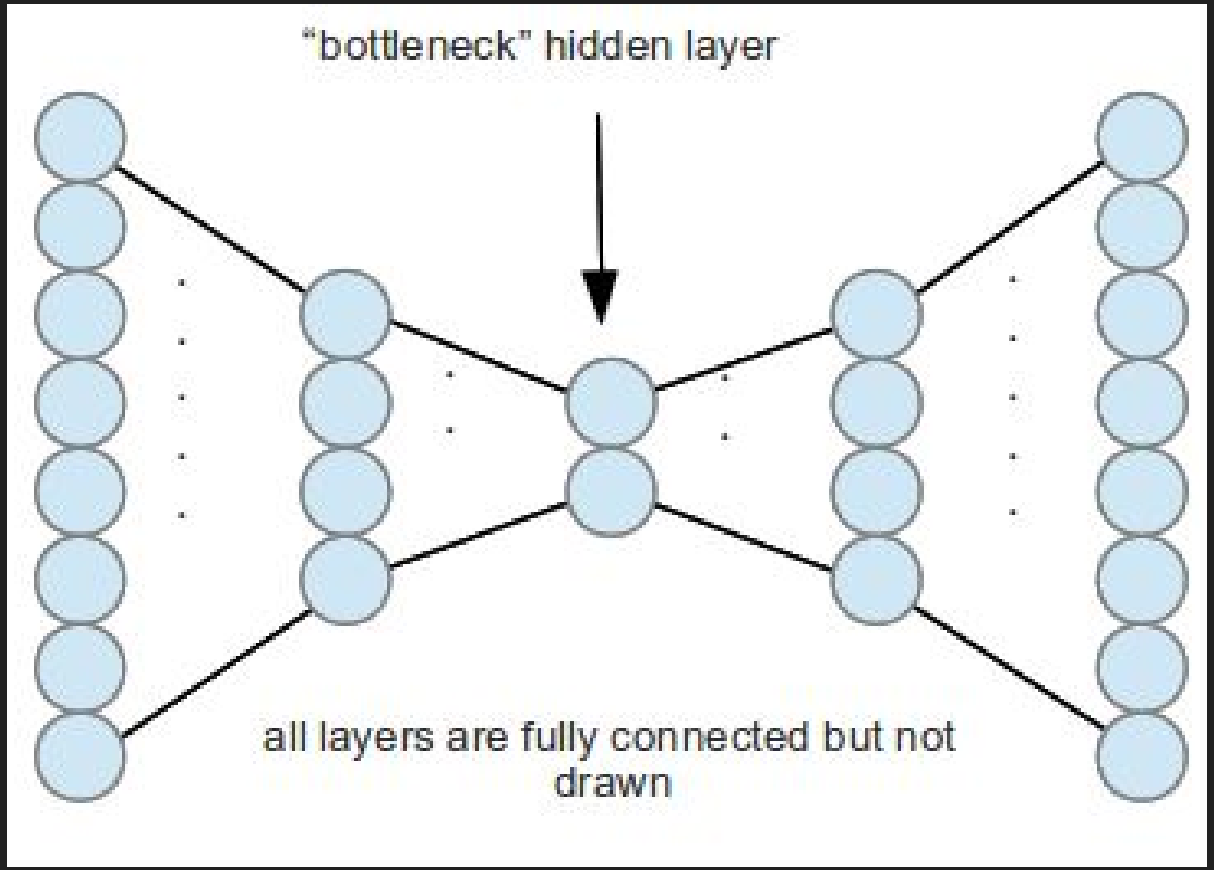
Auto Encoders

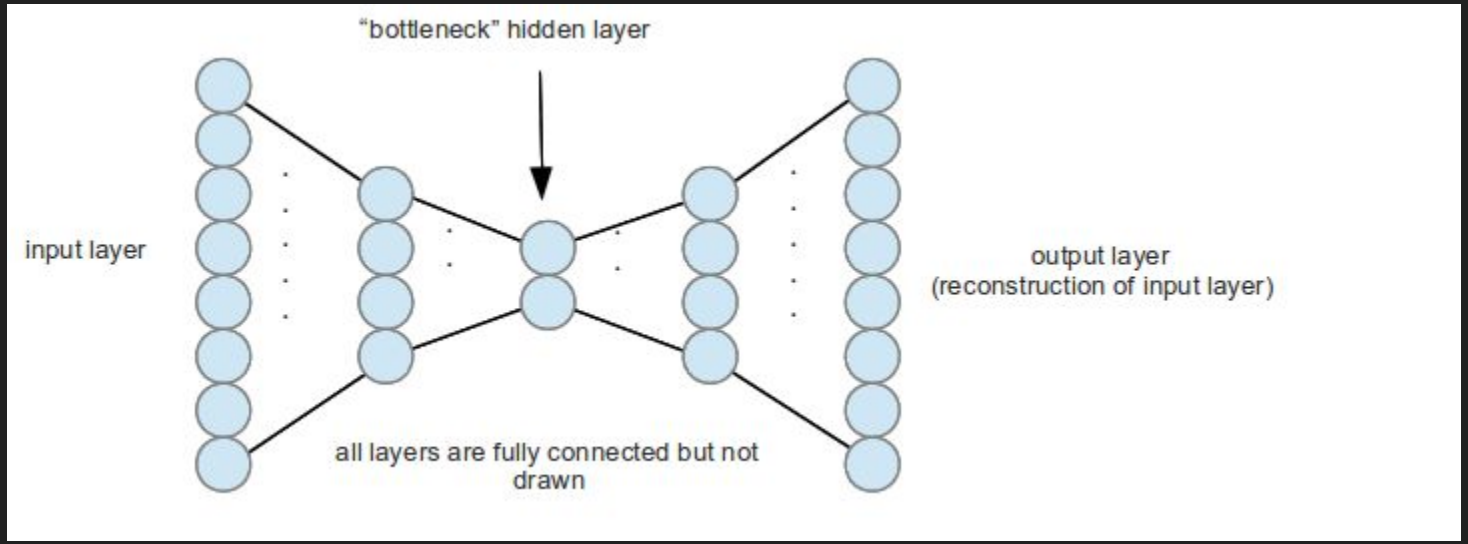
What function is an autoencoder learning?

- Auto encoders learn the identity function
- F(x) = x
- BUT they have fewer nodes in the middle


"bottleneck" hidden layer
all layers are fully connected but not drawn

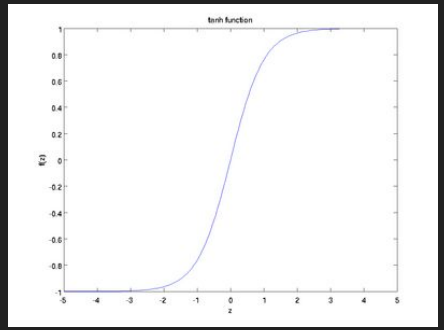
Auto Encoders

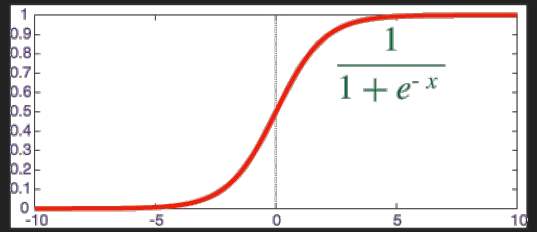
- This creates a compression of the data in the middle
