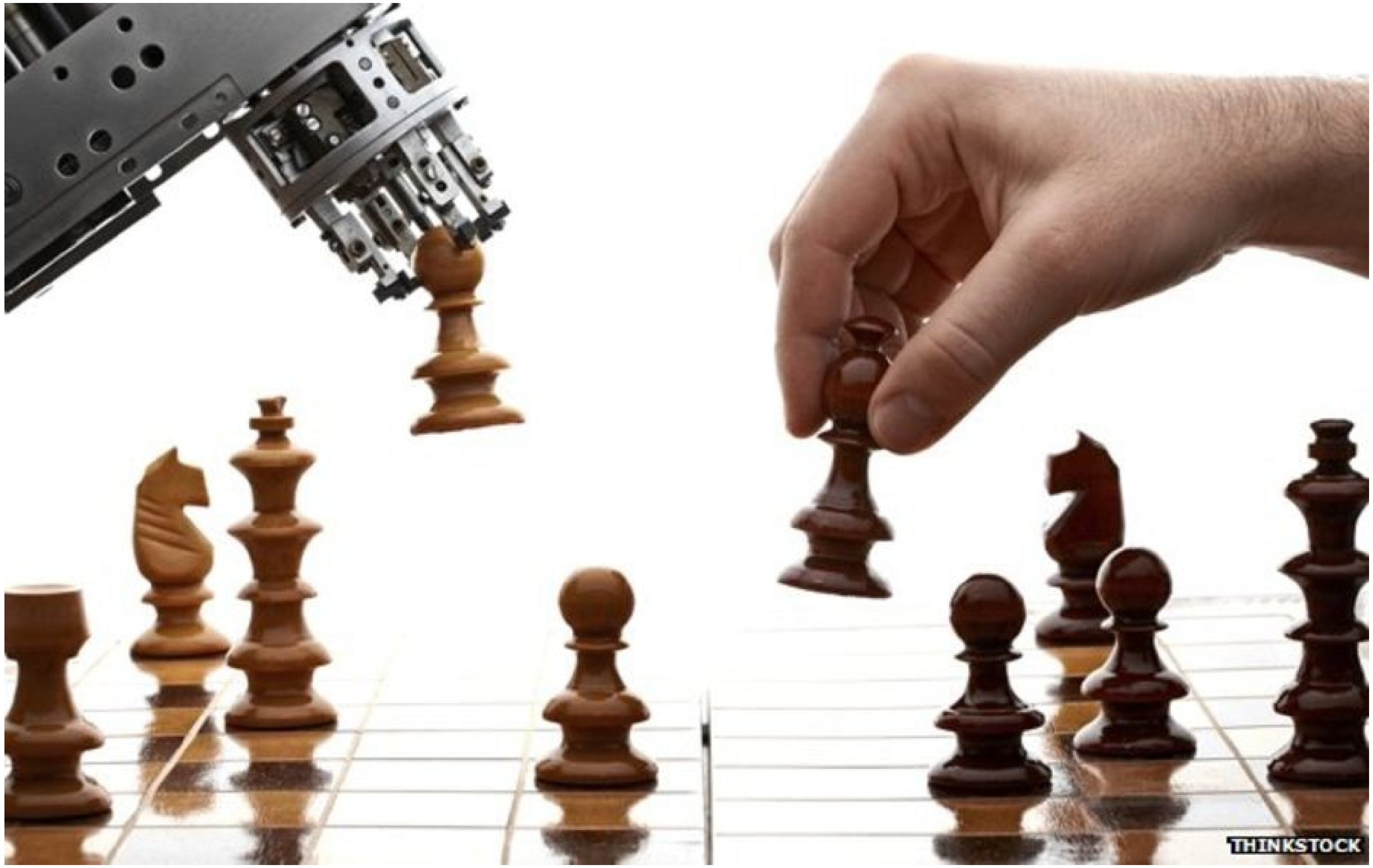



# Introduction to Artificial Intelligence

COSC 4550 / COSC 5550

Professor Cheney

11/15/17

mid-project check-in due Friday!

# unsupervised learning

learn underlying patterns in data
without any labeled training data (or rewards)

this is the hardest type of machine learning
(learning from the least information)

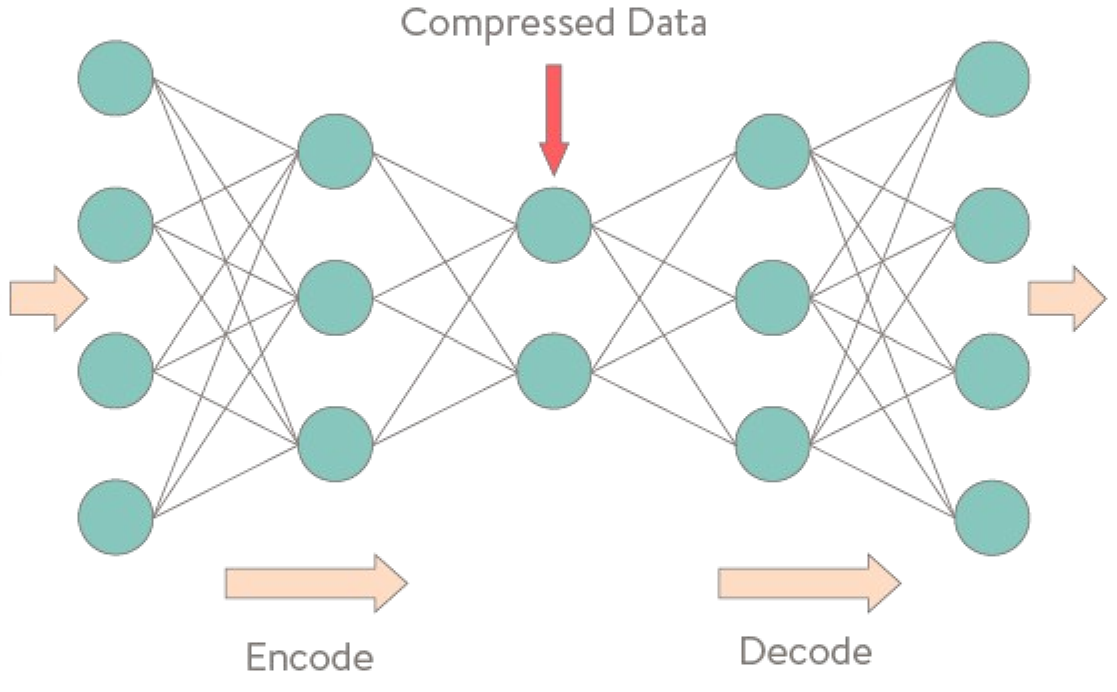
yet it might be the most important
(data streams without labels or rewards
make up the majority of data in the world)

as this problem is so broad and unstructured,
there are lots of approaches to unsupervised learning
(often with different types of patterns we're trying to learn)

# autoencoders

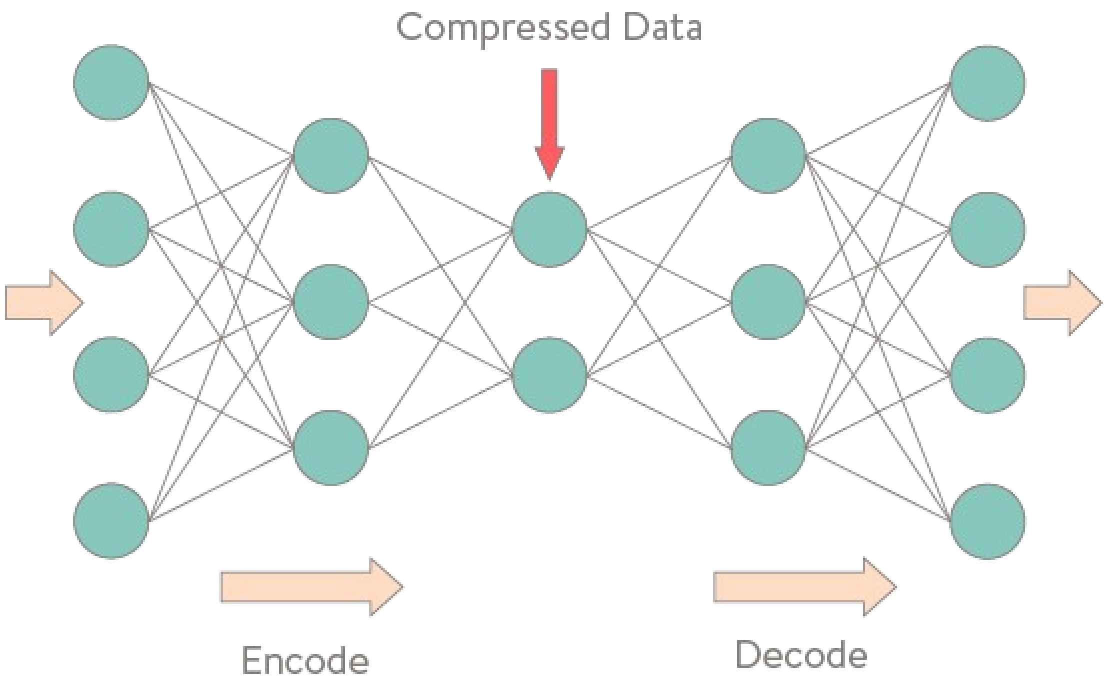
autoencoders seek to create a compressed (i.e. lower-dimensional) model of in input

they do so by trying to re-construct that input through a representation that includes a bottleneck layer

