Autoencoders

Review of Unsupervised Learning

Deep Learning

* Let’s quickly review what Unsupervised Learning is and the unique challenges it presents!
* Recall that Supervised Learning works with **labeled** data, giving us access to a direct check for errors for our models.

Deep Learning

- Most of the world's data is unlabeled!
- New articles, movie reviews, user Netflix ratings, images on the internet, etc...
- What approaches can we take to make use of this type of **unlabeled** data?

Deep Learning

- Two main approaches
    - Dimensionality Reduction
    - Clustering

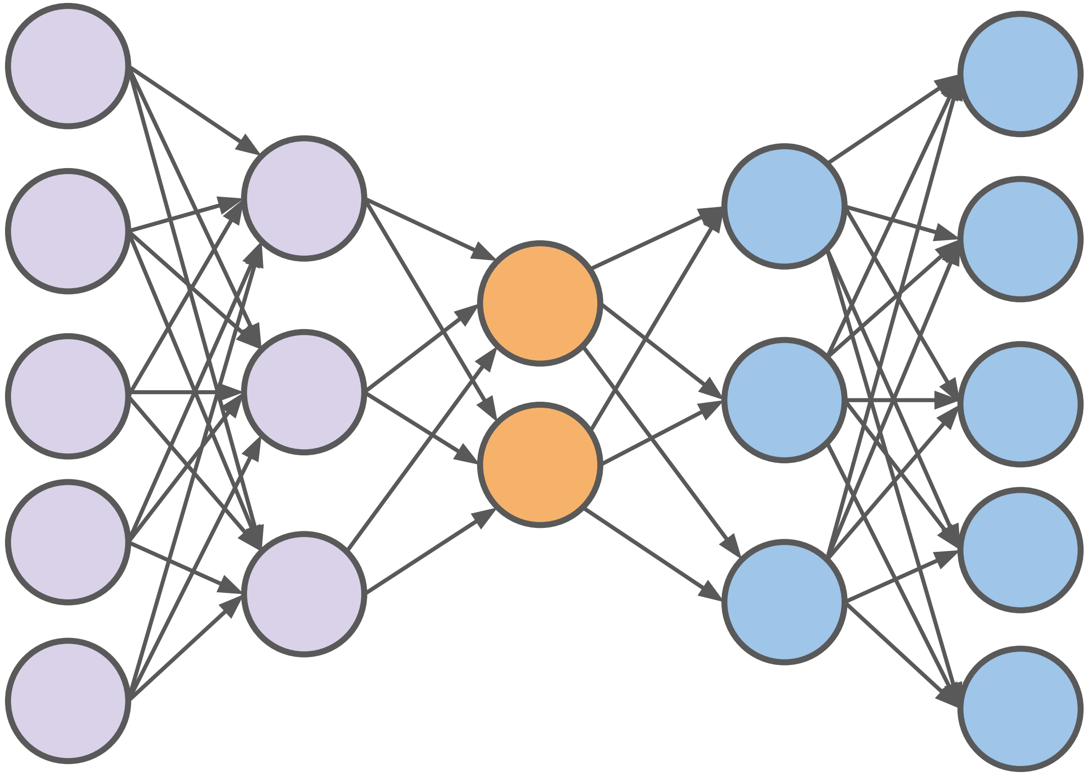
Autoencoder Basics

Deep Learning

- The autoencoder is actually a very simple neural network and will feel similar to a multi-layer perceptron model.
- It is designed to reproduce its input at the output layer.

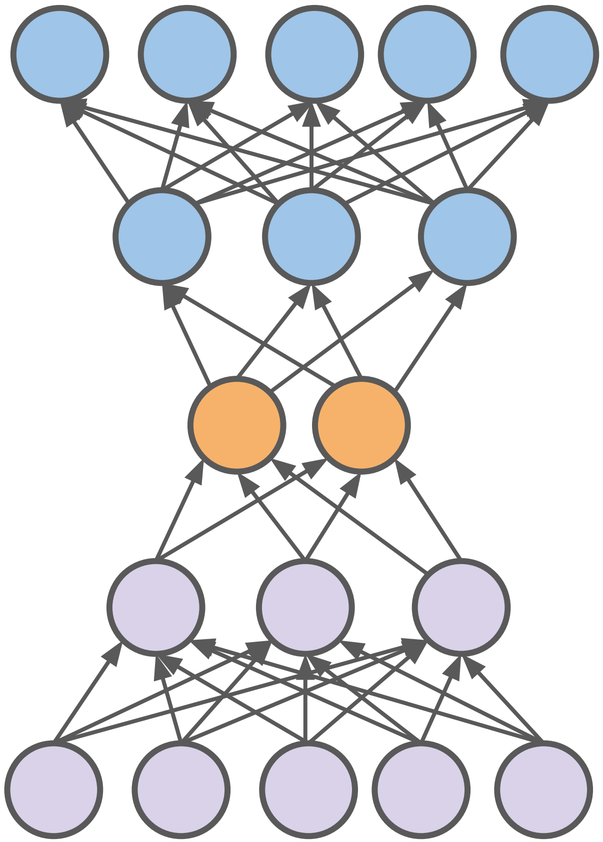
- The key difference between an autoencoder and a typical MLP network is that the number of input neurons is equal to the number of output neurons.
- Let's explore what this looks like and why we would use it!