- What's in the compressed form?



First, let's talk about neuron activations

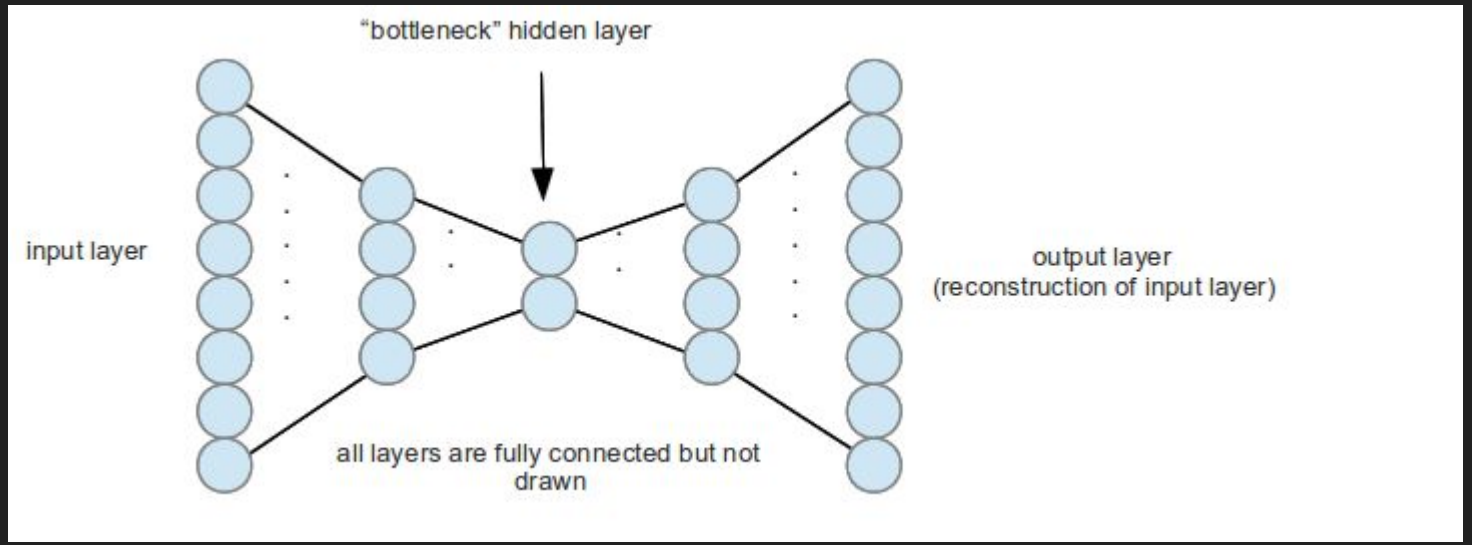
- Activation functions will turn the neuron "on" or "off"
- Easiest way to think about it is that 0 means the feature is not present, 1 means feature is present
- These turned on neurons are what the network "knows"
- Some activation functions are good at "on" or "off", but 0 the gradients when in the on/off positions, so they're hard to train