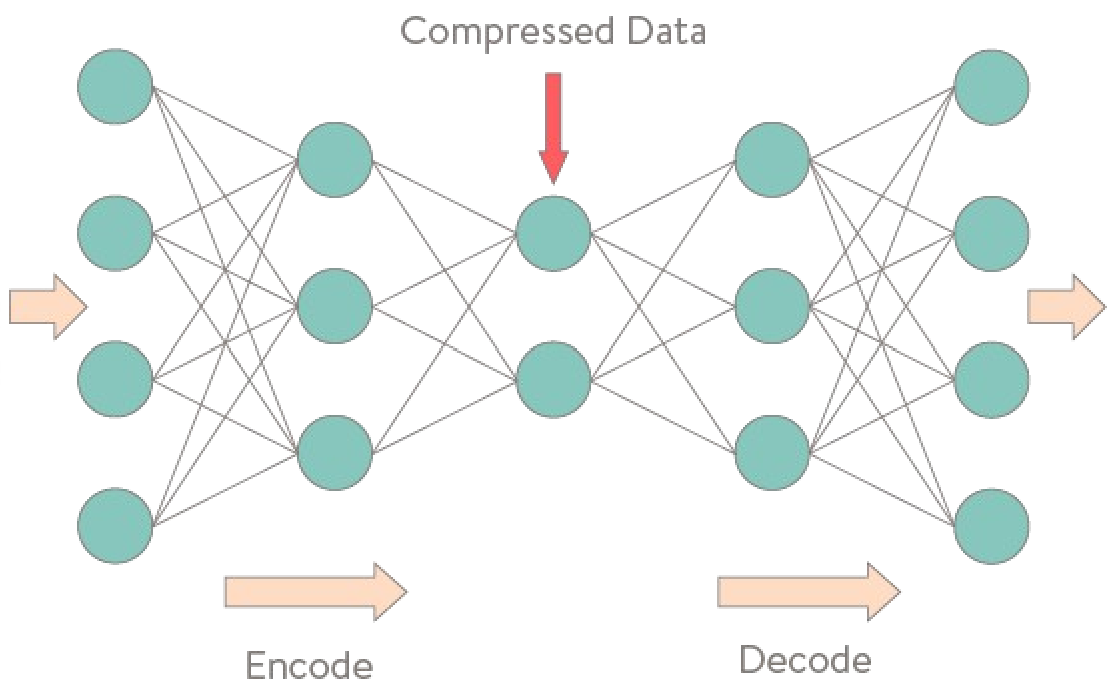


since the output layer is the same size as the input layer, the error is simply the difference between the two

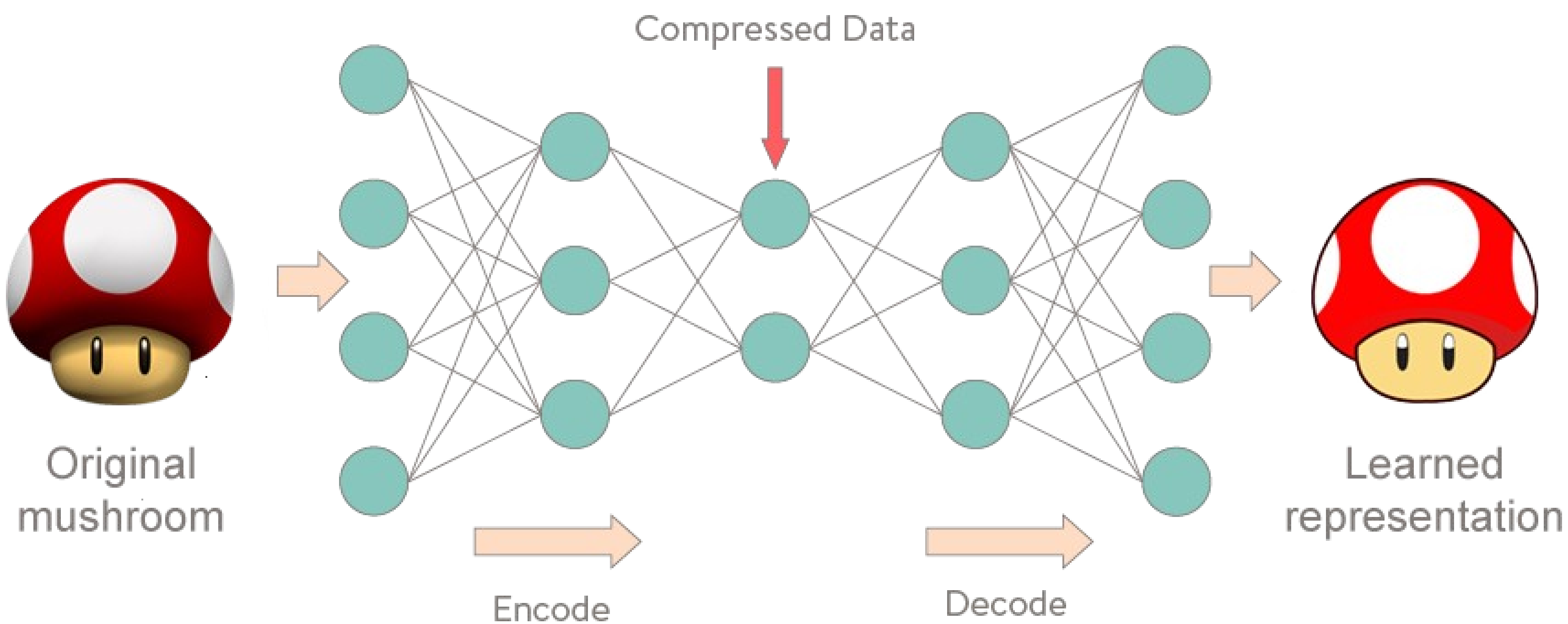
this means networks do not need any supervised labels, as the error signals come from reconstruction errors



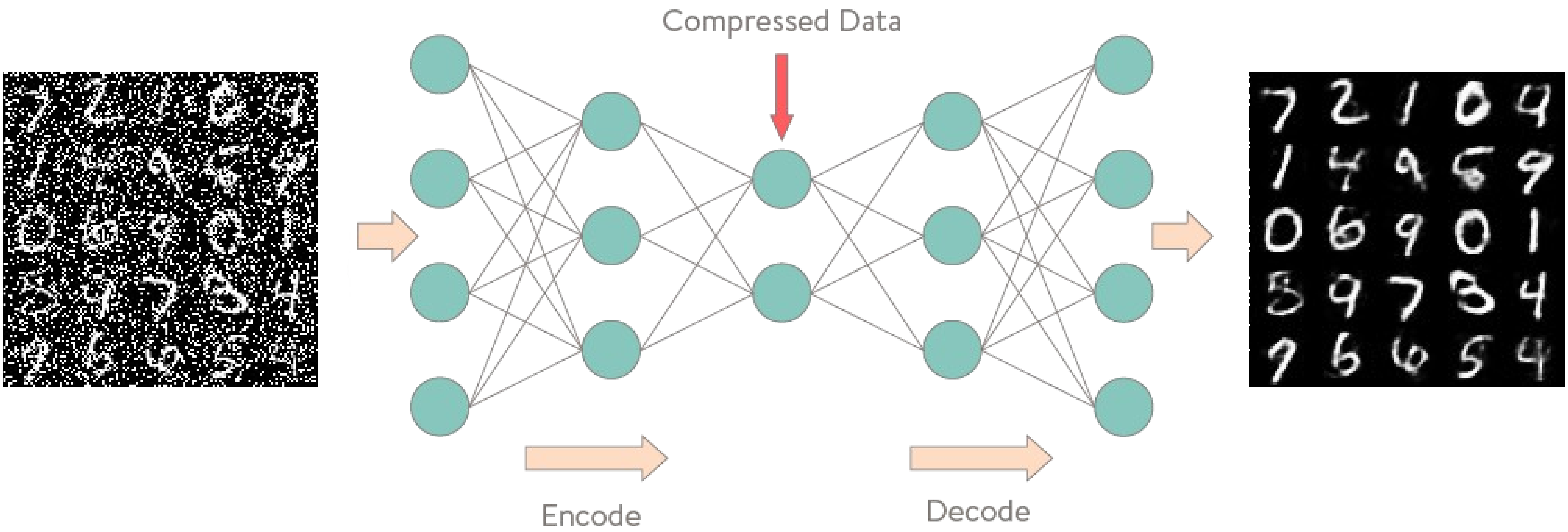
since the code layer is smaller than than the input/output (here 2-dimensional instead of 4-D), the autoencoder must learn an compressed (2-D) representation of the data, and it must capture (most of) the important features in the full (4-D) input to be able to reconstruct it!



this dimensionality reduction can lead to certain attributes of the image to be ignored (or all to be learned more succinctly… or just worse)

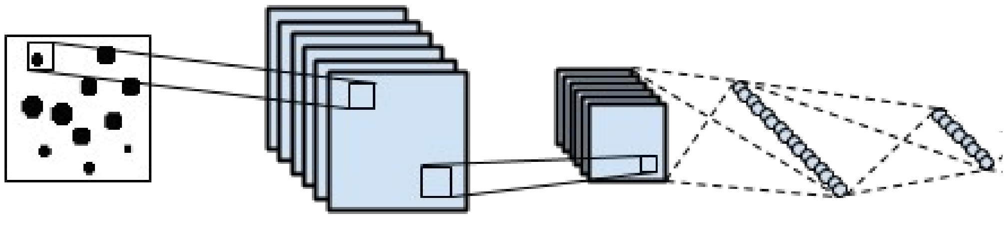


as noise is difficult to represent
(e.g. requires a powerful representation to overfit to noise)
autoencoders are often used for “denoising” data



