APS360 Fundamentals of AI

Lisa Zhang

Lecture 3; Jan 14, 2019

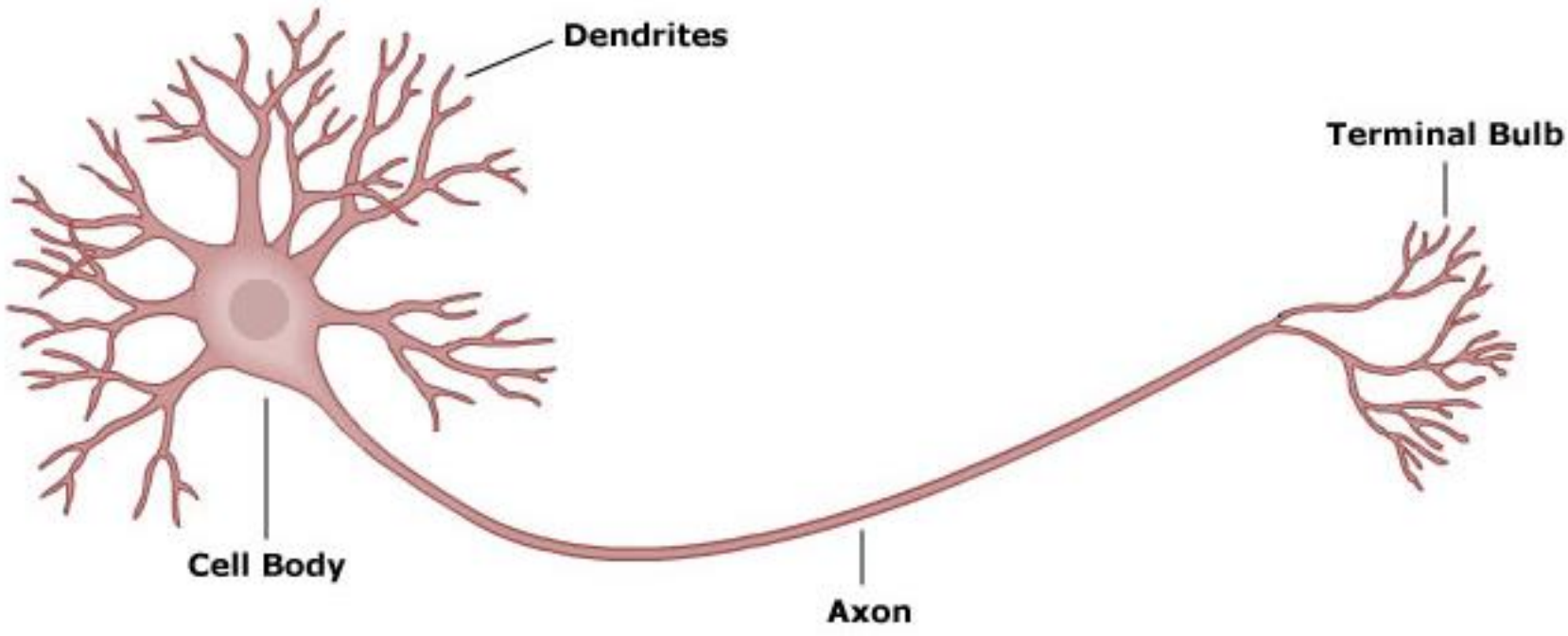
Agenda

Review Terminology!