Tanh = 

Sigmoid = 

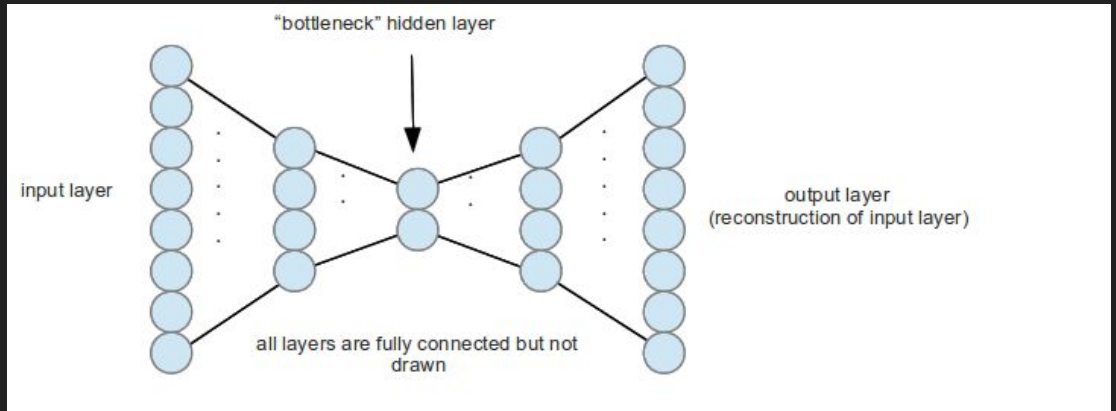
Back to the autoencoder

- Auto encoders will (in theory) learn to activate every neuron if it is perfect, lossless compression



Auto encoder example

- 100 inputs
  - 20 independent numbers
  - 80 made from linear combinations of the 20
    - So the 80 are dependent and have no new information
- Represent the 100 inputs with 20 hidden neurons in the middle

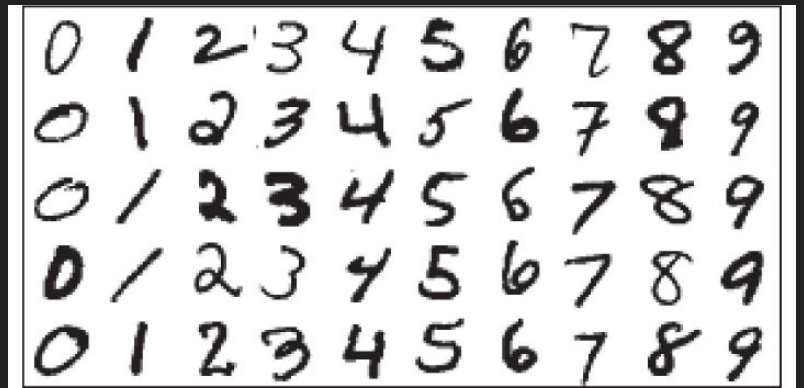


Variations/Uses of the Auto-encoder

- Sparse autoencoder
- Denoising autoencoder
- Pretraining with autoencoders
- Google Translate