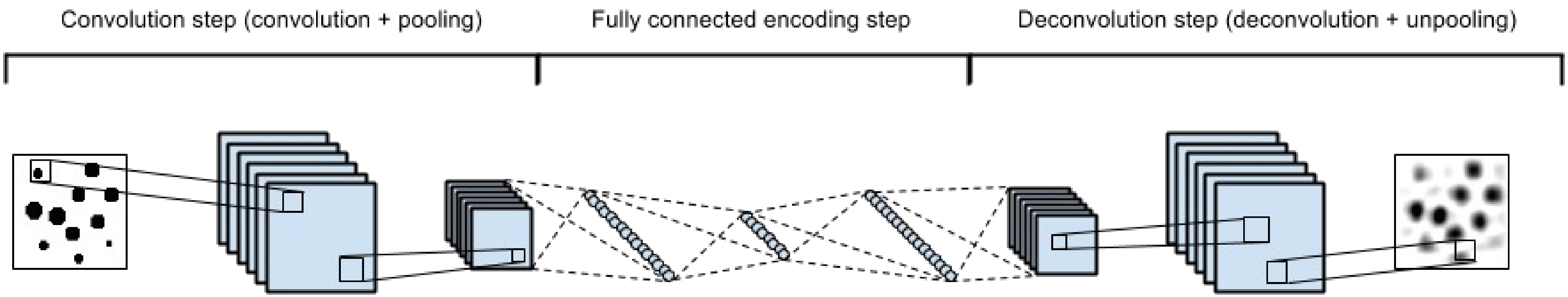
what if we used convolutional layers to build our neural network autoencoder?

convolution progresses just like image classification with deep neural networks did

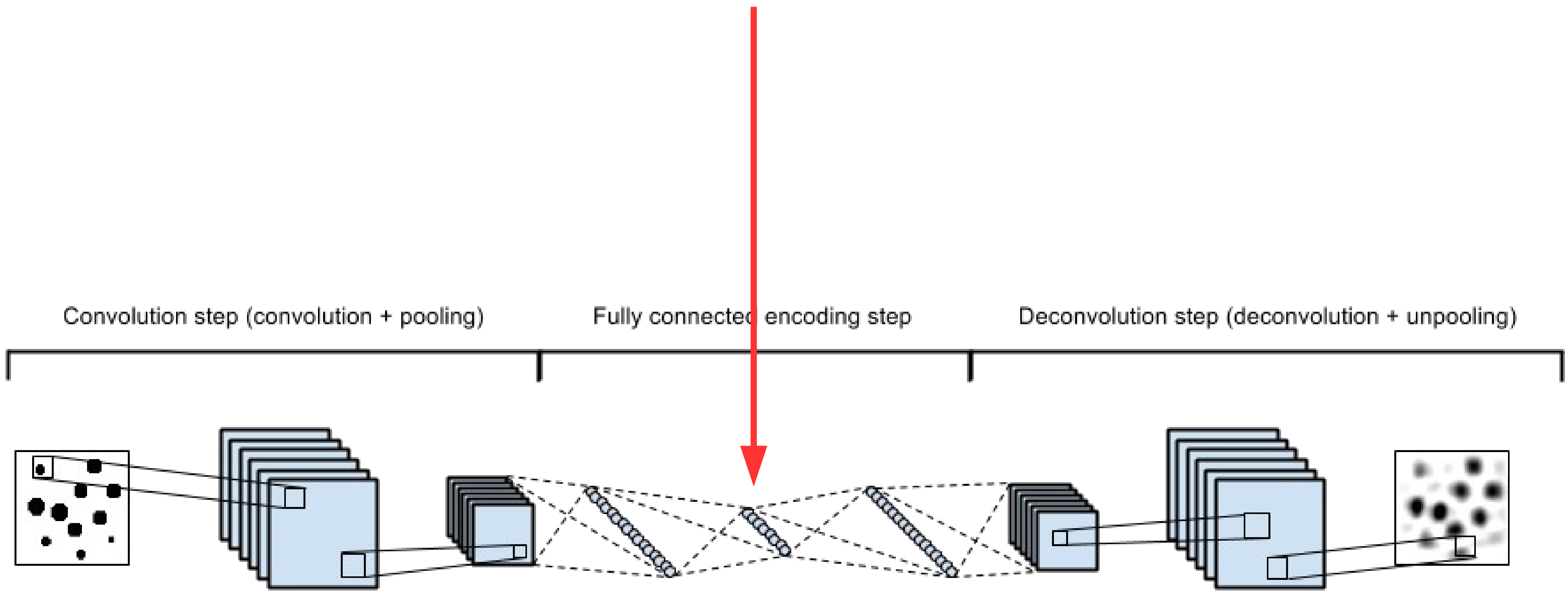


but now we use “decovolution” to take an input and filter and apply it repeatedly over the image, to upscale it

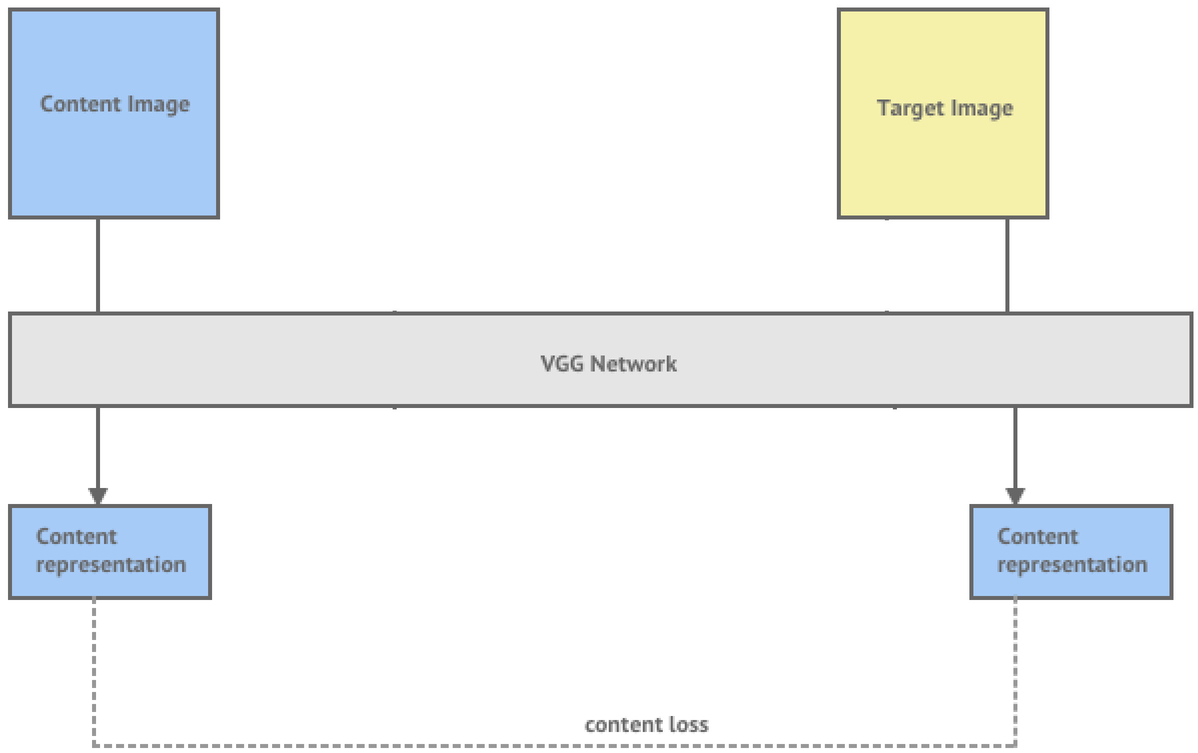
by connecting these two pieces into one big neural network, we can then use backpropogation **end-to-end** to train both the deconvolutional and convolutional layers from the image reconstruction error



this creates a *much* sparser representation
of a high-dimensional inputs (e.g. image)
by the activation of the smallest hidden layer
(i.e. the code layer)



# this idea of reconstructing an image can also be extended to construct an image from different part of two inputs

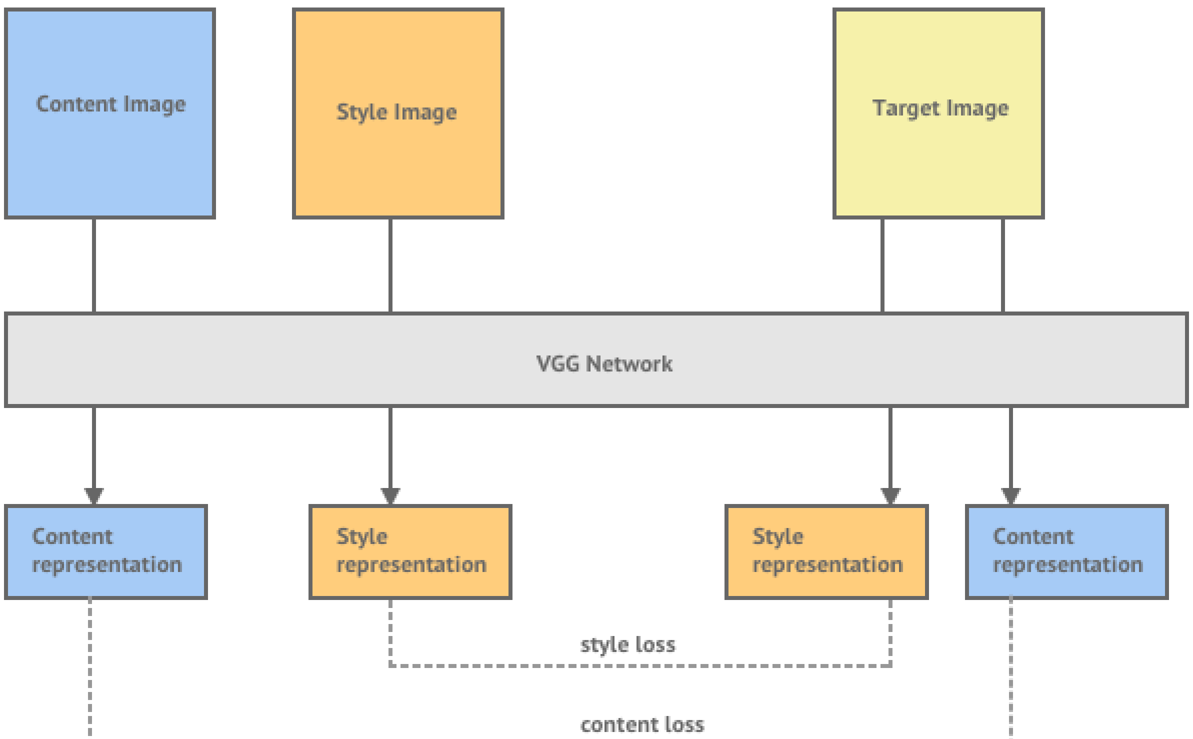



A Neural Algorithm of Artistic Style

Leon A. Gatys,[1,2,3*] Alexander S. Ecker,[1,2,4,5] Matthias Bethge[1,2,4]