Example Autoencoder



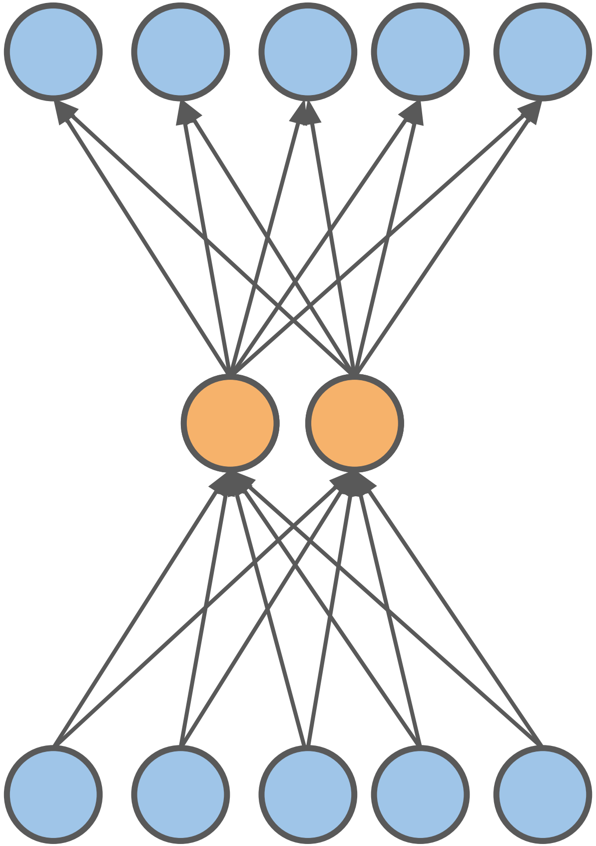
Deep Learning

- Let's walk through the layers and explain the basic idea of an autoencoder.



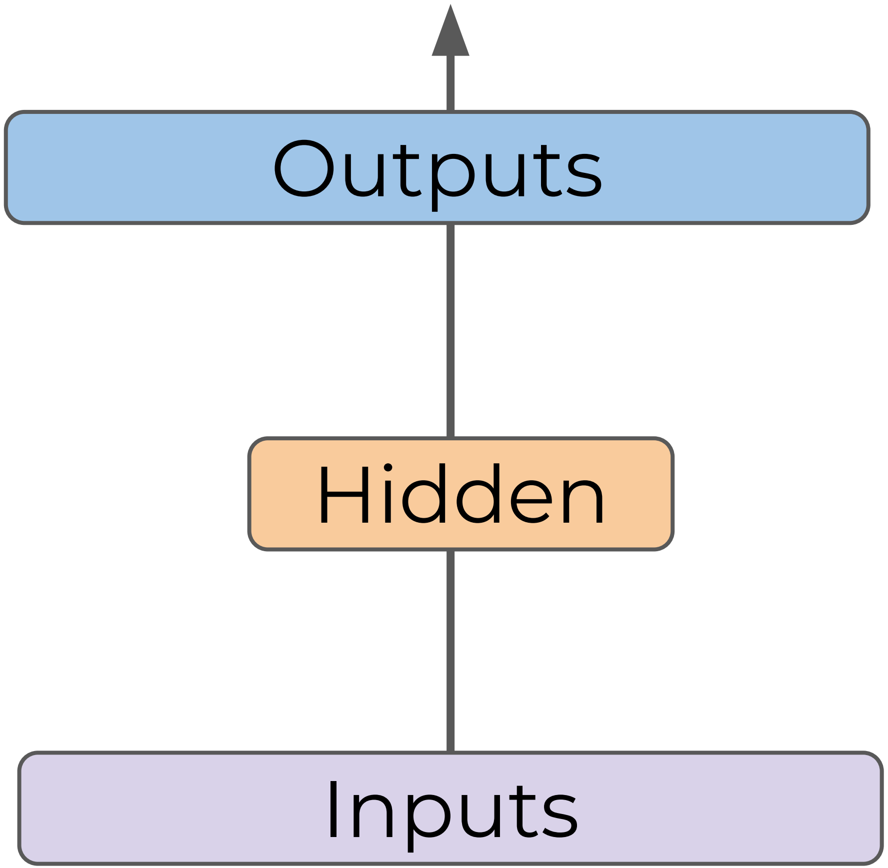
Deep Learning

- Let's simplify this even further with just a single hidden layer.