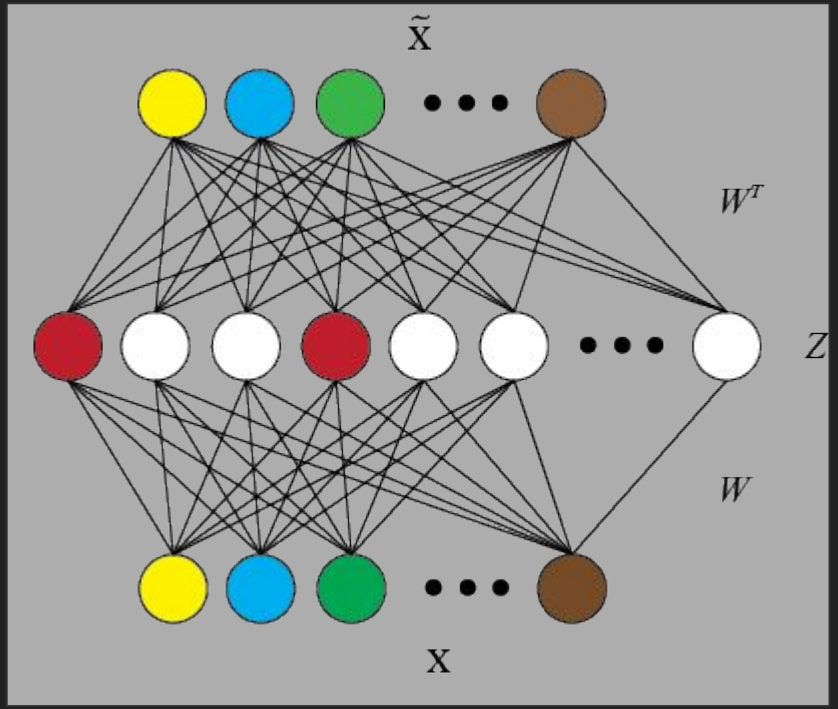
Sparse Autoencoders

- We will specifically discuss the k-sparse Autoencoder
  - Other variations also exist
- In this example, we will teach an autoencoder to recognize handwritten digits from the MNIST dataset
  - Most classic deep learning dataset
  - Each number is 20x20 pixels
  - 400 input features to network
    - (400 total pixels)

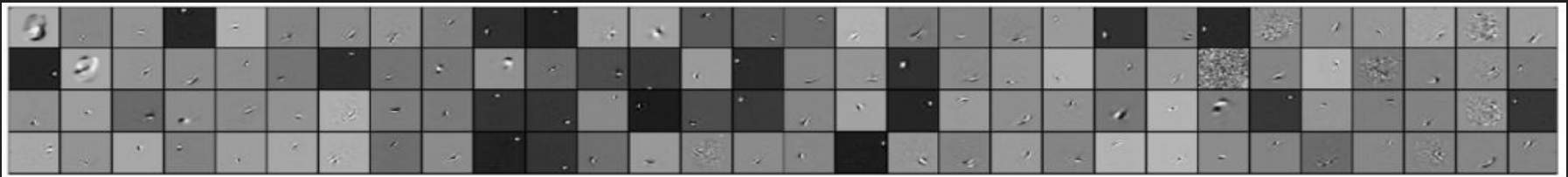


k-Sparse Autoencoders

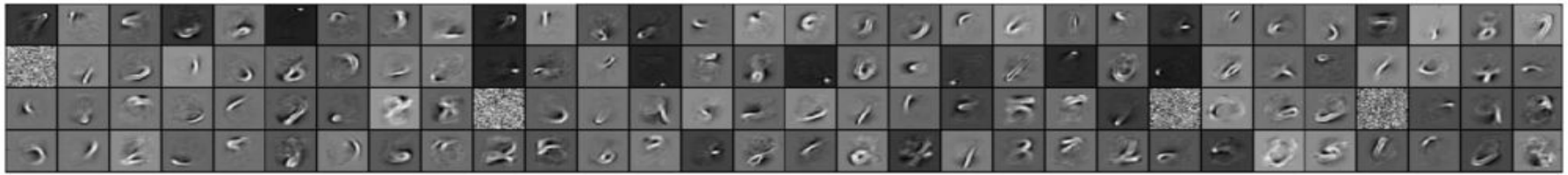
- X is the input digit as 400 features
- X-tilde is the reconstructed input
- Z is our hidden layer
- We take k neurons from z
  - S.t. that k has the highest activated values
- This specializes the neurons
  - (similar to dropout)



