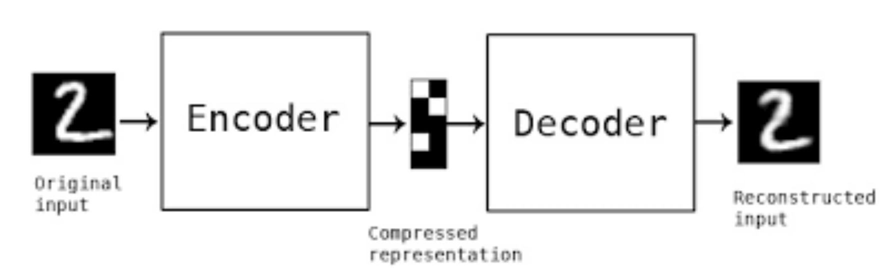
Autoencoders

Taryn Heilman
Deep Learning Workshop
May 12, 2018
galvanize



Learning Objectives

- **Review** previous neural network cells and architectures
- **Understand** autoencoders and their many uses in machine learning
- Be ready to **tune** a fully connected and then a convolutional autoencoder

Review relevant concepts

What is an epoch?

Where is the learning in neural networks stored?

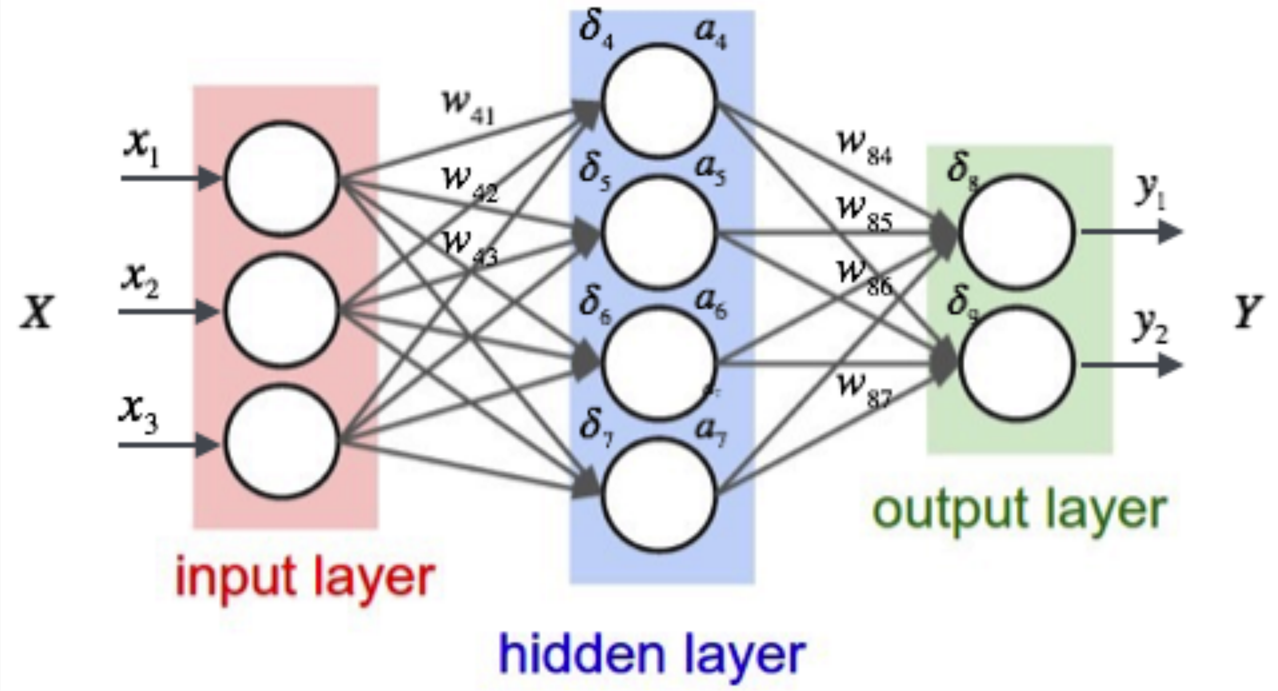
How do weights get updated?