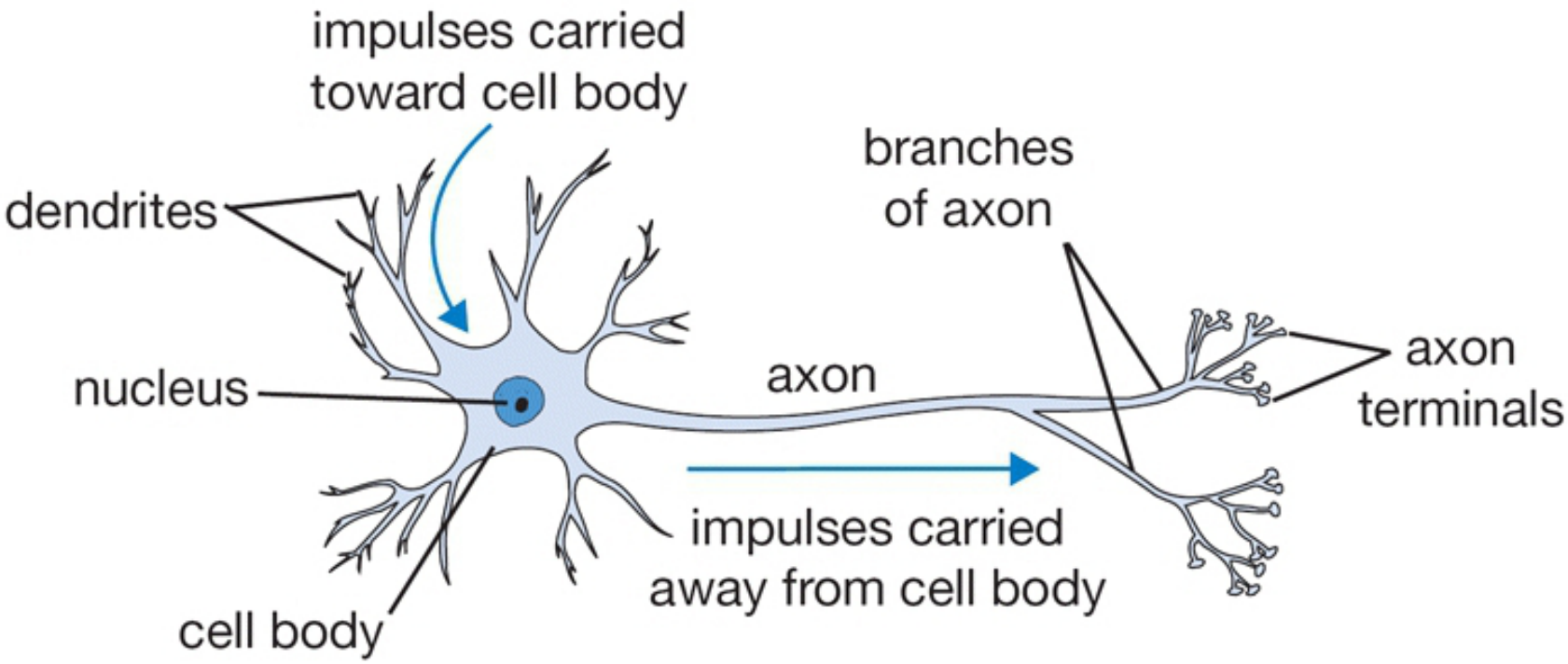
- Activation Functions
- Neural Network Architecture
- Training and test sets