Each square shows which pixels “activated” a neuron



(a) $k = 70$



(c) $k = 25$

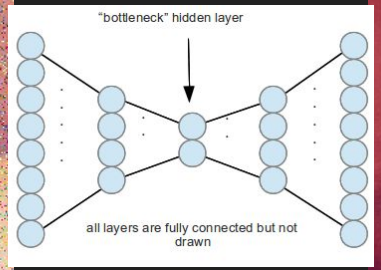
(d) $k = 10$

Denoising Autoencoders

- What if we want to de-corrupt data?
- H(x) is a corrupting function
  - Introduces noise to input
- We want our neural network function F to learn the uncorrupted input
  - F(H(x)) = x

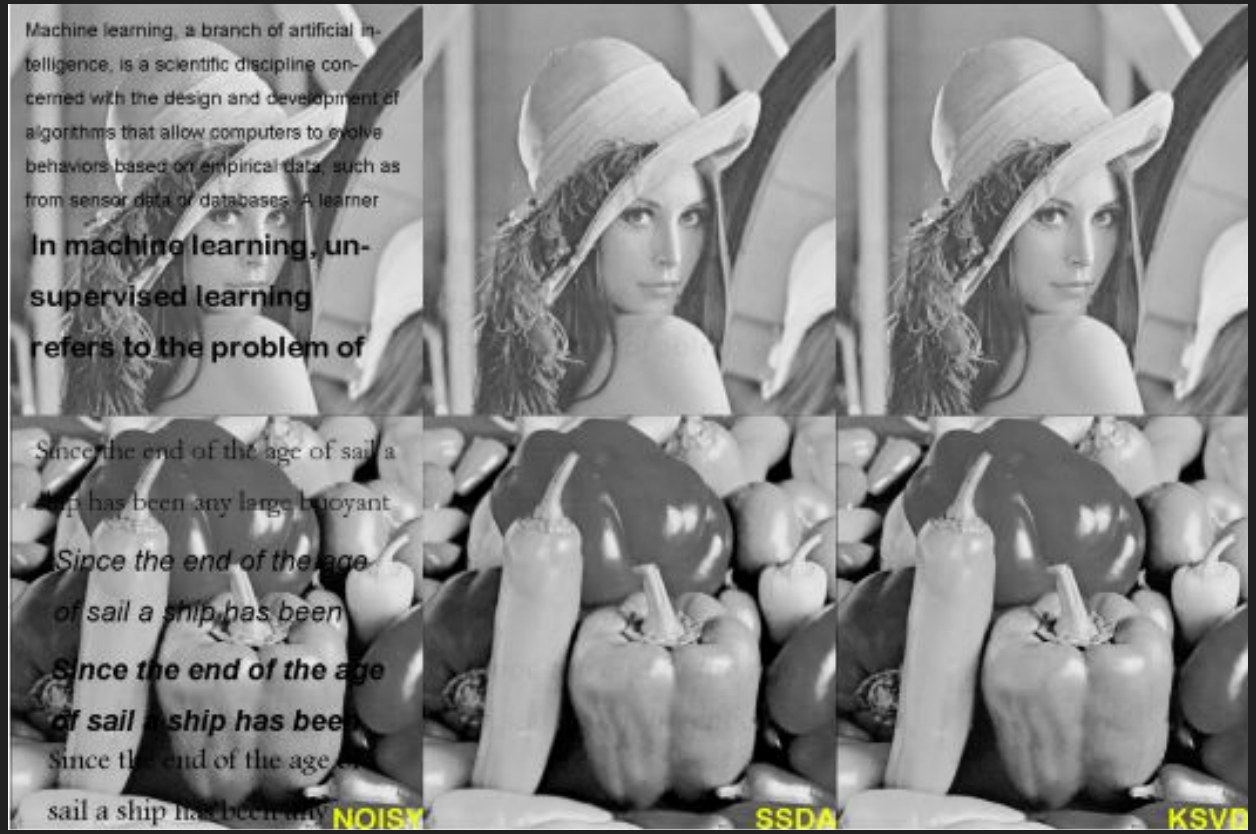
Denoising Autoencoders

H(x)