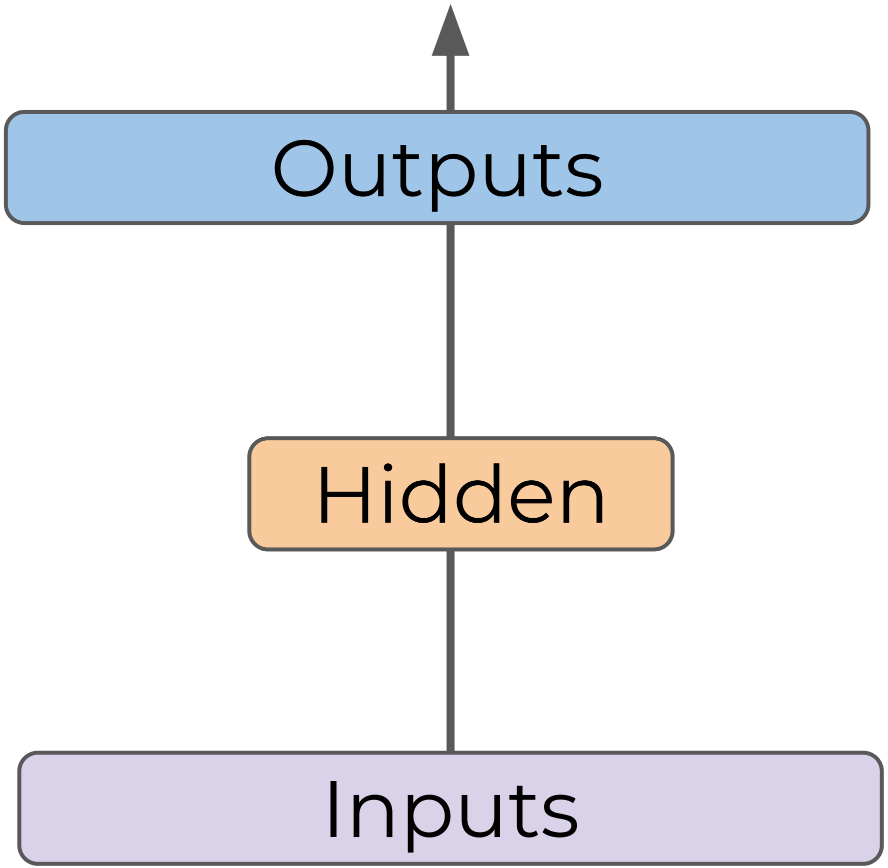
Deep Learning

- Let's simplify this even further with just a single hidden layer.



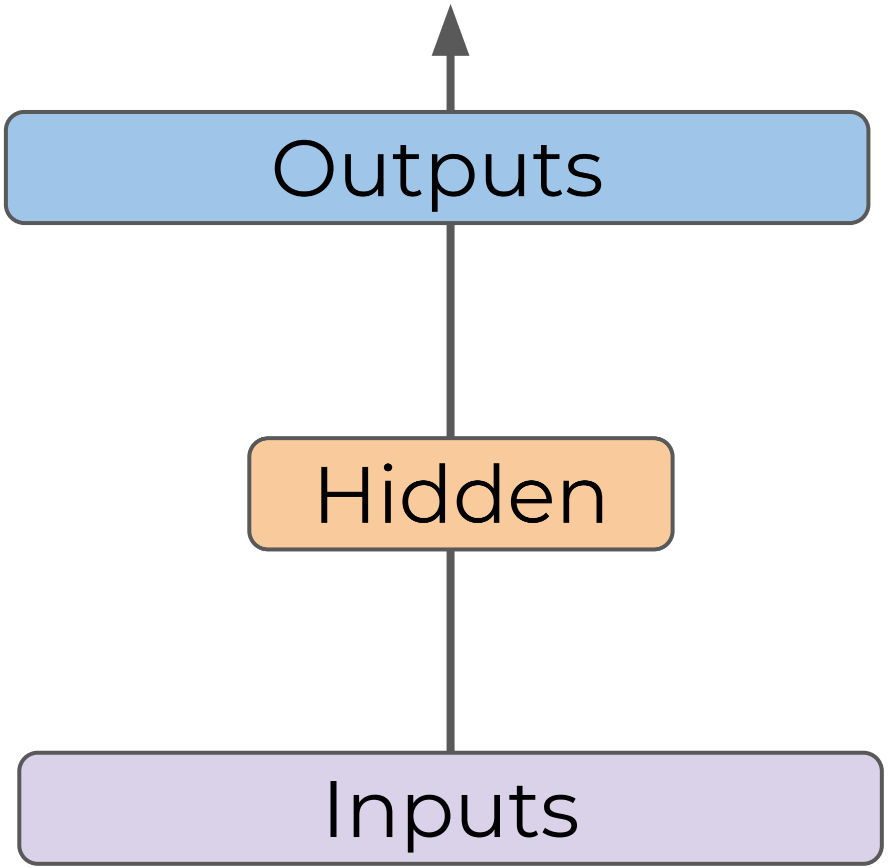
Deep Learning

- Feed forward network trained to reproduce its input at the output layer.



Deep Learning

- Output size is the same as the input layer.