Terminology from Last Week

Biological Neuron



Alternative Figure Showing Information Flow

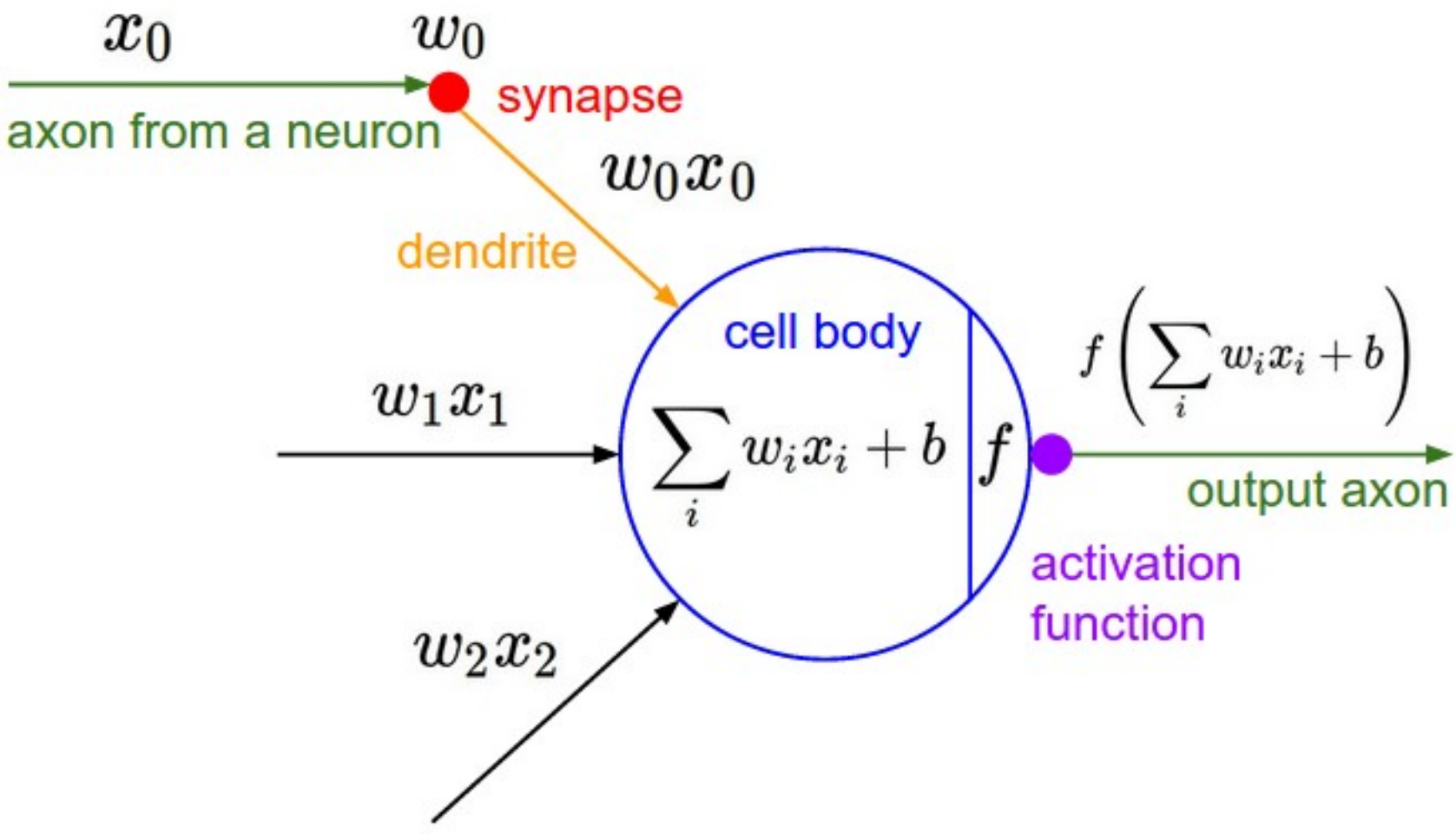


Anatomy

Information flow:

- (Axon of preivous cell)
- (Synapse)
- Dendrites
- Cell Body
- Axon
- Synapse
- (Dendrites of the next cell)