Name some hyper-parameters that you can tune in a neural network

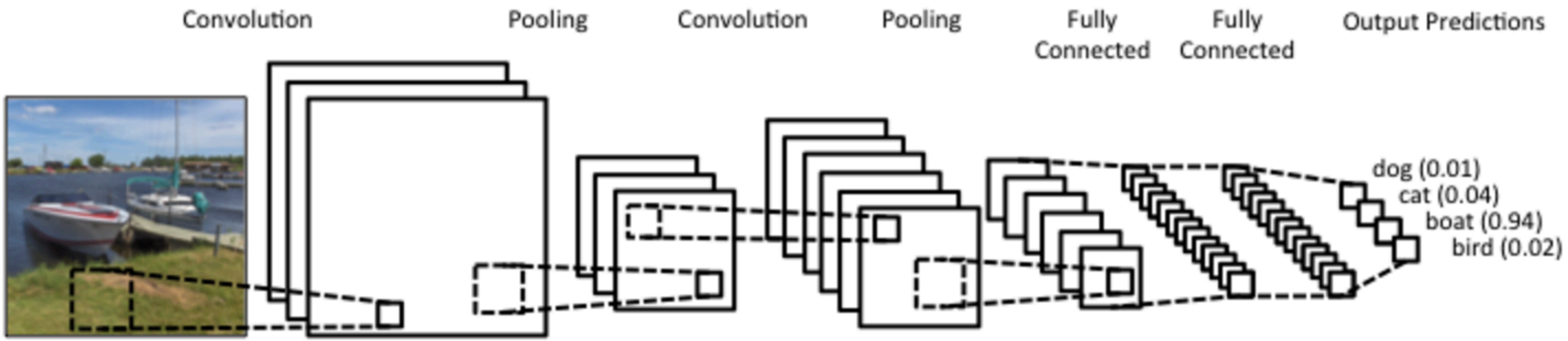
Which activation function do you use for your output layer for a classification network?

What does “supervised learning” mean?

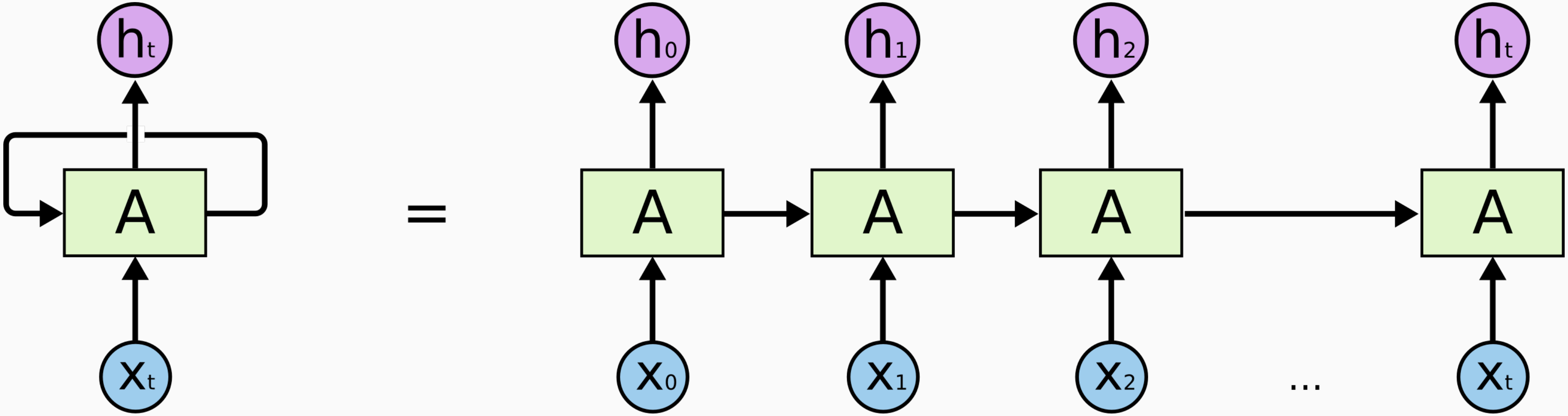
Before: Multi-Layer Perceptron (Fully connected neural networks)



Before: Convolutional Neural Networks



Coming Up: Recurrent Neural Networks



Now: Autoencoders

- Last few sessions have covered different types of “cells” that can make up neural networks.
- Autoencoders are a type of network/architecture that can be constructed with any of the types of neural network cells we covered previously