this idea of reconstructing an image can also be extended to construct an image from different part of two inputs

e.g. taking the content from one, and style from the other



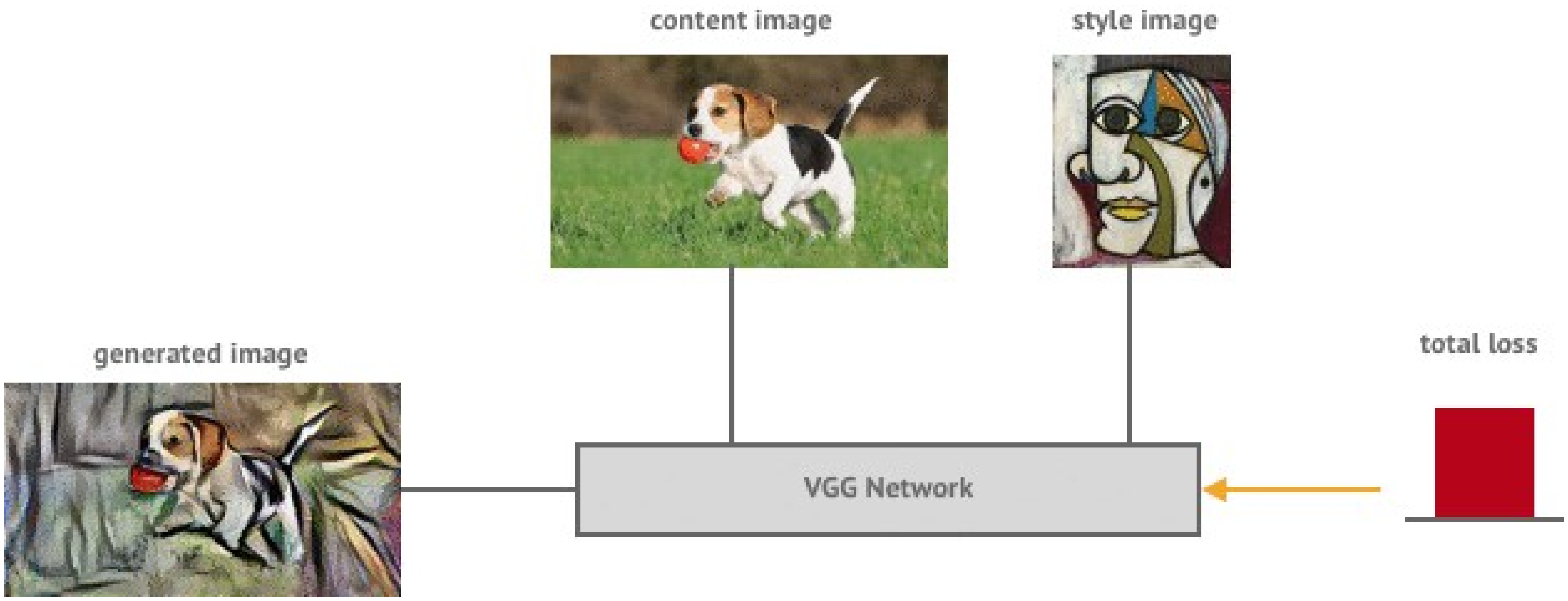

side note:

style loss is calculated based on spacial correlations of pixels in the resulting image

while content loss is based on the differences of hidden layers between encoding and decoding layers of the network

A Neural Algorithm of Artistic Style

Leon A. Gatys,[1,2,3*] Alexander S. Ecker,[1,2,4,5] Matthias Bethge[1,2,4] this creates a constructed image with the style of one input image, and the content of the other input image!



content + style = mash-up

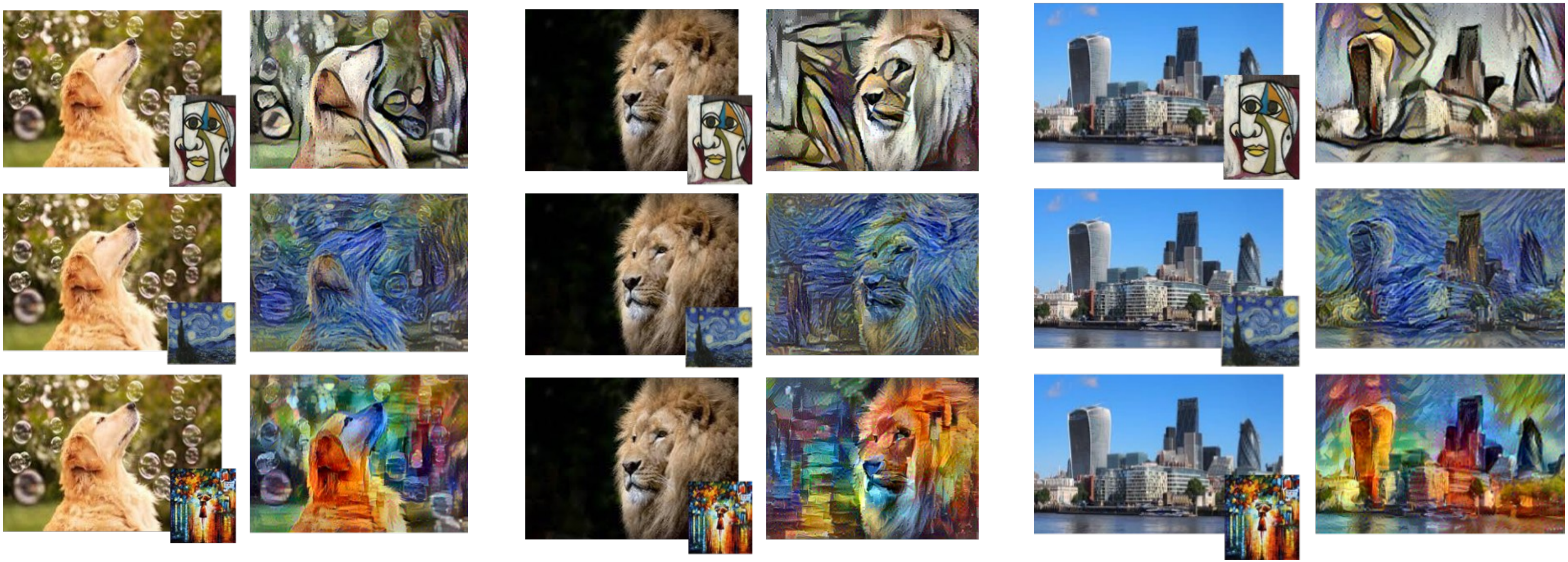
 +  = 

Content: Neckarfront in Tübingen, Germany

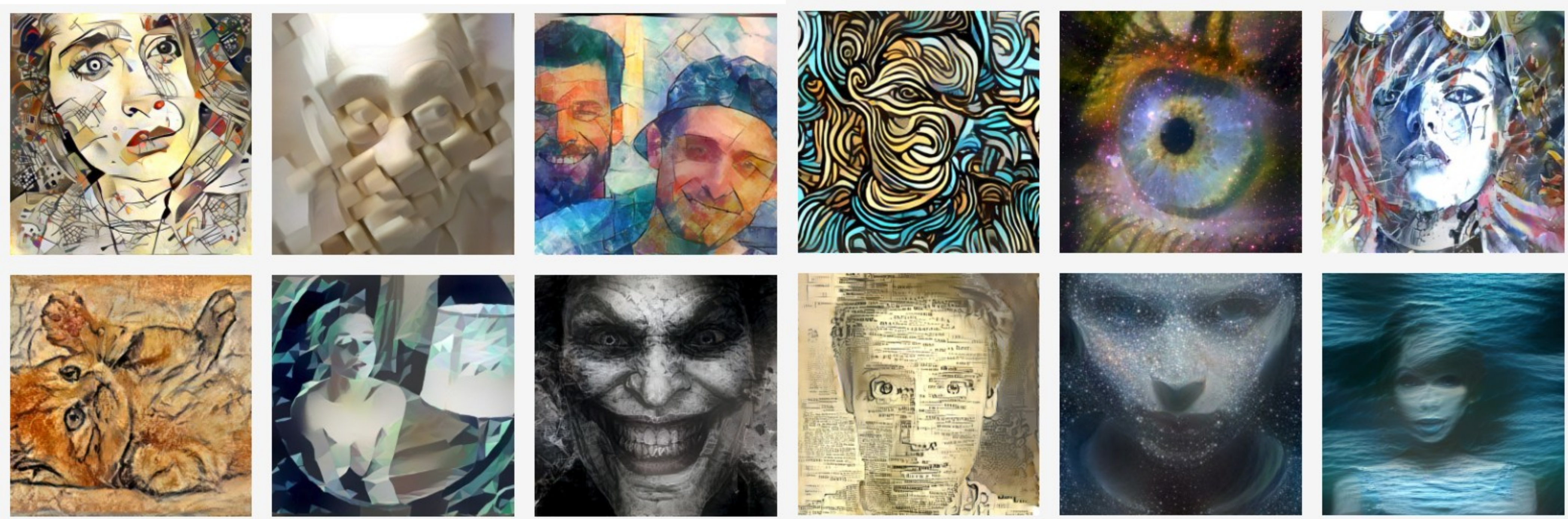
Style: The Shipwreck of the Minotaur, JMW Turner