Artificial Neuron

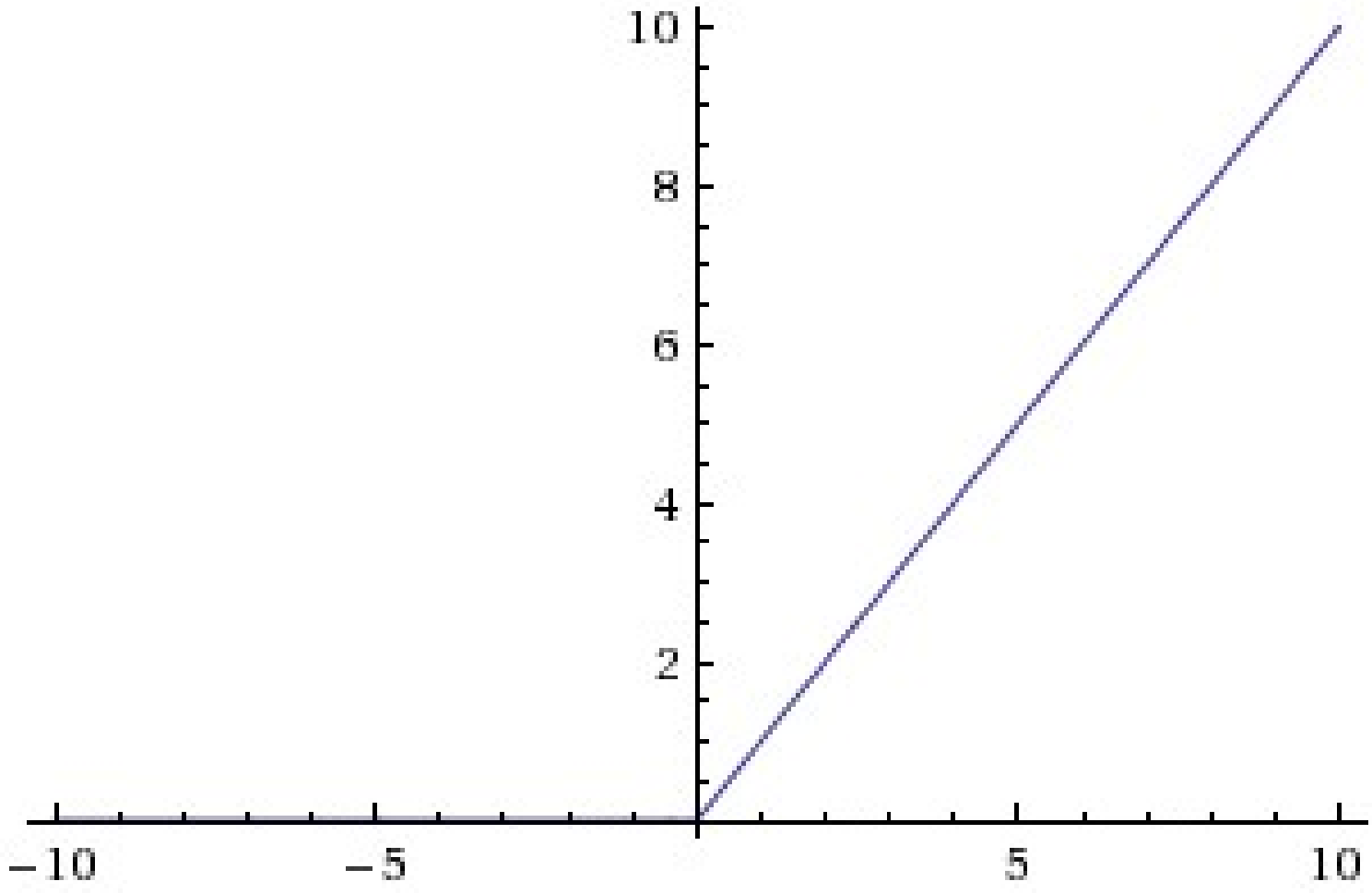


Activation Function

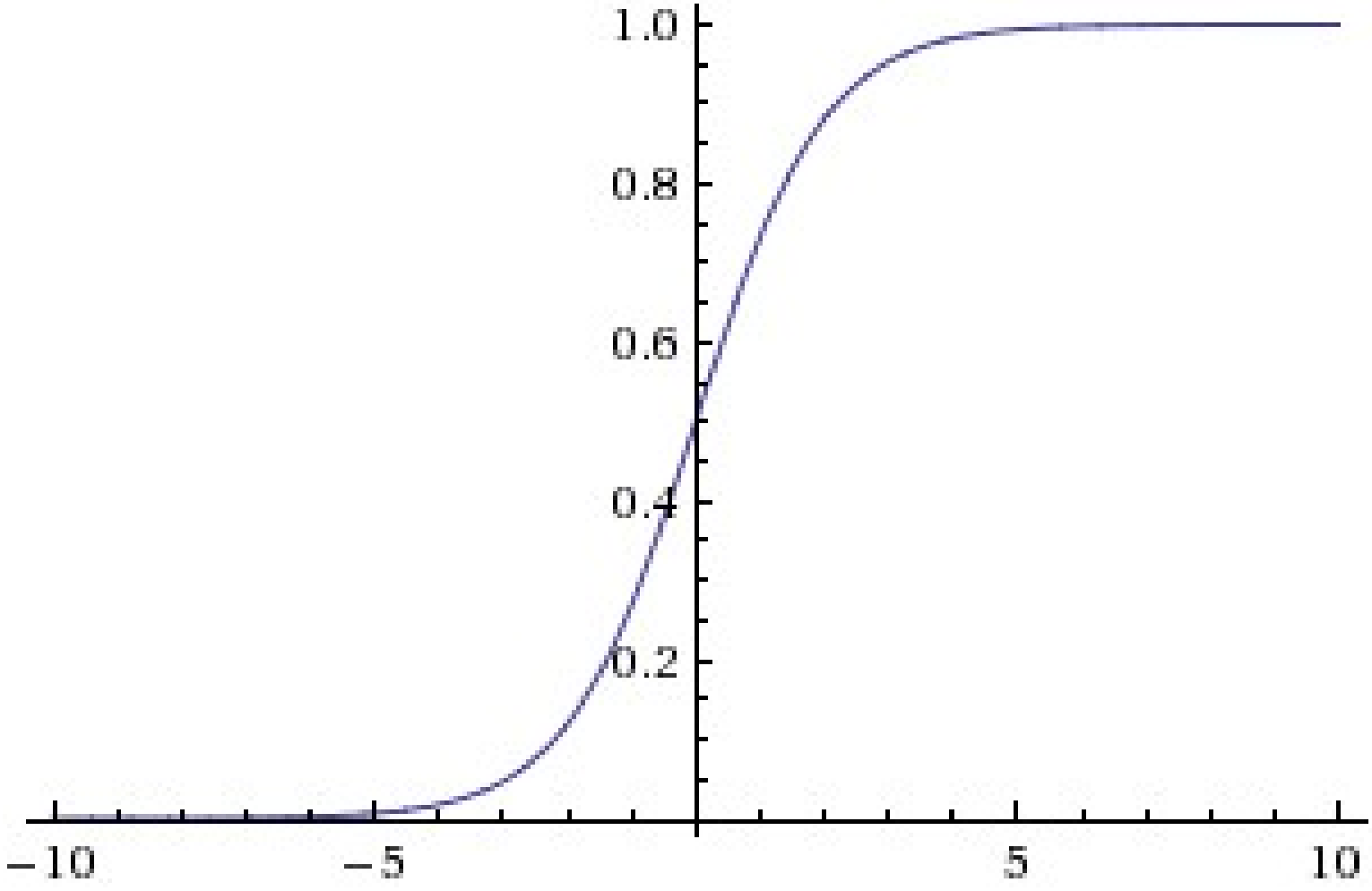
An **activation function** computes the activation of the neuron based on the total contributions from neurons in the layer below.