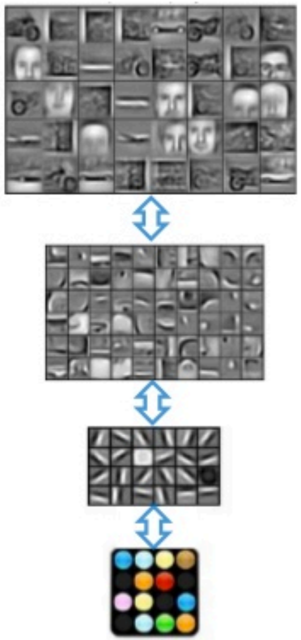
Meaning of Neural Net Architecture

As **hidden layers** increase : more complex interaction terms or more complex features learned

As **nodes/neurons** increase: number of “features” or dimensionality retained

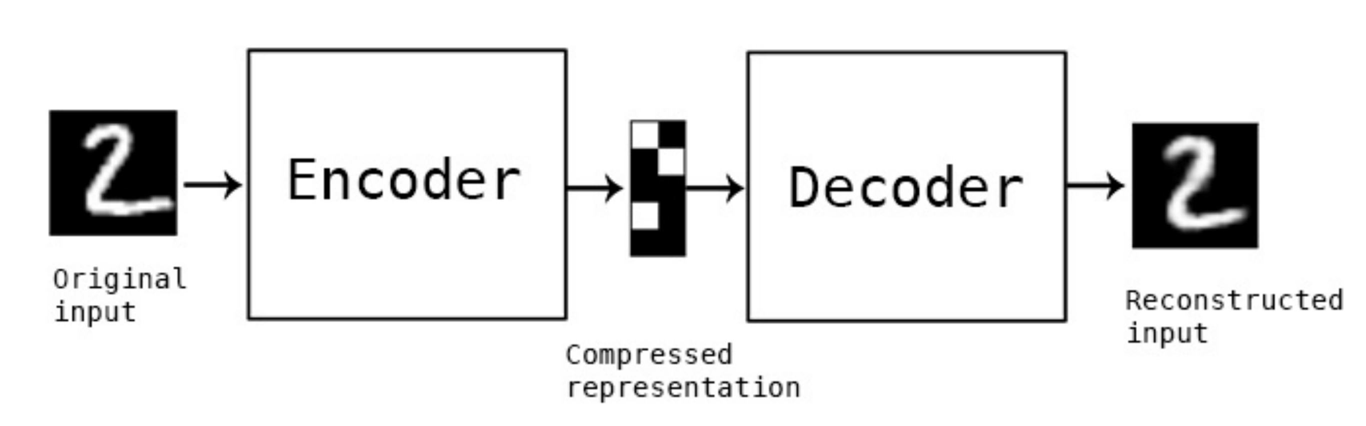
Learning Feature Hierarchy

- Deep Learning
  - Deep architectures can be representationally efficient.
  - Natural progression from low level to high level structures.
  - Can share the lower-level representations for multiple tasks.