Style: The Starry Night, Vincent van Gogh

Style: Der Schrei, Edvard Munch



# DEEPART.io



# generative adversarial networks

this approach is related to autoencoders and the methods that we've studies on deep networks for image classification

this unsupervised learning method that seeks to model the process that created your current data distribution

by modeling the generating process, there is an implicit assumption that you understand the underlying trends and behavior of the data, but creating new instances of patterns can also be important for applications (e.g. drug discovery)

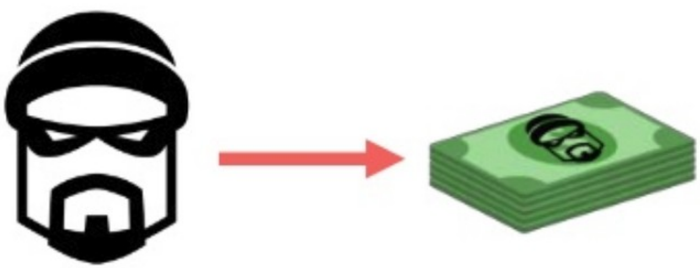
to accomplish this without labeled training data,
we actually train two networks

the first network generates a new data point (e.g. image)
that it tries to make as similar to the training data as possible
(e.g. set of unlabeled images)

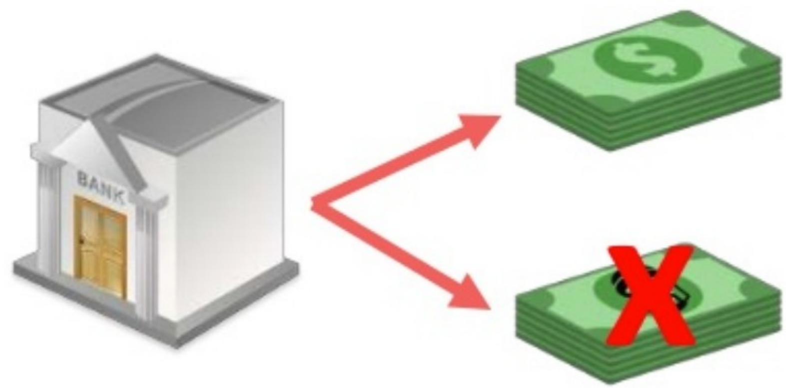
the second network is then given the real training data
as well as the new (fake/generated) data points
from the generator network, and tries to classify
which data points are real and which ones are
from the training set an which ones are generated

# What are GANs?

First, an intuition



**Goal:** produce counterfeit money that is as similar as real money.

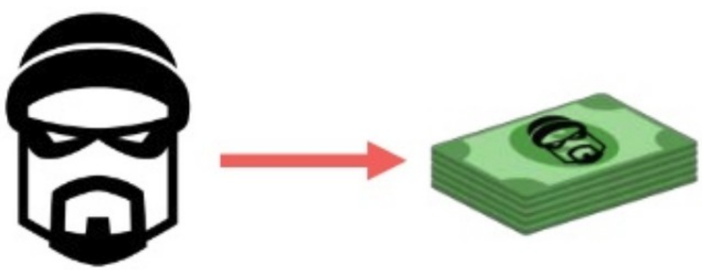



**Goal:** distinguish between real and counterfeit money.

# What are GANs?

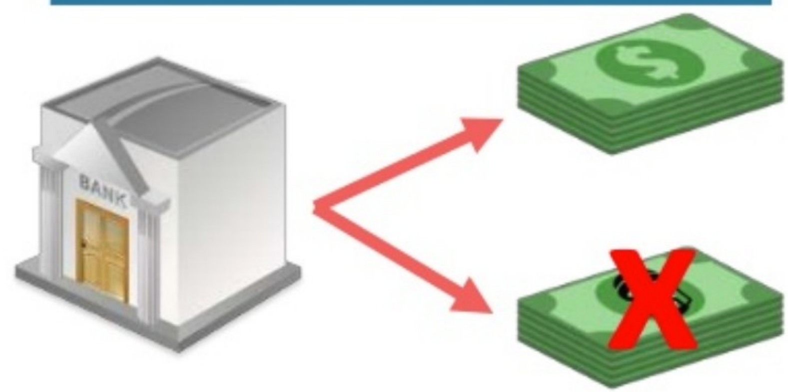
First, an intuition

## generator



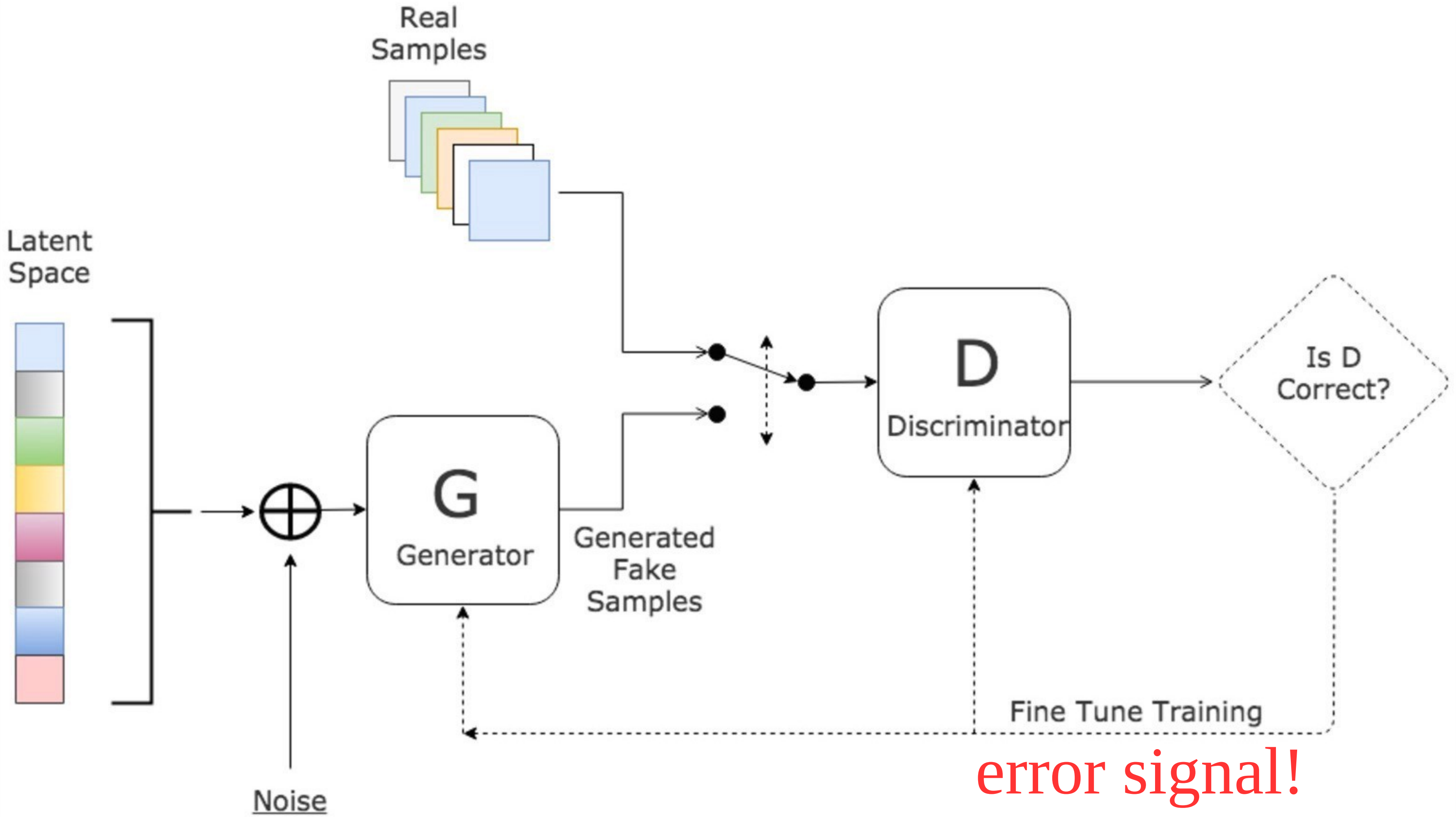
**Goal:** produce counterfeit money that is as similar as real money.