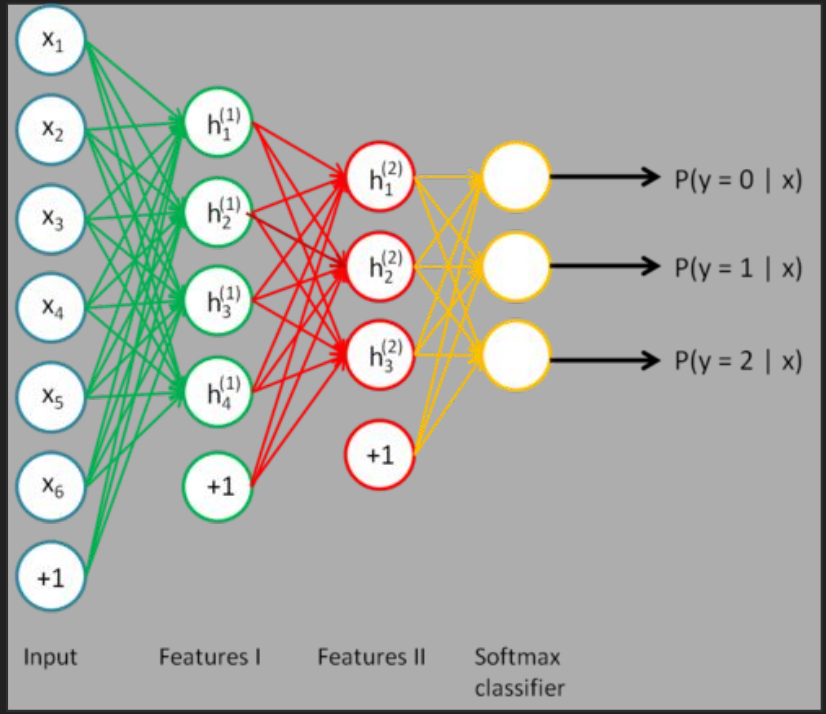
F(H(x)) = x

How well does a denoising autoencoder do?



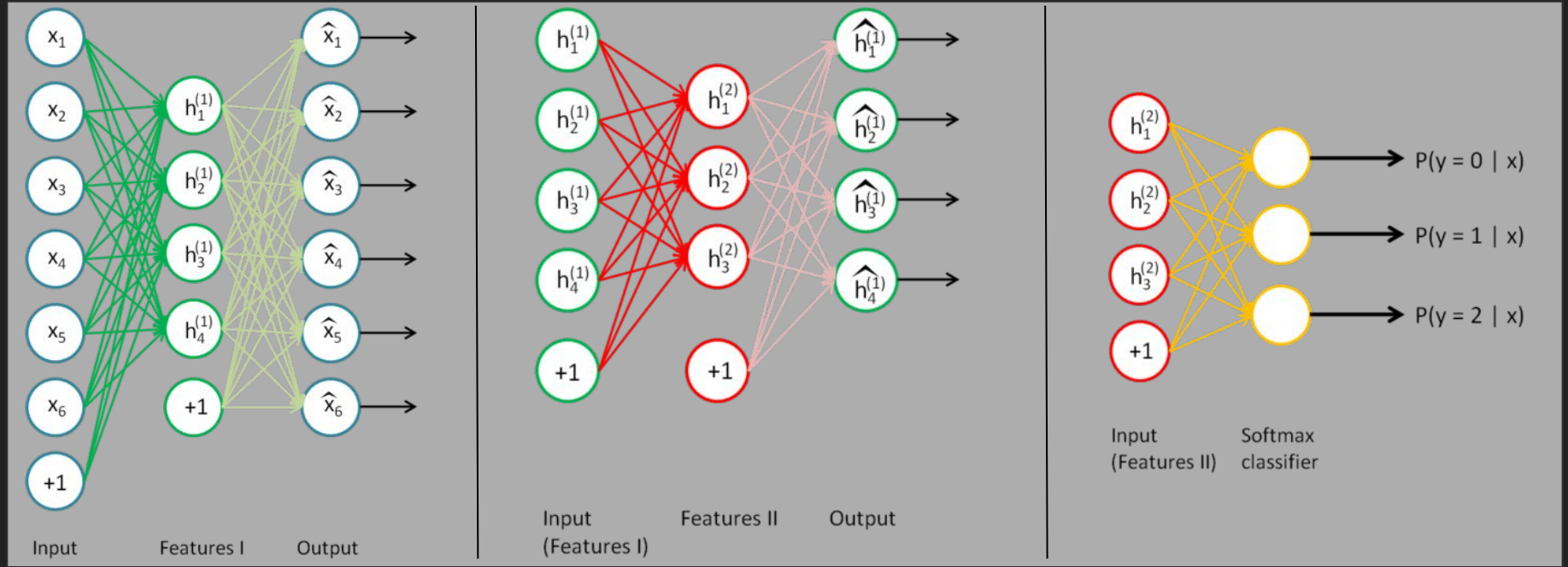
Pretraining with autoencoders

- Problem: Some networks never begin learning at all!
  - Because their weights are poorly initialized
- We can use autoencoders to “pretrain” a network so that it starts its learning as fast as possible