Autoencoders

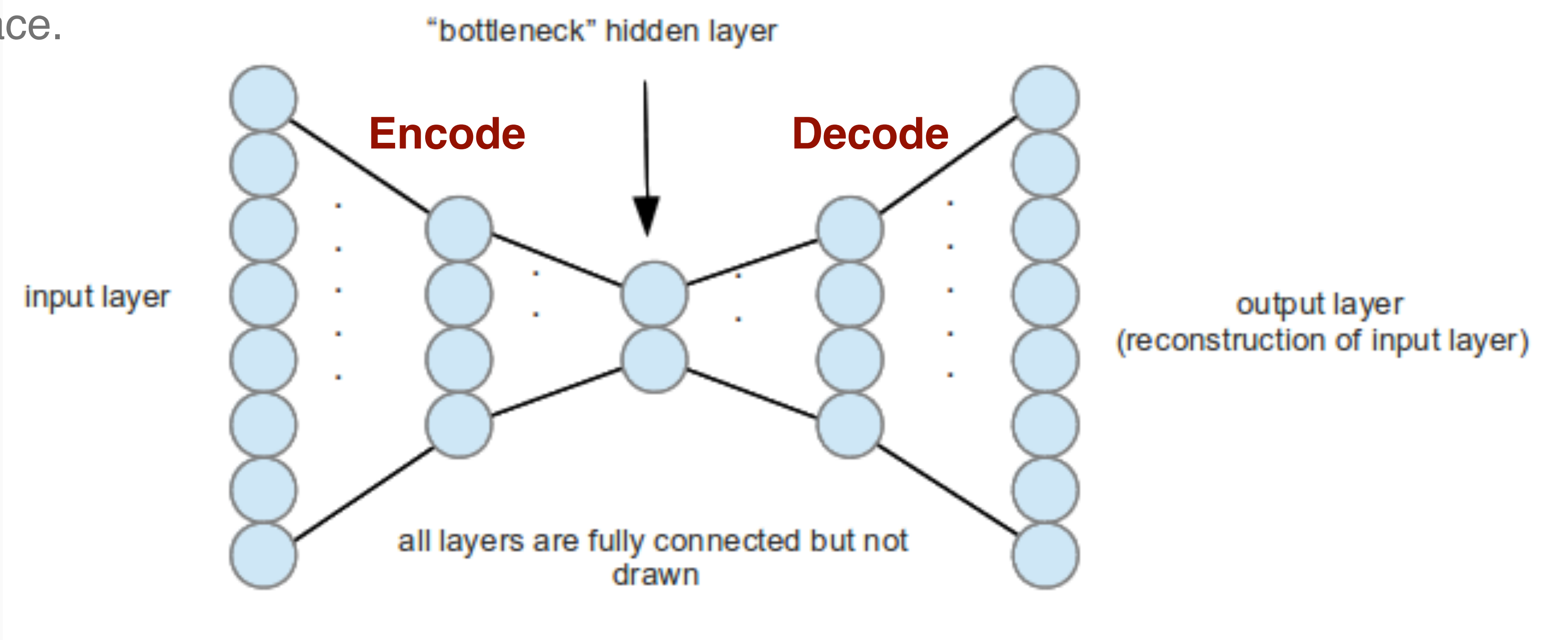
- Self-supervised deep learning (fit X vs X)
- Can be used with standard row & column data, images, sequences
- Model is trained to reconstruct the input as output
- Common first step in deep learning problems, helps model generalize inputs



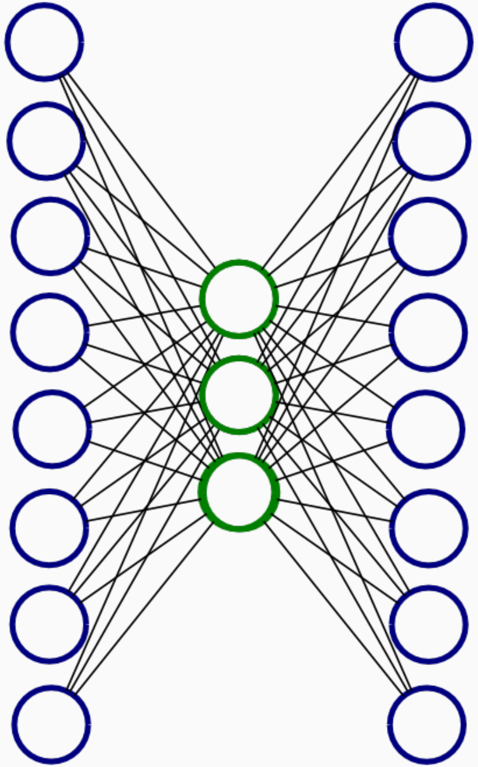
Autoencoders

galvanize

If all of your hidden layers had the same number of nodes, then the model would learn to just pass the input along at every layer. Autoencoders work by first decreasing the number of nodes, and then increasing back to the original feature space.