## discriminator




**Goal:** distinguish between real and counterfeit money.

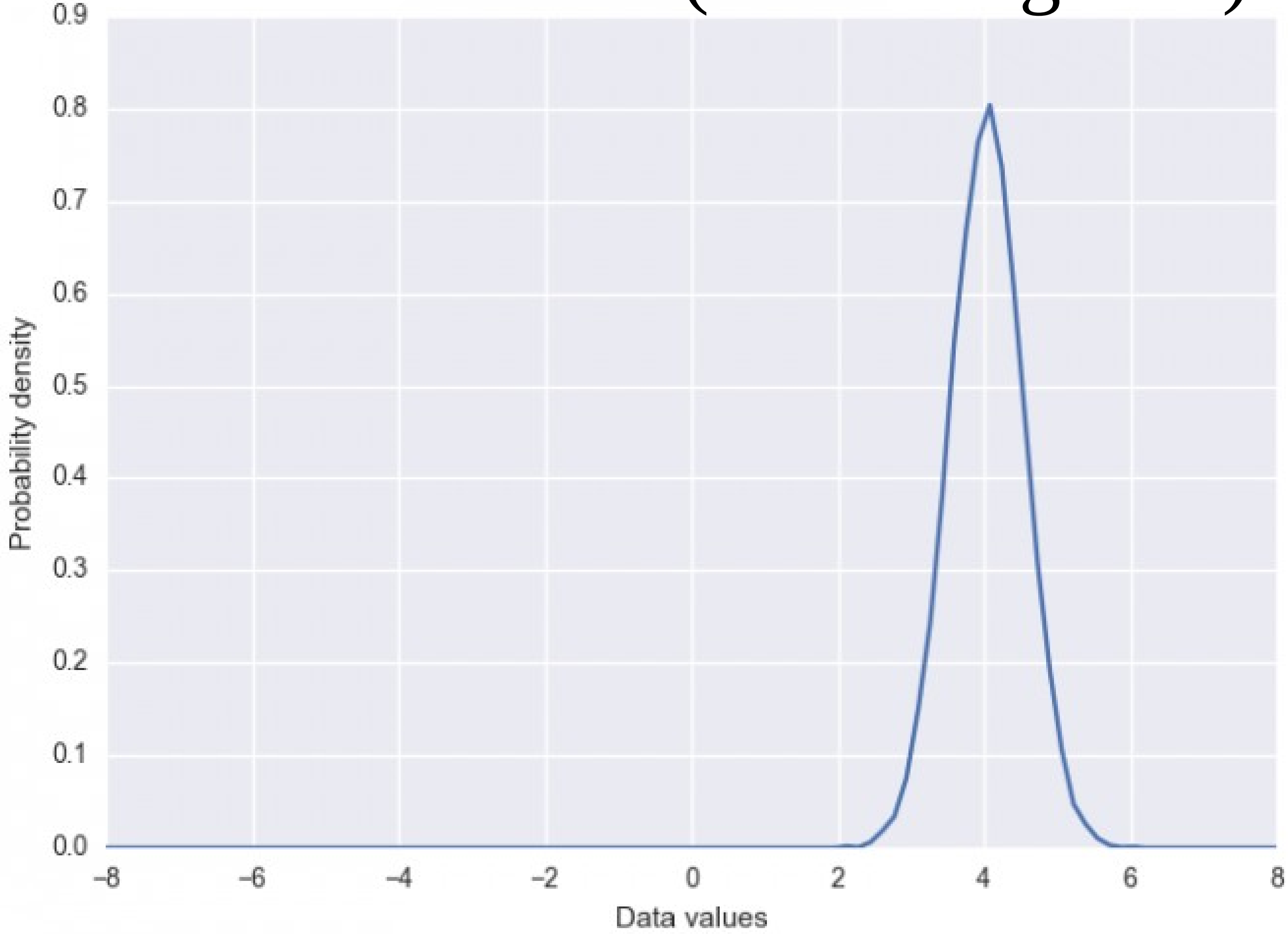
# Generative Adversarial Network



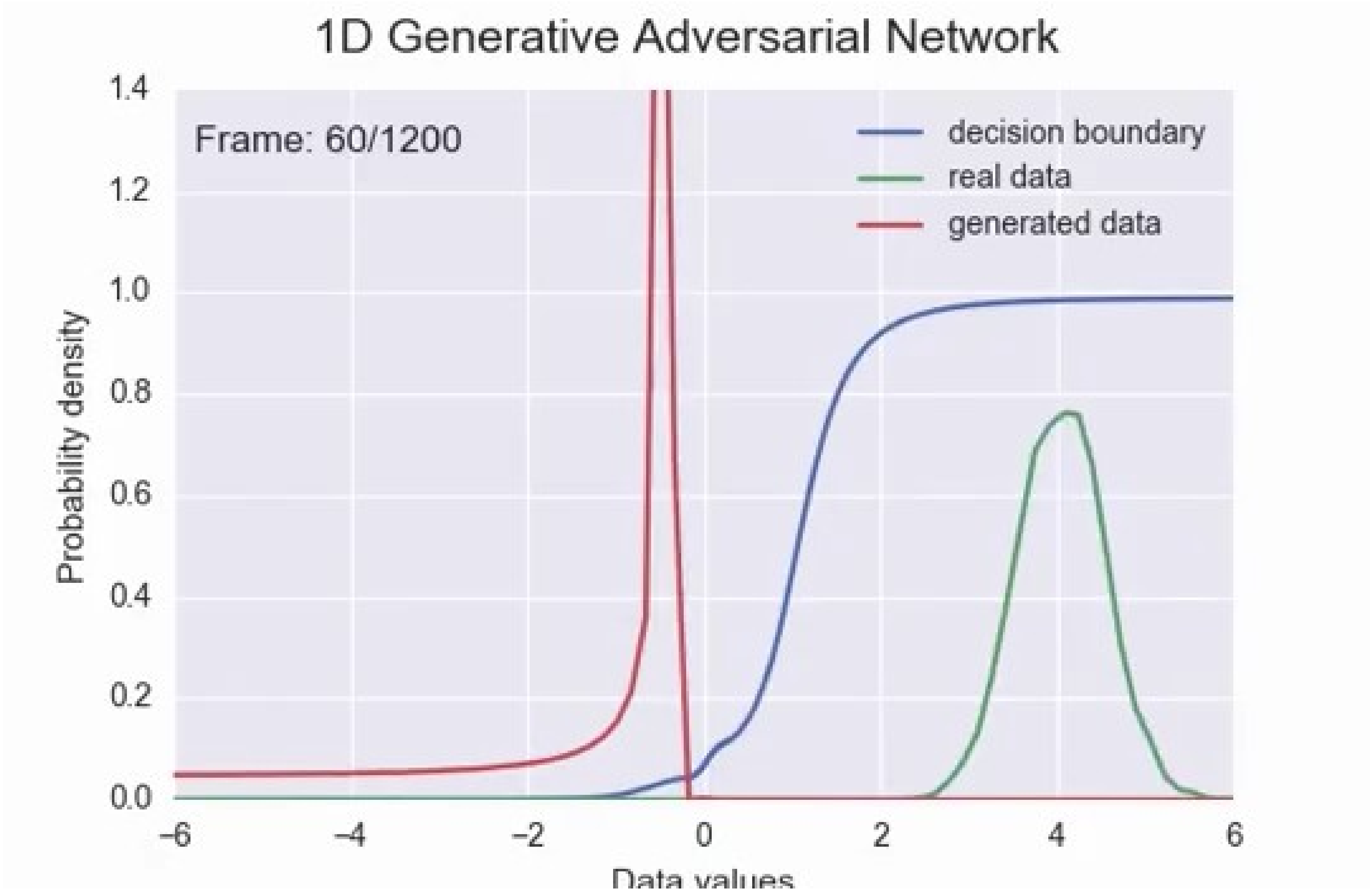

error signal!

https://www.kdnuggets.com/2017/01/generative-adversarial-networks-hot-topic-machine-learning.html let's consider a 1-D example, where the generator is trying to estimate the distribution of a single value

true distribution (of training data):