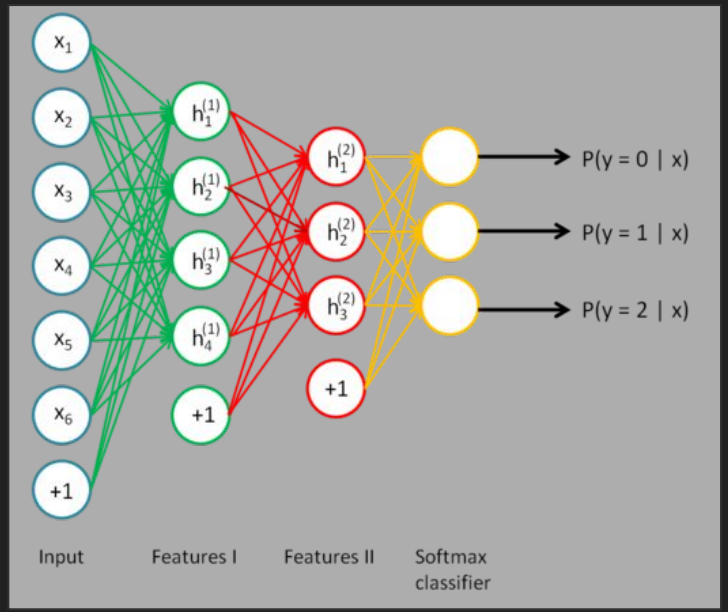
Pretraining with Autoencoders

- So to pretrain this network, we autoencode each successive layer and train these autoencoders one at a time