Yet another visual representation, but this one has a cat



Practical Considerations

- **Vanilla MLP**: None. You are in control of the number of nodes at each layer, so it is not complicated to decrease and then increase the sizes
- **RNN**s: You will need to use “return sequences” architecture and padding to keep your sequences at the same length throughout the process. Number of nodes can also be increased or decreased at each layer without practical issues.
- **CNN**s: In order to shrink your images down, you are either AGGRESSIVELY convolving them (big kernels with large strides), using pooling, or both. How to get them back up to the original size? Use a **conv2d_transpose**

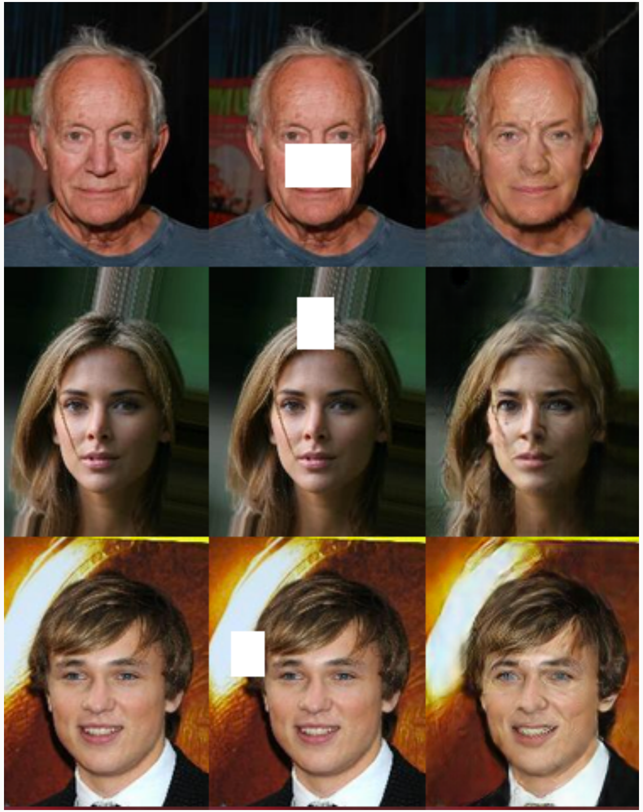
Varieties of Autoencoders (a subset..)

- Traditional - output is trained to reconstruct input
- Denoising - inputs have some noise added, output trained to reconstruct without the noise
- Sequence to Sequence - same as traditional, but with sequences (RNN layers)
- Variational - encoder has additional constraint: must generate latent vectors that roughly follow a unit gaussian distribution. Can use this feature to feed new vectors into the decoder and generate new data.

Applications (a subset)

- Traditional autoencoder: fill missing values, anomaly detection, can remove noise
- Machine translation
- Dimensionality reduction/embedding (encoder part)
- Creating disturbing images or funny text (GANs)
- For more on auto encoders for anomaly detection, check this out: https://www.slideshare.net/ds_mi/anomaly-detection-using-deep-autoencoders-gianmario-spacagna