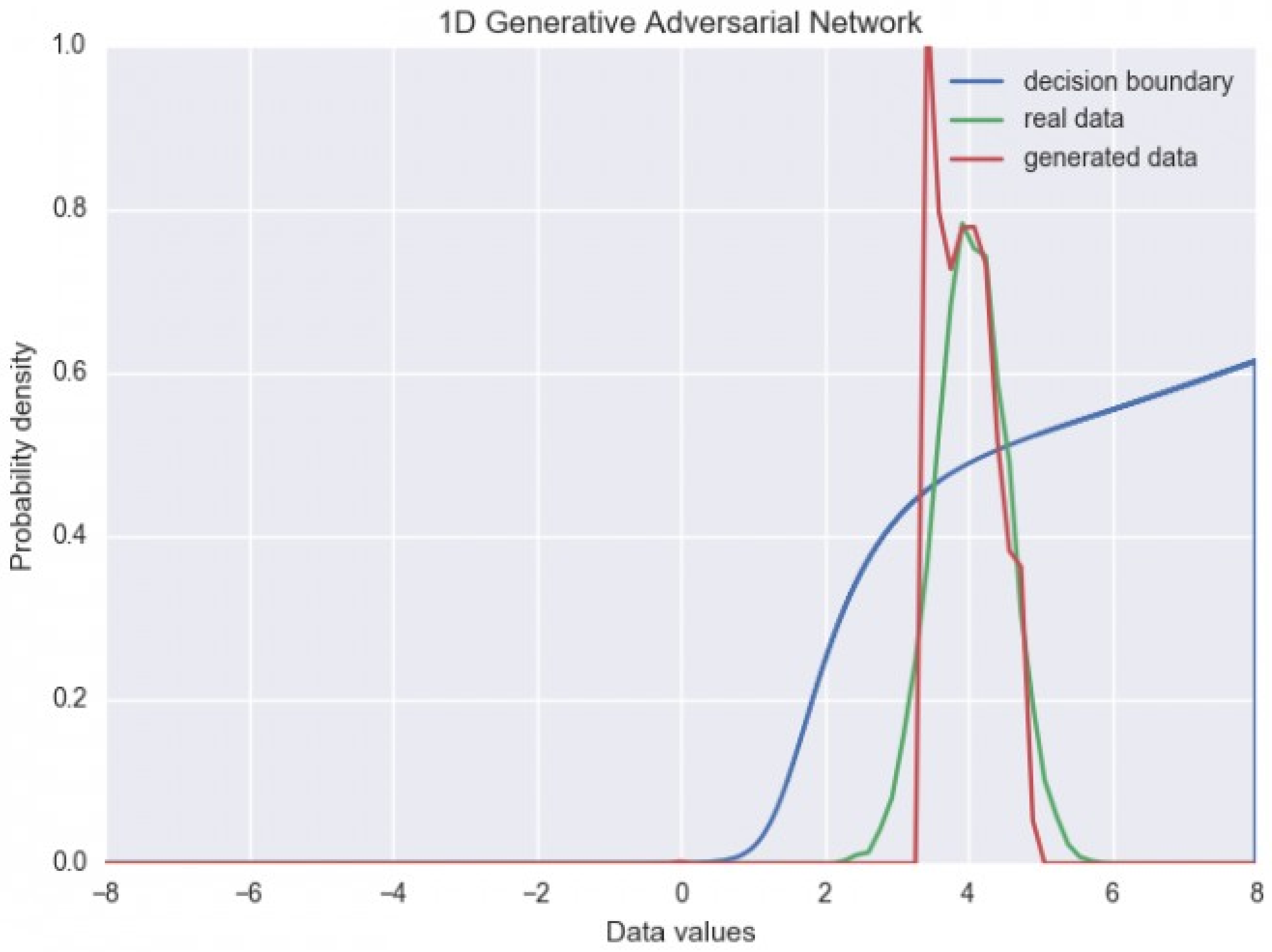

http://blog.aylien.com/introduction-generative-adversarial-networks-code-tensorflow/




1-layer neural network generator
3-layer neural network discriminator

http://blog.aylien.com/introduction-generative-adversarial-networks-code-tensorflow/

by the end of training, the generator network is able to (almost perfectly) replicate the distribution from the original training example set that was given to the discriminator



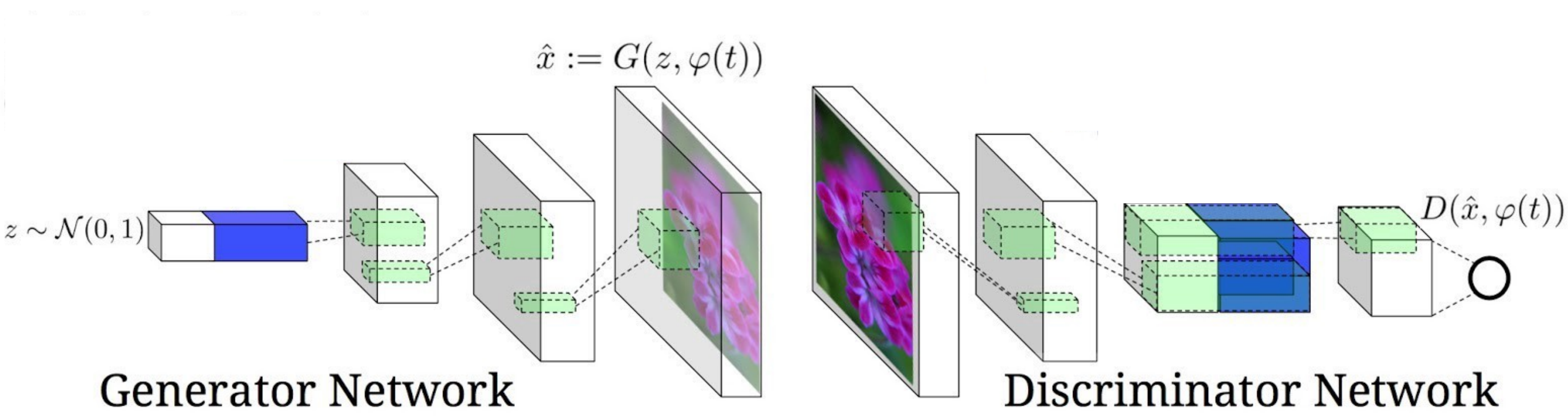

http://blog.aylien.com/introduction-generative-adversarial-networks-code-tensorflow/

we almost perfect learned a model of the example
data points with no labeling or rewards! this is super cool!

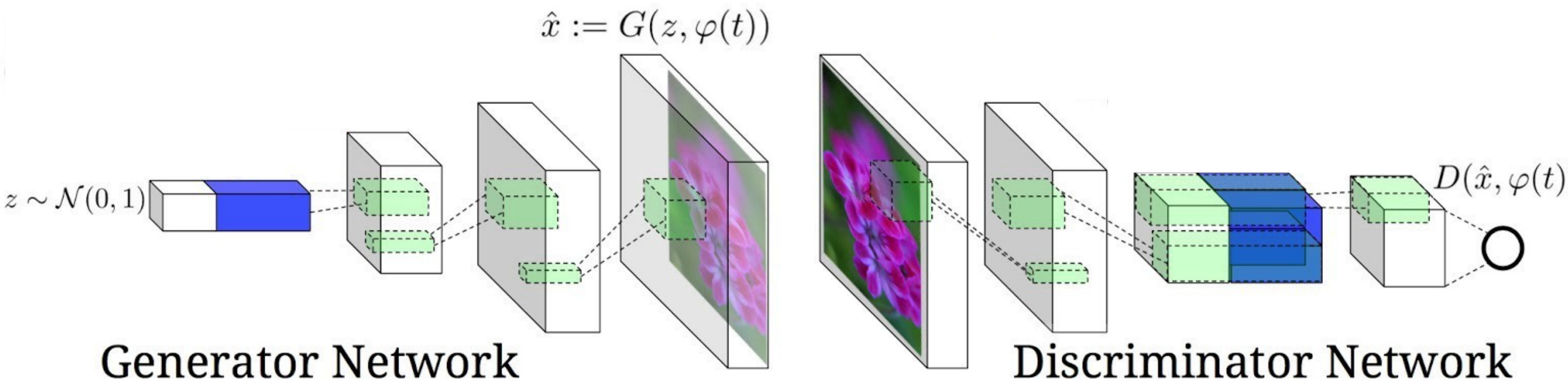
what if we used deep convolutional neural networks
as our generator and discriminator,
and used images as our example dataset?

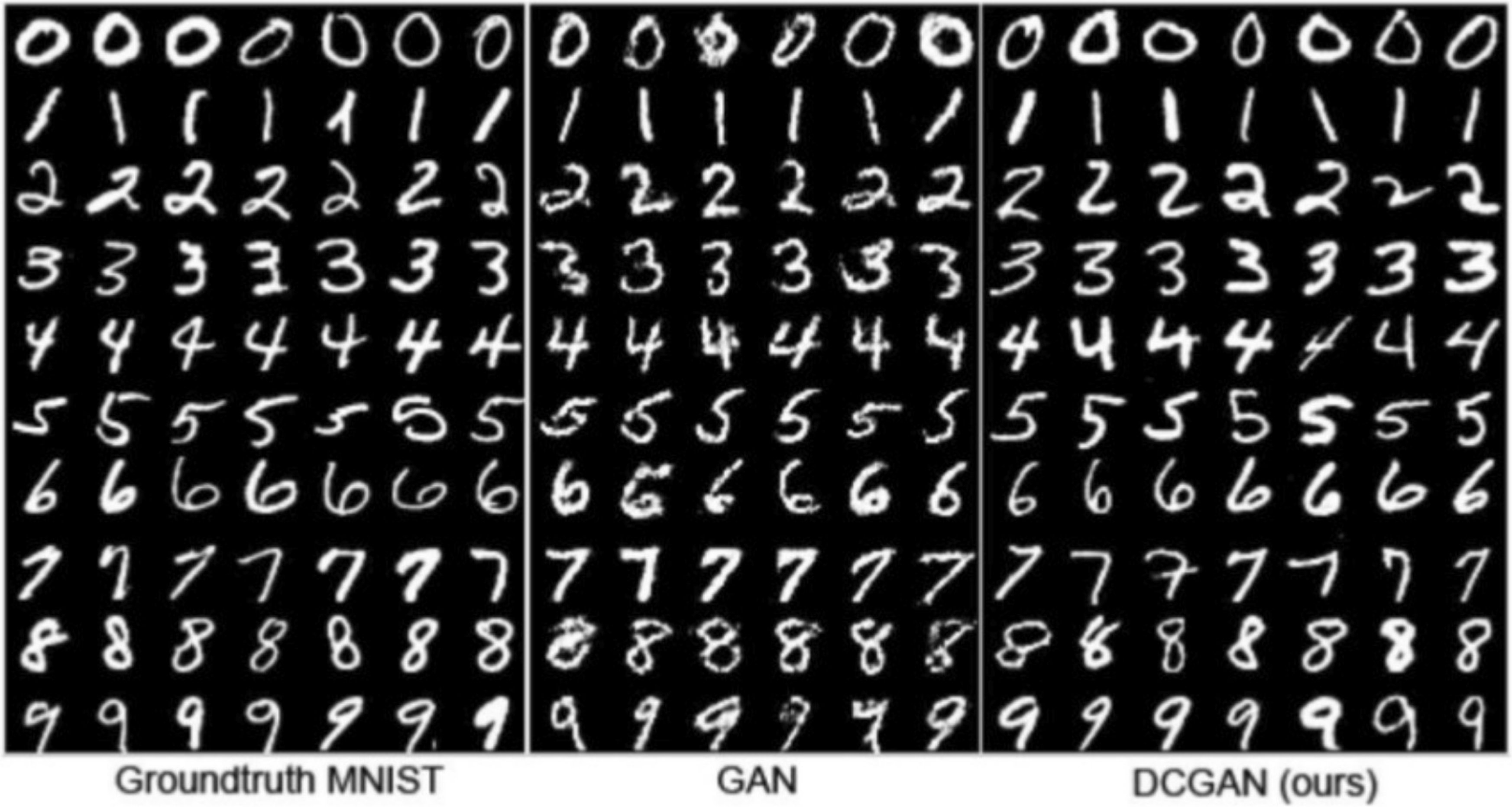
discriminator networks are just like the deep convolutional neural networks we've seen for image classification before

but generator networks are now “decovnolutional” deep neural networks, that go from an vector to an image