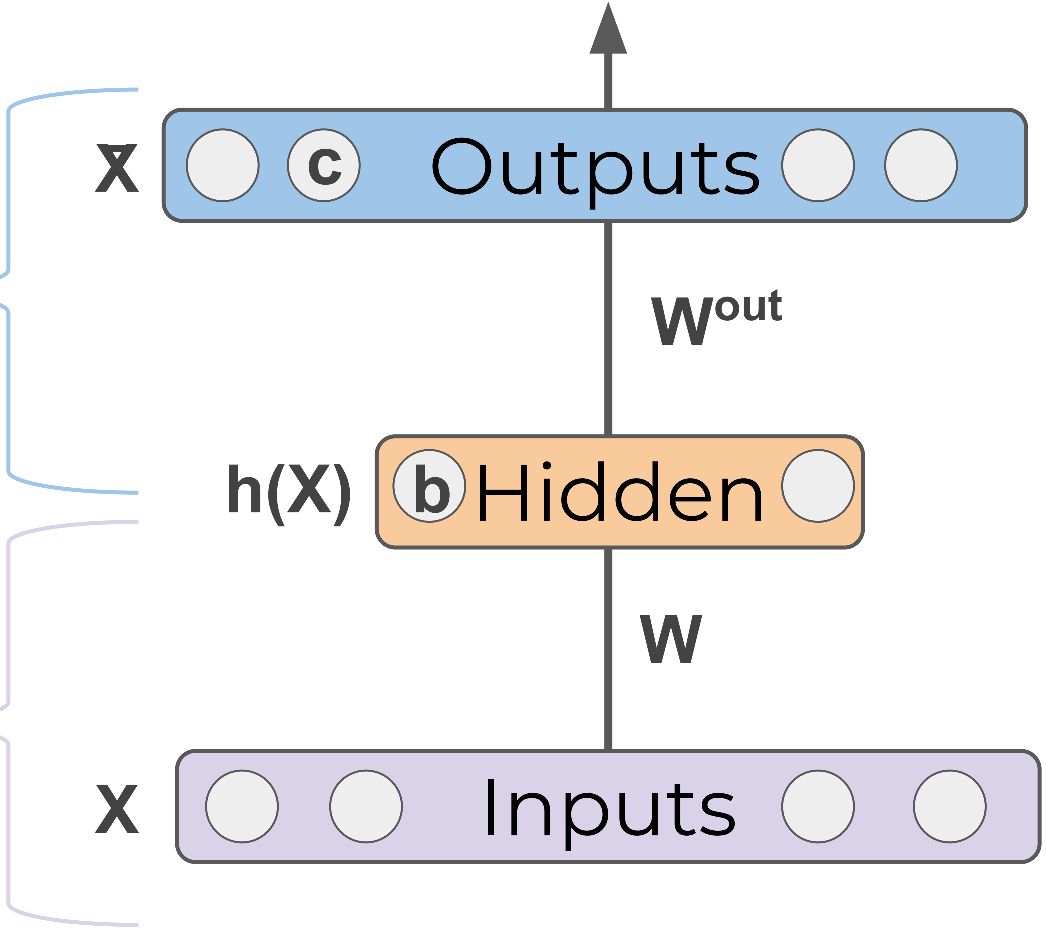


Decoder

$\bar{X} = \mathrm{sigm}(c + W^{out}\, h(x))$

Encoder

$h(x) = \mathrm{sigm}(b + Wx)$



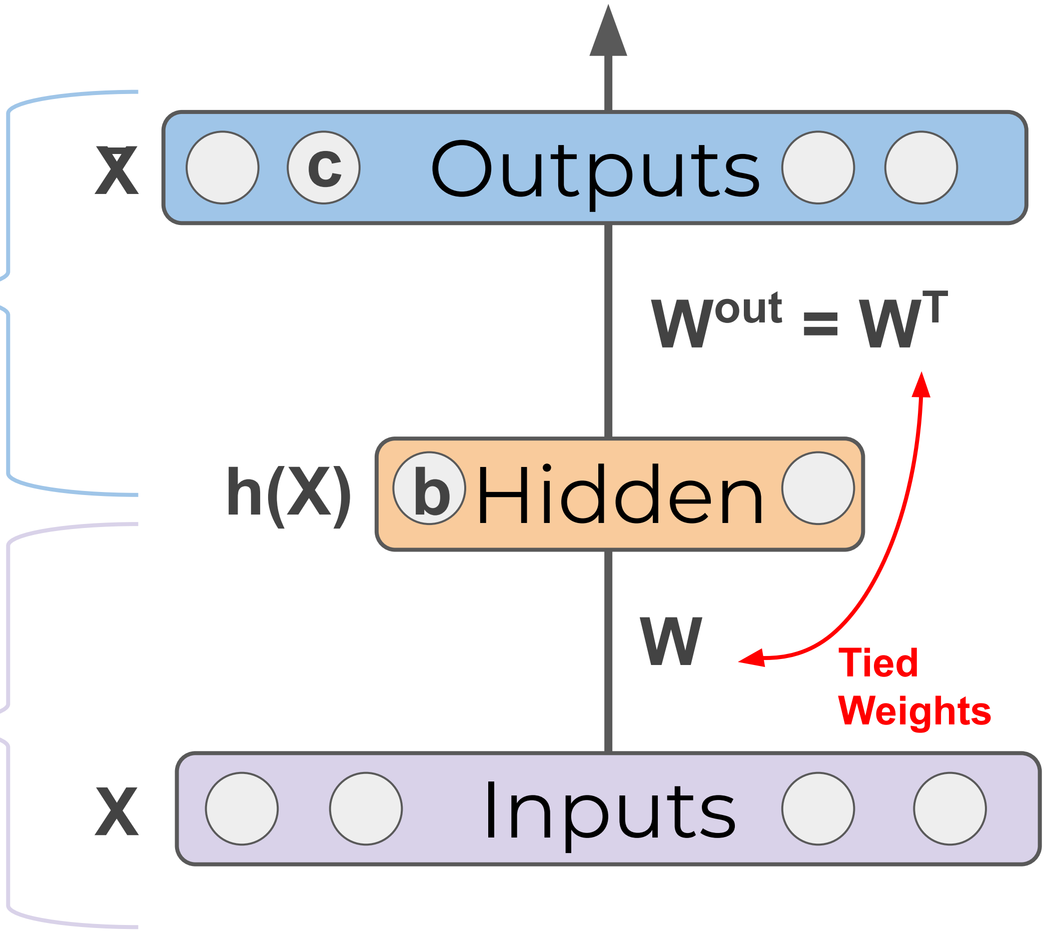
Deep Learning

Decoder

$\bar{X} = \mathrm{sigm}(c + W^{out}\, h(x))$

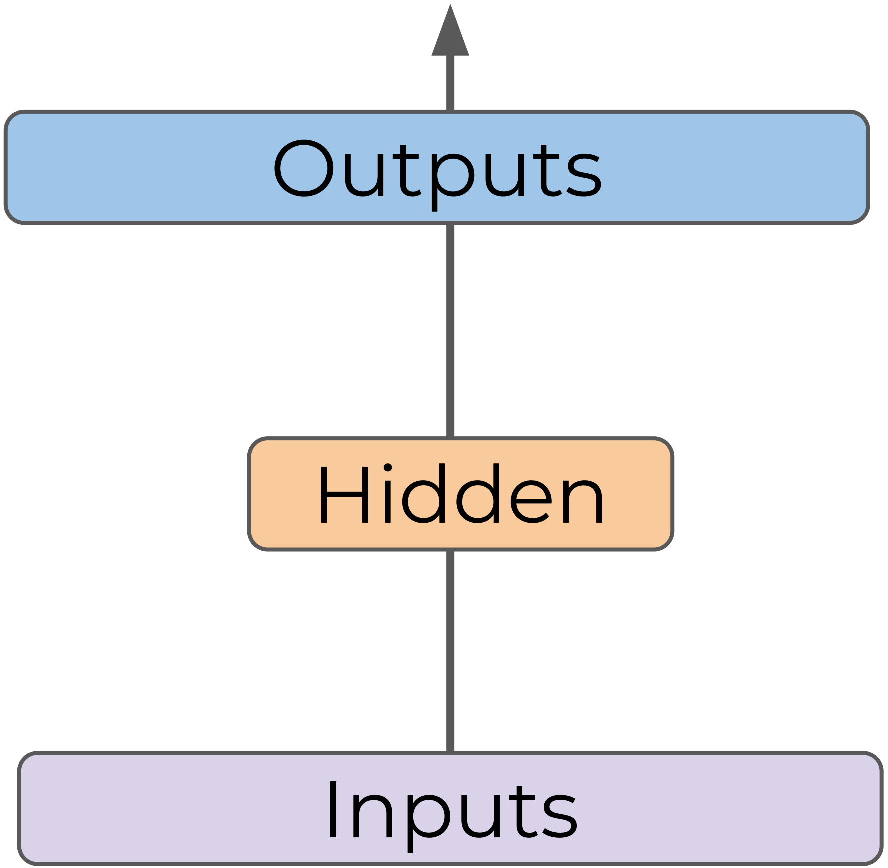
Encoder

$h(x) = \mathrm{sigm}(b + Wx)$



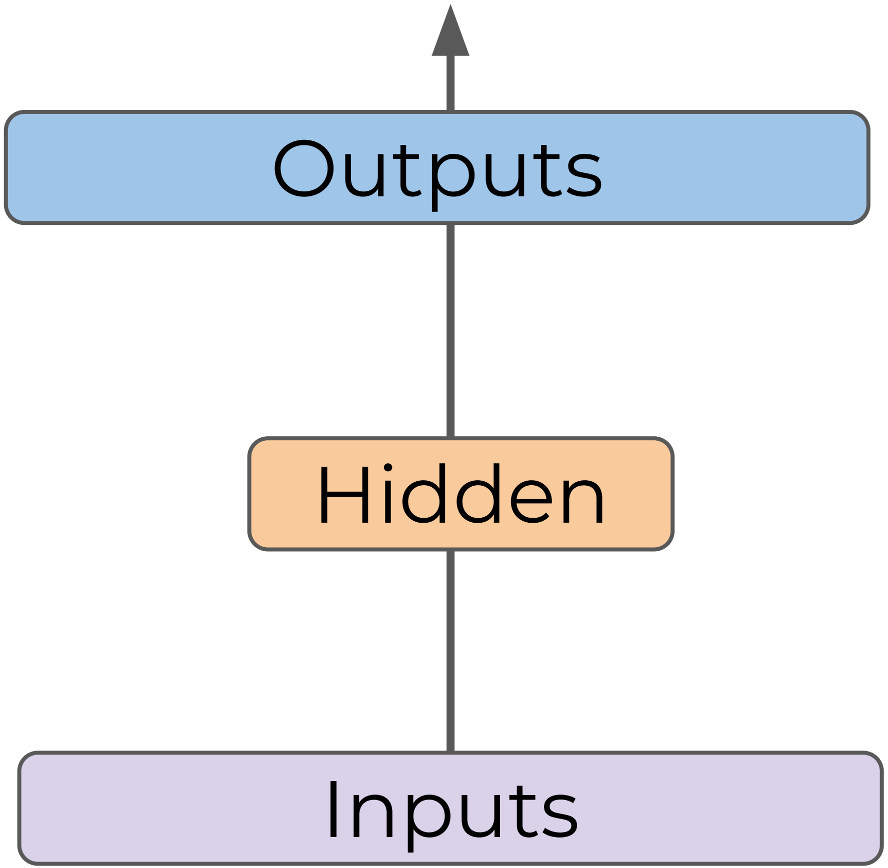
Deep Learning

- The hidden/internal representation maintains all the information of the input.



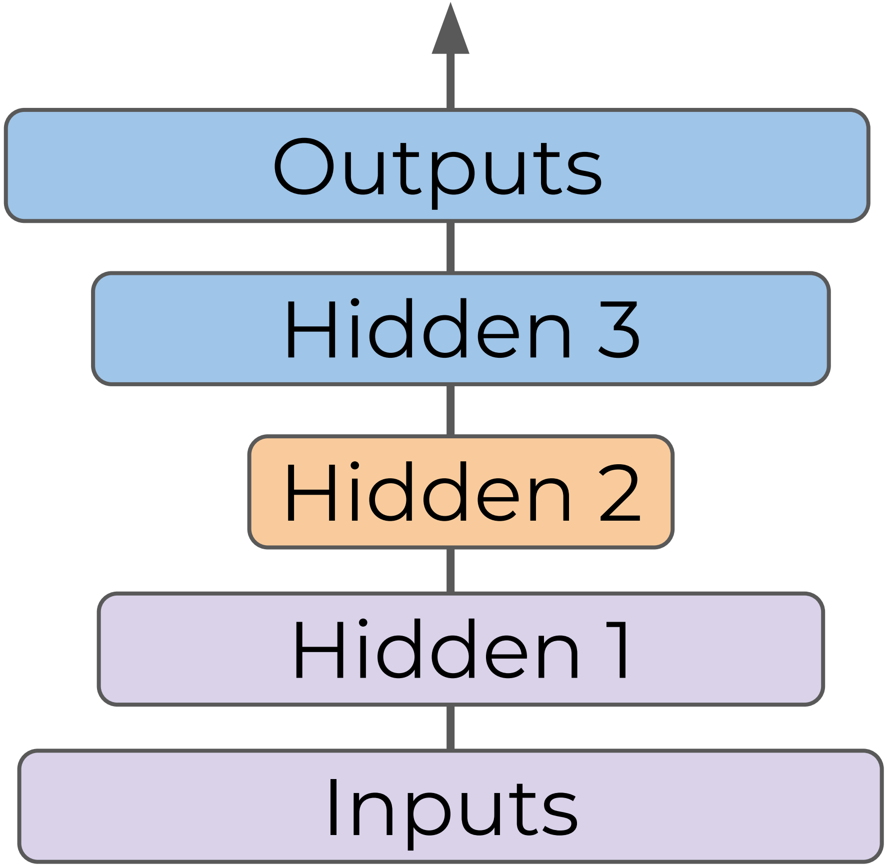
Deep Learning

- We can use the hidden layer to extract meaningful features, we will also explore PCA with Autoencoders.



Deep Learning

- Later on we'll explore stacked autoencoders with more hidden layers.



Deep Learning

- You should know understand a very basic autoencoder.
- Up next we'll discuss how we can use an Autoencoder to perform dimensionality reduction.

Dimensionality Reduction with Linear Autoencoder

- Linear Autoencoders can be used to perform a principal component analysis, which allows us to reduce the dimensionality of our data.

- Dimensionality reduction allows us to get a lower dimension representation of our data.
- The encoder creates new (fewer) features from the input features.

Deep Learning

- For example we can input a 3 dimensional data set and output a 2 dimensional representation of it.
- Keep in mind, we are not simply choosing 2 of the previous 3, but instead constructing 2 new features from the 3.

Deep Learning

- We achieve this effect by using a linear autoencoder.
- Linear autoencoders perform the linear transformations by only using the weights and bias terms.