Trained auto encoders can fill in missing data

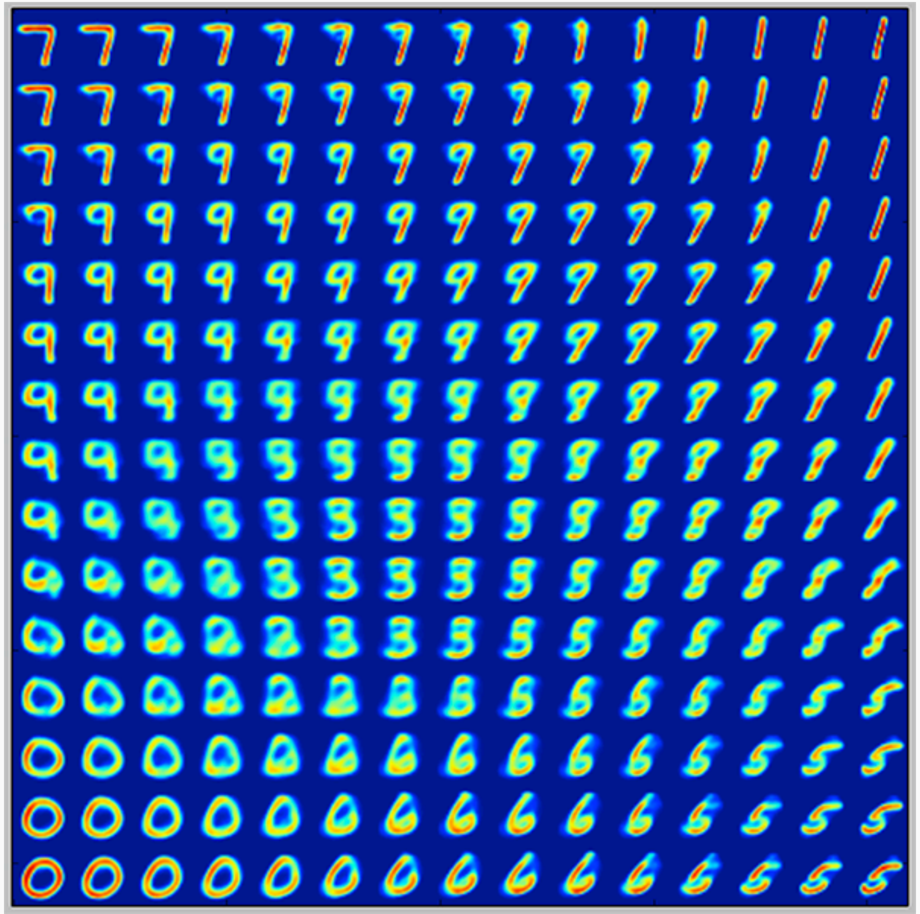


https://www.cc.gatech.edu/~hays/7476/projects/Avery_Wenchen/

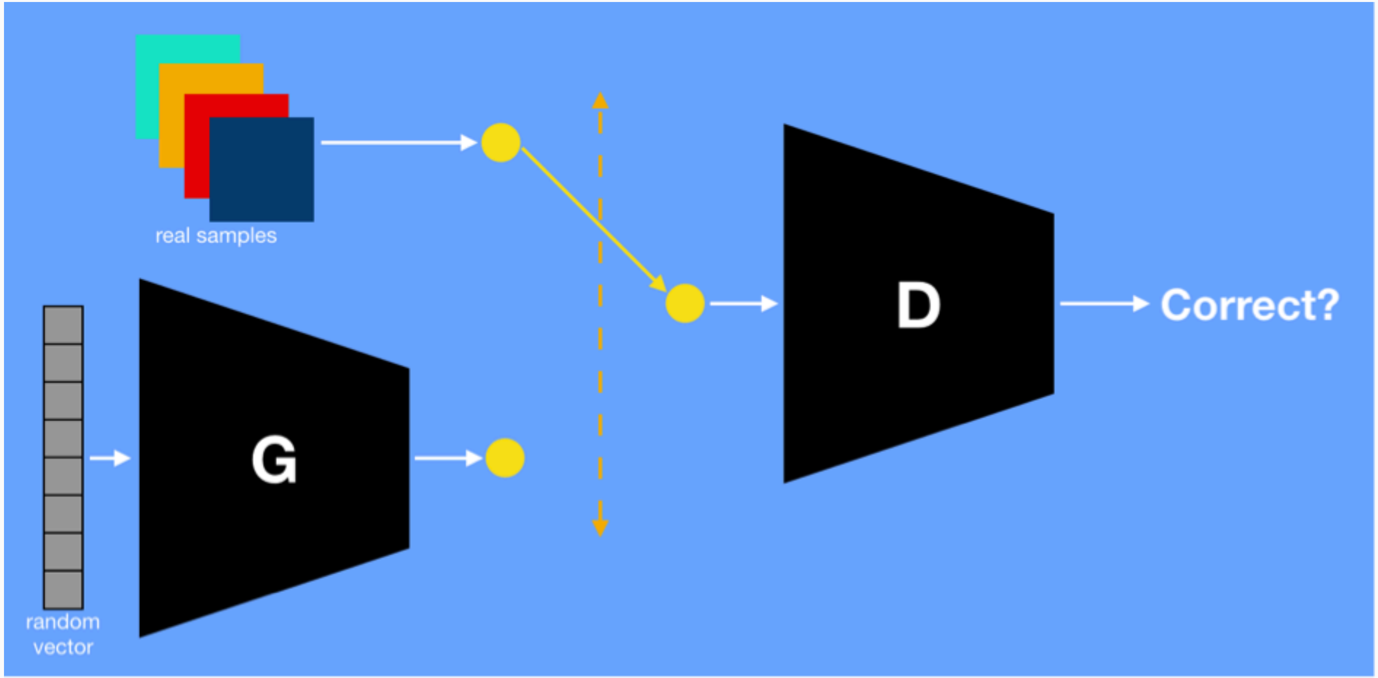
Denoising Autoencoders

https://blog.keras.io/building-autoencoders-in-keras.html

Variational Autoencoder on MNIST



Related: GAN (Generative Adverserial Network)