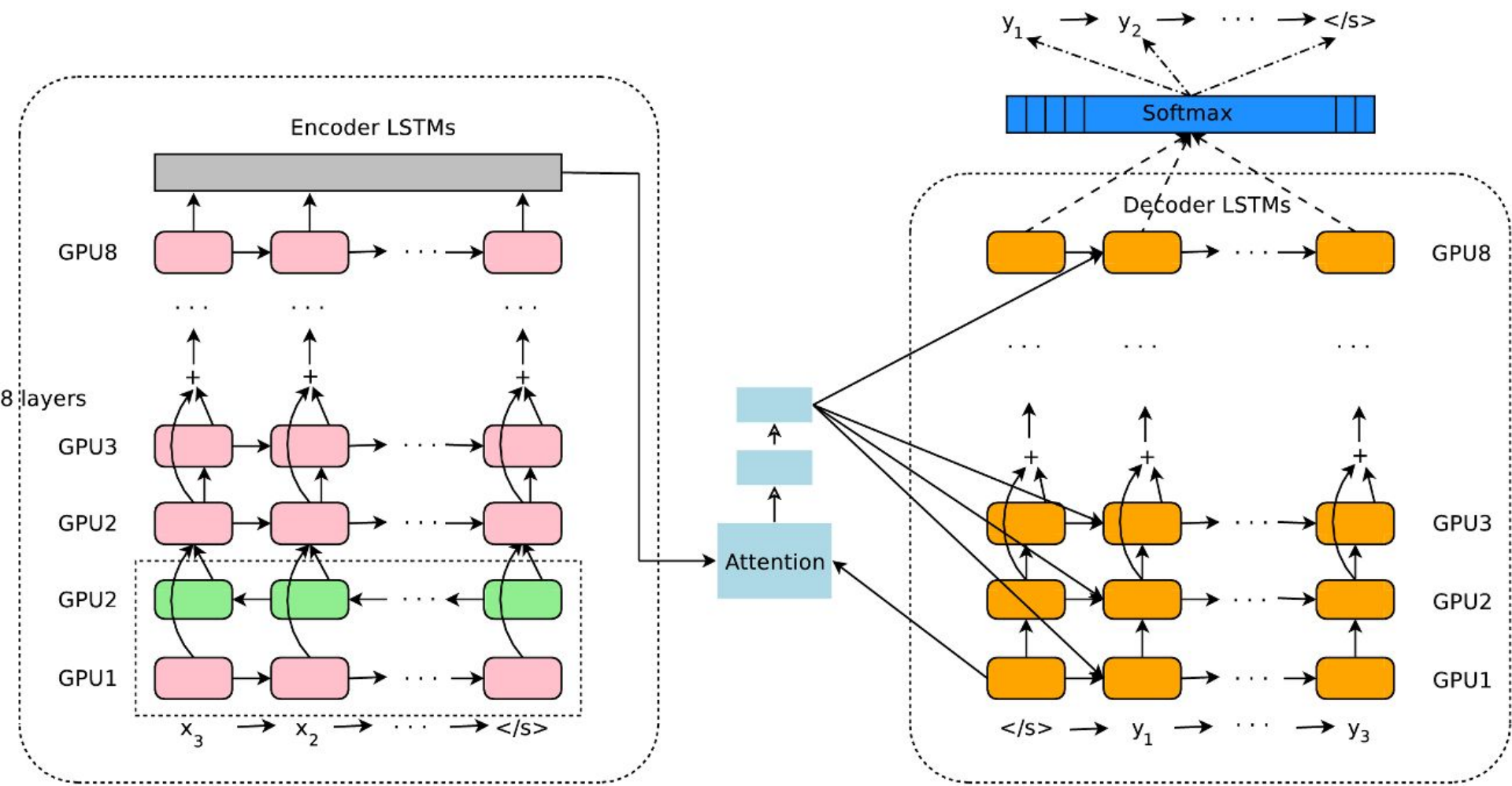
Pretraining with Autoencoders

- Once these autoencoders are pretrained, we initialize the network with the weights of all of the autoencoders put together
- This has been shown to greatly increase the rate of learning for the entire network



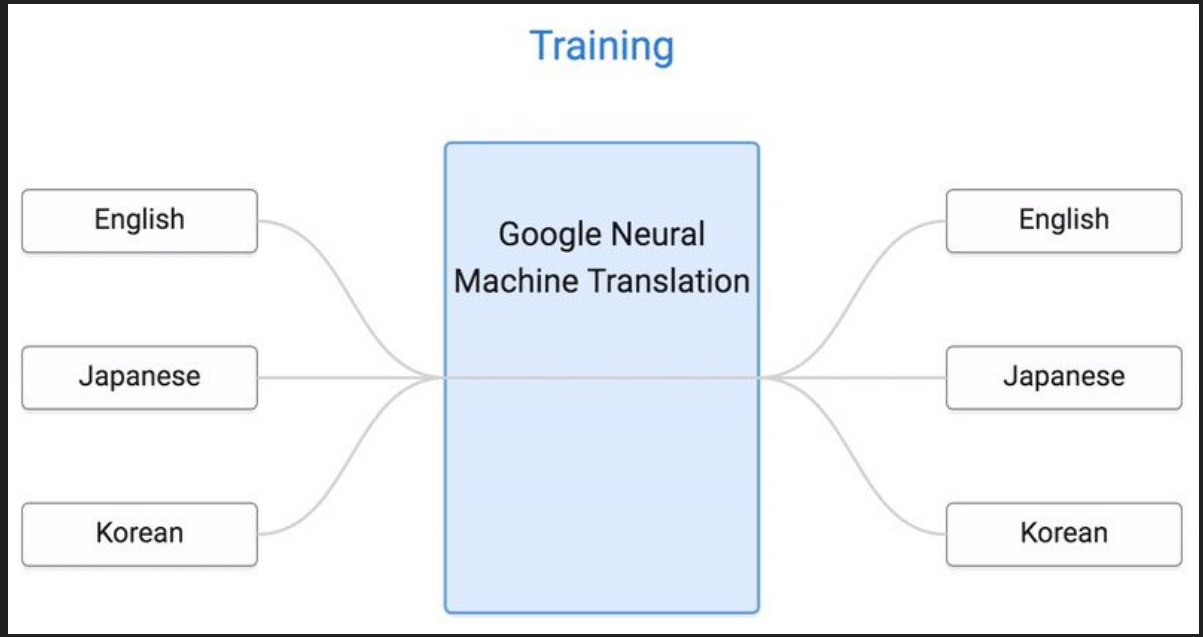
Lastly, Google Translate

- It uses autoencoders!
  - In association with LSTMs, but we'll discuss those later
  - Blog post and official article for those interested


Encoder LSTMs
GPU8
8 layers
GPU3
GPU2
GPU2
GPU1
x3
x2
</s>
Attention
y1
y2
</s>
Softmax
Decoder LSTMs
GPU8
GPU3
GPU2
GPU1
</s>
y1
y3

This model can even translate unseen language pairs!

- Teach it to translate between korean/japanese and English
- Test korean to japanese or japanese to korean (which it hasn't ever trained on)
- And it works!



Next week

- Convolutional neural networks
  - Used in every image processing task