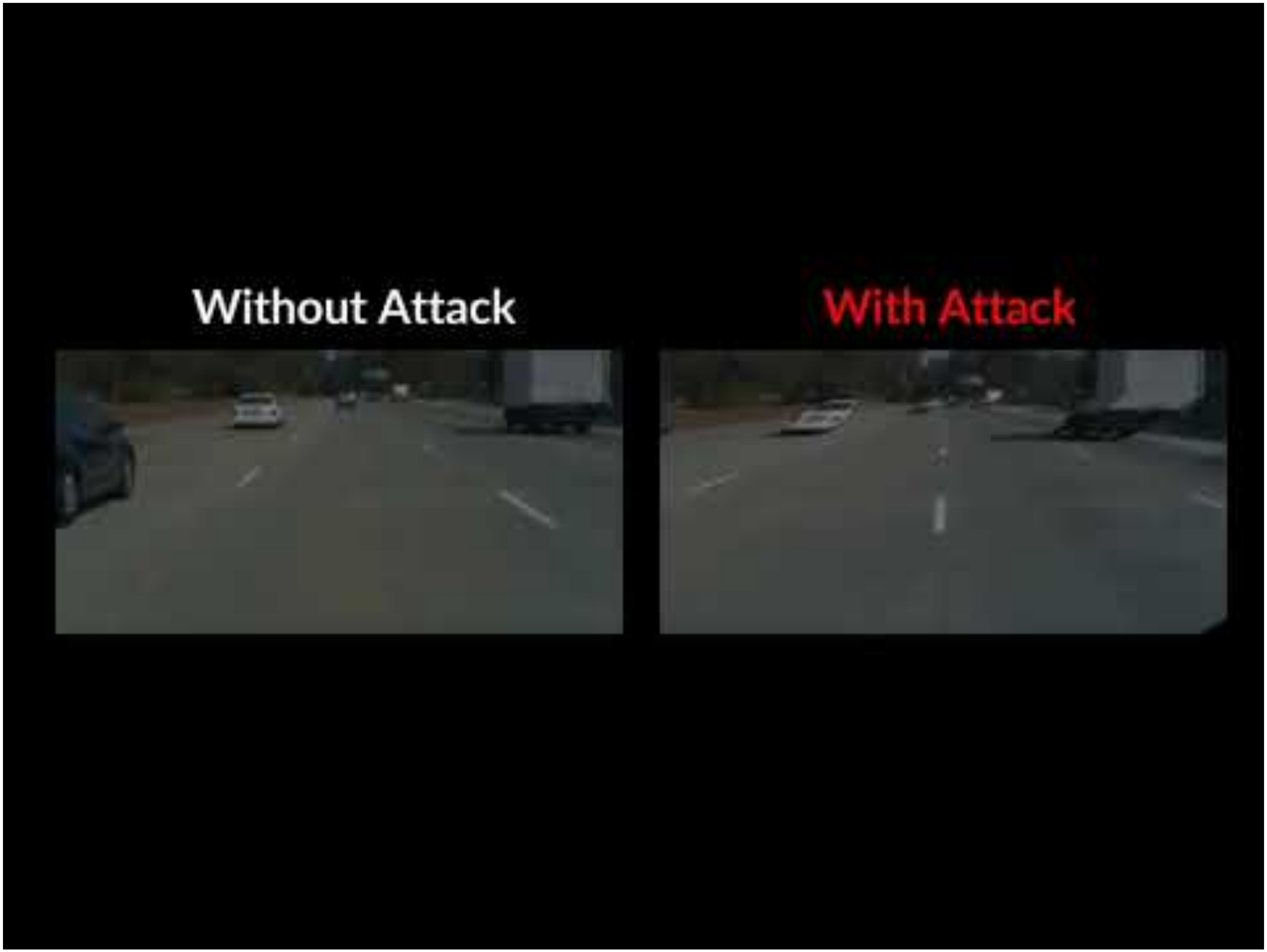
The applications of black-box attacks are limitless. Here are some plausible examples:

1. Trick self-driving cars into seeing a stop sign as a green light — this could cause car crashes!
2. Trick content filtering systems into letting offensive/illegal content through.
3. Trick ATM check scanners into thinking the handwriting on a check says the check is for a greater amount than it actually is.


Advarsarial Attack with FGSM (Untargetted)
(ε = 0.020): (badger, 5.88% Confidence)


Top 5 predicted labels
0.06
0.05
0.04
0.03
0.02
0.01
0.00
badger
giant_panda
brown_bear
clumber
lesser_panda

Lane-Keeping Assistance System



Possible protections

1. Create lots of hacked images and include them in the training dataset. It seems to make the neural network more resistant to these attacks. This is called Adversarial Training and is probably the most reasonable defense to consider adopting right now.
2. Treat ML models in your architecture like any other component that can potentially be bypassed. Think through the implications of what would happen if a user intentionally sets out to fool them and think of ways to mitigate those scenarios.