GAN with discriminator from
https://www.kdnuggets.com/2018/01/generative-adversarial-networks-overview.html#.Wl04qlFzquo.linkedin

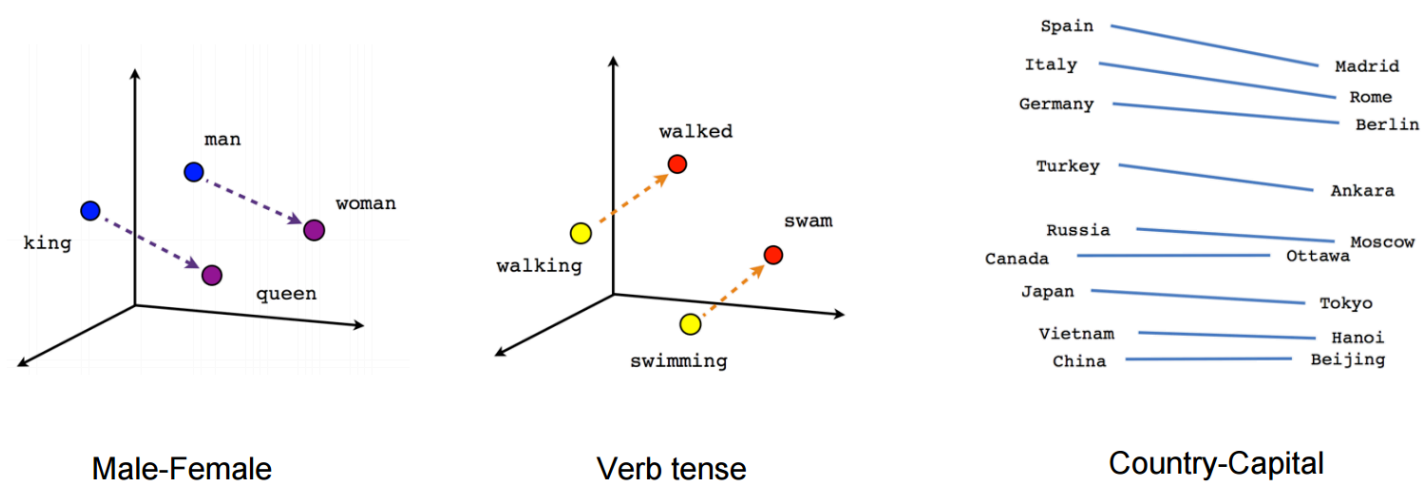
Autoencoders can also induce nightmares

galvanize

Made with: https://deepdreamgenerator.com/

Very related topic…embedding

- Embedding layers take inputs in high dimensional space (typically text, but not always) and learn representations of these inputs in lower dimensions
- One very popular example is a model called **word2vec**



https://medium.com/towards-data-science/deep-learning-4-embedding-layers-f9a02d55ac12

Loss Functions

- For basic, vanilla autoencoders, two options: MSE or binary cross entropy, pixel wise
- For your assignment today, you will be working with the MNIST database. The code you are given uses MSE (for pixel values). Can also use pixel-wise cross entropy. (your error will be on a wildly different scale, but this may work better as a loss function - this is a hyperparamter you can to tune!)
- For more advanced autoencoders (i.e. anything that is NOT just reproducing the input image), your loss function will vary based on use case!

Recap: Learning Objectives

galvanize

- ✓ **Review** previous neural network cells and architectures
- ✓ **Understand** autoencoders and their many uses in machine learning
- ✓ Be ready to **tune** a fully connected and then a convolutional autoencoder

Resources

http://kvfrans.com/variational-autoencoders-explained/

https://www.kdnuggets.com/2018/01/generative-adversarial-networks-overview.html#.Wl04qlFzquo.linkedin

https://blog.keras.io/building-autoencoders-in-keras.html