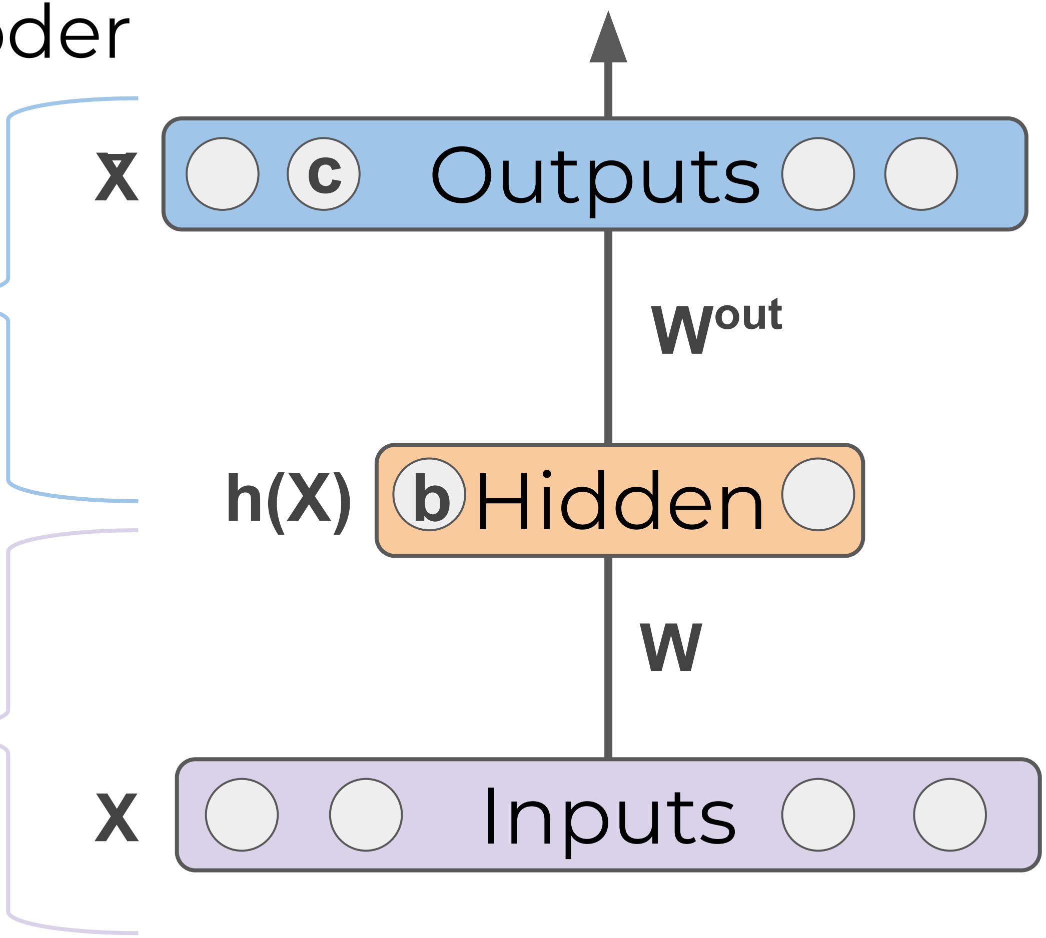
Linear Autoencoder

Decoder

$\bar{X} = c + W^{out} h(x)$

Encoder

$h(x) = b + Wx$

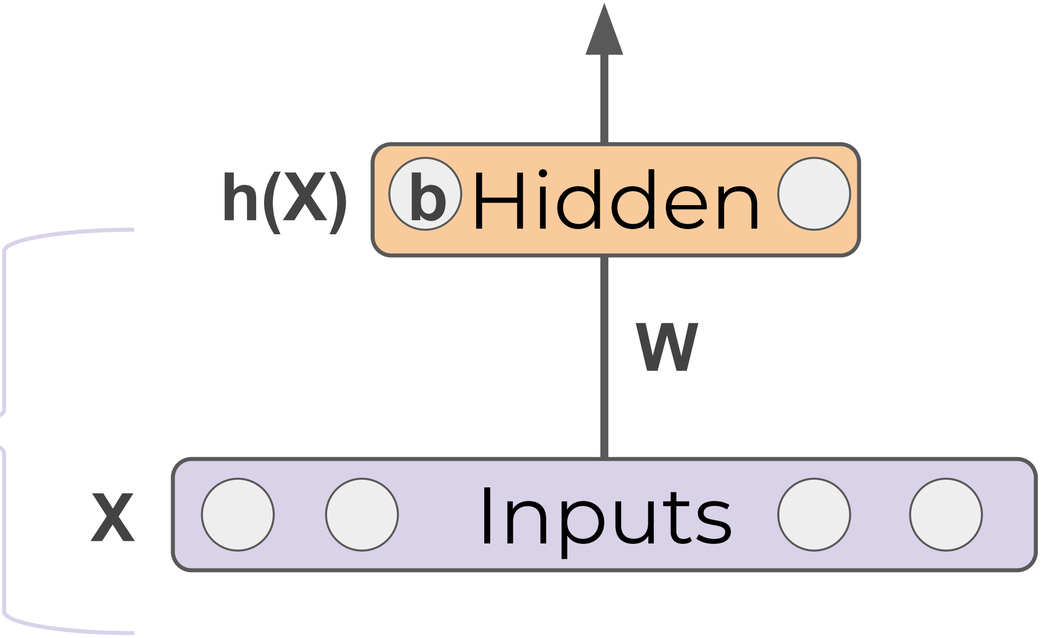


Linear Autoencoder

After training we can use the hidden layer to obtain a reduced dimensionality

Encoder

h(x)= b + Wx

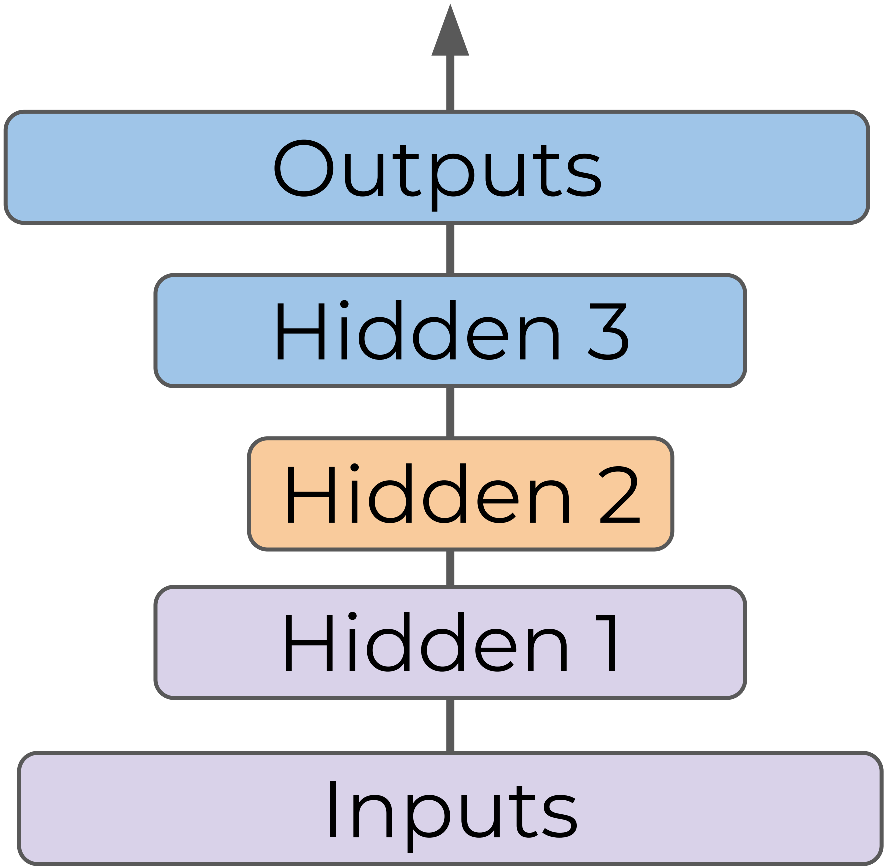


Deep Learning

- Let’s code this out to see how it works!
- Also check out the resource links for the mathematical proof of this.
- Let’s go to the jupyter notebook!

Deep Learning

- Let’s walk through the layers and explain the basic idea of an autoencoder.



Stacked Autoencoder

Saving and Restoring Models

Tensorboard

Deep Learning

- Tensorflow has many abstractions!
  - TF Learn
  - Keras
  - TF-Slim
  - Layers
  - Estimator API
  - and more!

Deep Learning

- Many of these abstractions reside in TensorFlow’s tf.contrib section.
- Typically libraries get developed and accepted into contrib and then “graduate” to being included as part of standard Tensorflow.