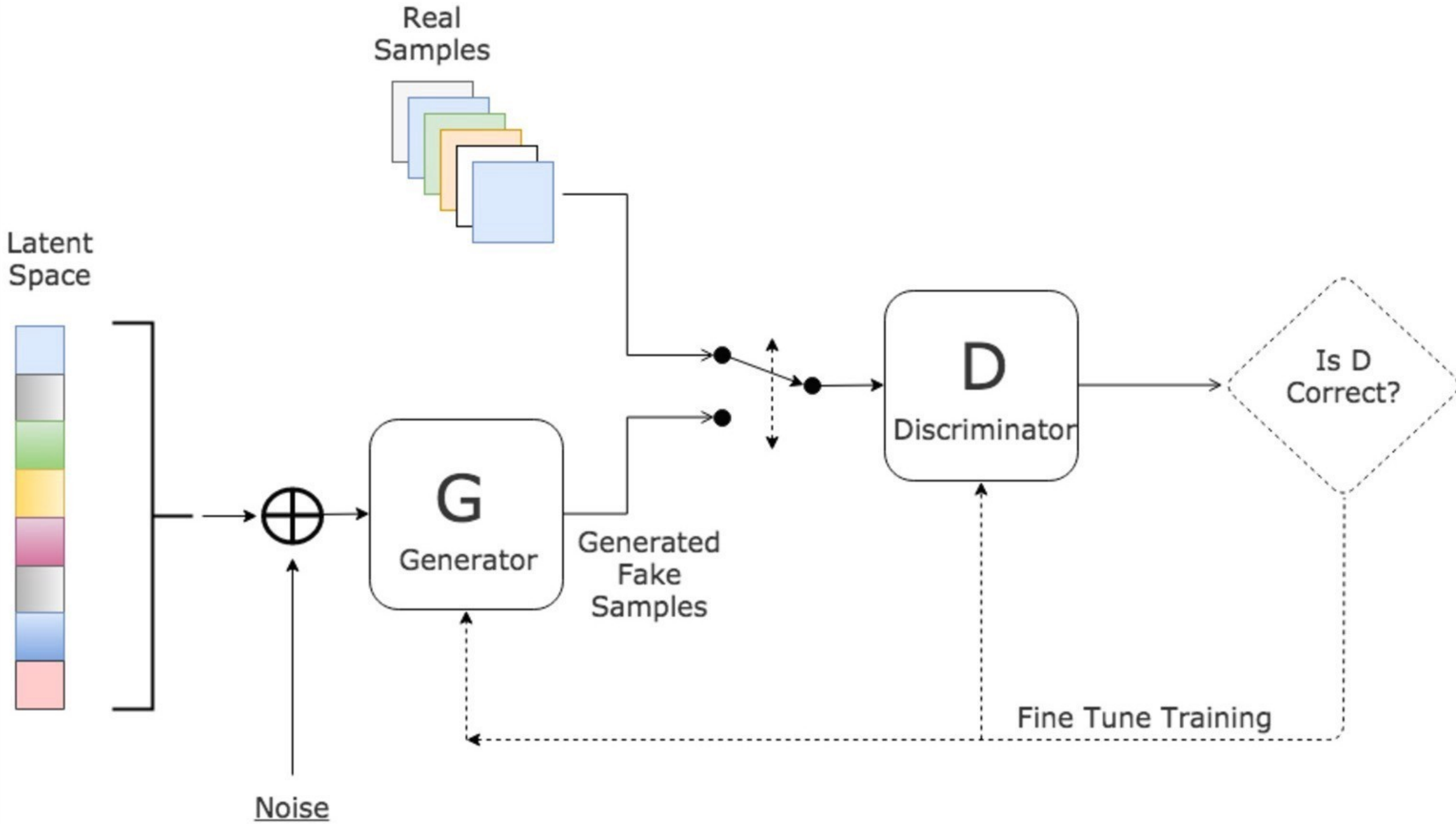
by connecting these two together
(where the output layer of the generator
is the input layer to the discriminator),
we can take the classification error of the discriminator
and backpropagate it all the way through both networks
(opposite to autoencoders) with **end-to-end training**!






*Source: Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2015).*

# Generative Adversarial Network

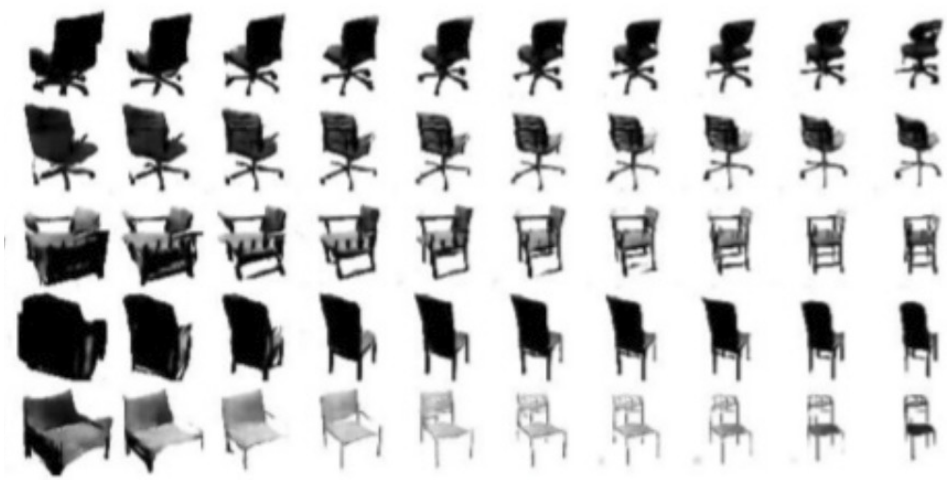



https://www.kdnuggets.com/2017/01/generative-adversarial-networks-hot-topic-machine-learning.html

# InfoGAN

Deep Convolutional GAN – Xi Chen et al. (2016)

(a) Rotation

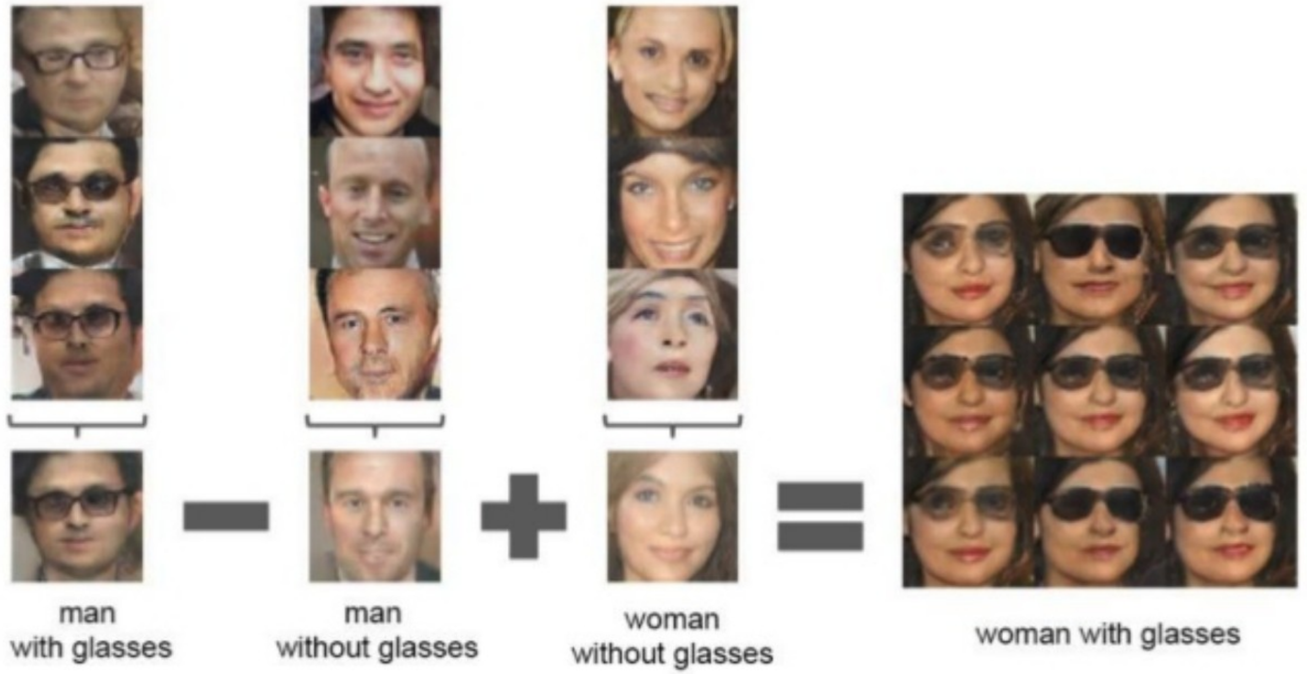


(b) Width

*Source: Chen, Xi, et al. "Infogan: Interpretable representation learning by information maximizing generative adversarial nets." Advances in Neural Information Processing Systems. 2016.*

# DCGAN – Vector Arithmetic

Deep Convolutional GAN – Alec Radford et al. (2016)




*Source: Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2015).*

# Generative Visual Manipulation on the Natural Image Manifold

Jun-Yan Zhu
Philipp Krähenbühl
Eli Shechtman
Alexei A. Efros

Berkeley
Adobe

ECCV'16
EUROPEAN CONFERENCE ON COMPUTER VISION