Deep Learning

- It is difficult to tell what abstractions are worth learning and which are not.
- The speed of development of TensorFlow has caused abstractions to come and go quickly!

Deep Learning

- This series of lectures will focus on presenting the most common abstractions used:
    - Estimator API
    - Keras
    - Layers

Deep Learning

- We will focus on understanding how to use these abstractions to build deep densely connected neural networks.
- Using these abstractions makes it easy to stack layers on top of each other for simpler tasks!

Deep Learning

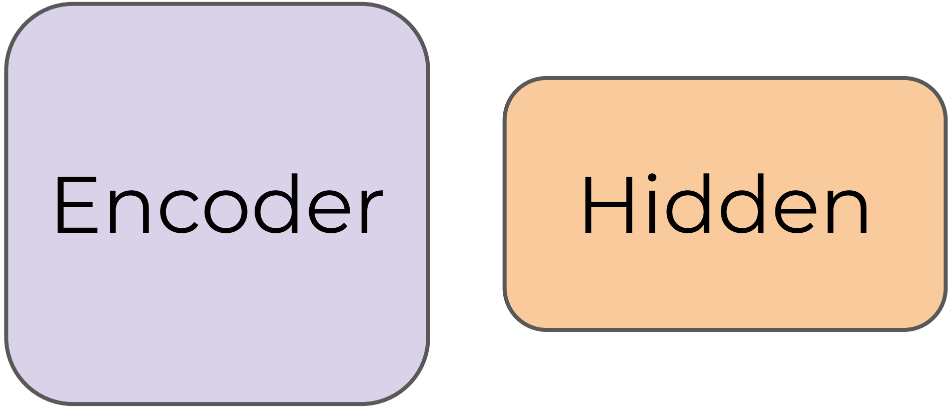
- All the code is available in Deep-Nets-with-TF-Abstractions.ipynb
- Let's start by exploring the data set we will be using!

Dimensionality Reduction with Linear Autoencoder Exercise

TensorBoard

Stacked Autoencoder

Example Simple Autoencoder



Example Simple Autoencoder