The activation function should be **nonlinear**. (Why?)

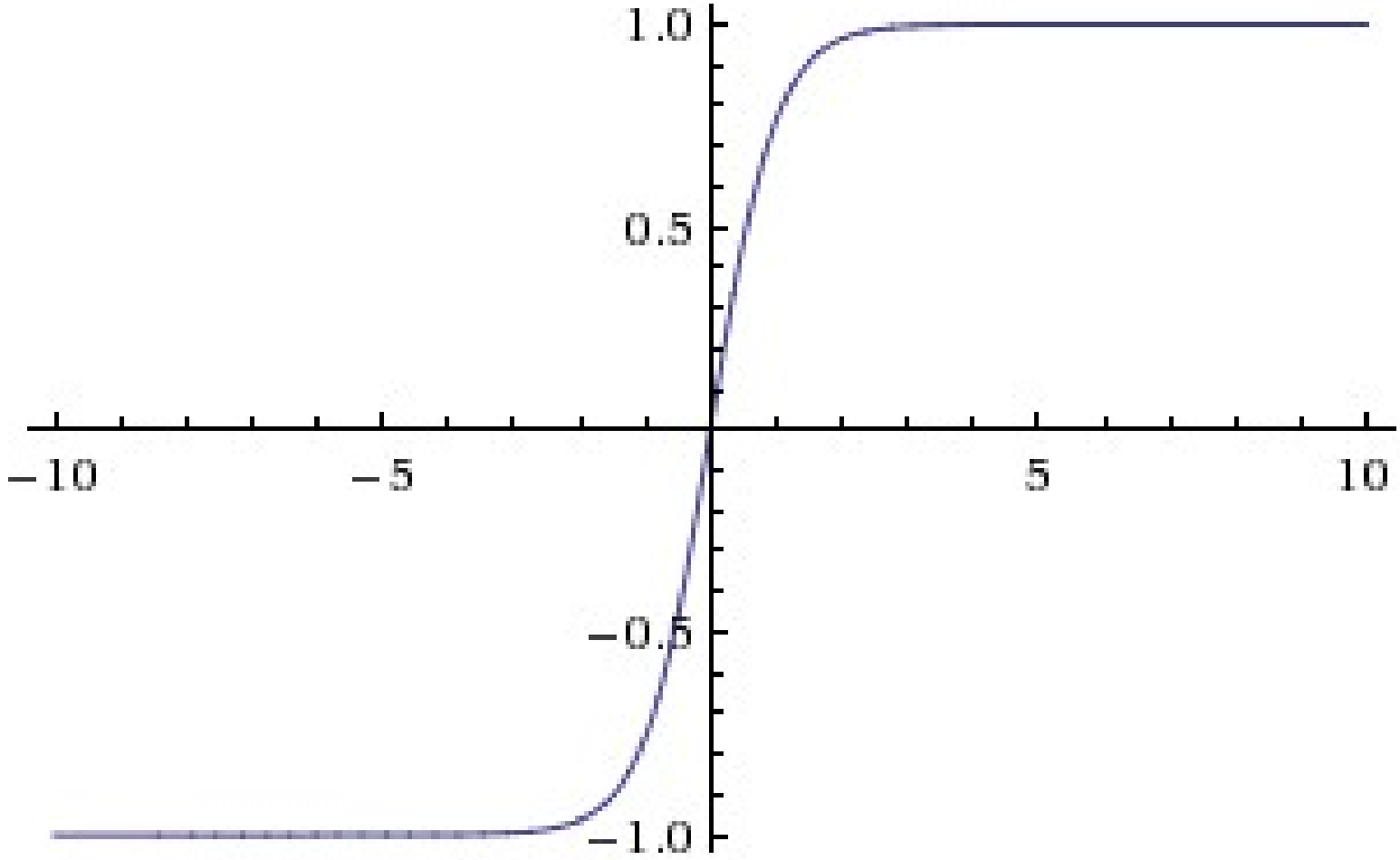
ReLU Activation



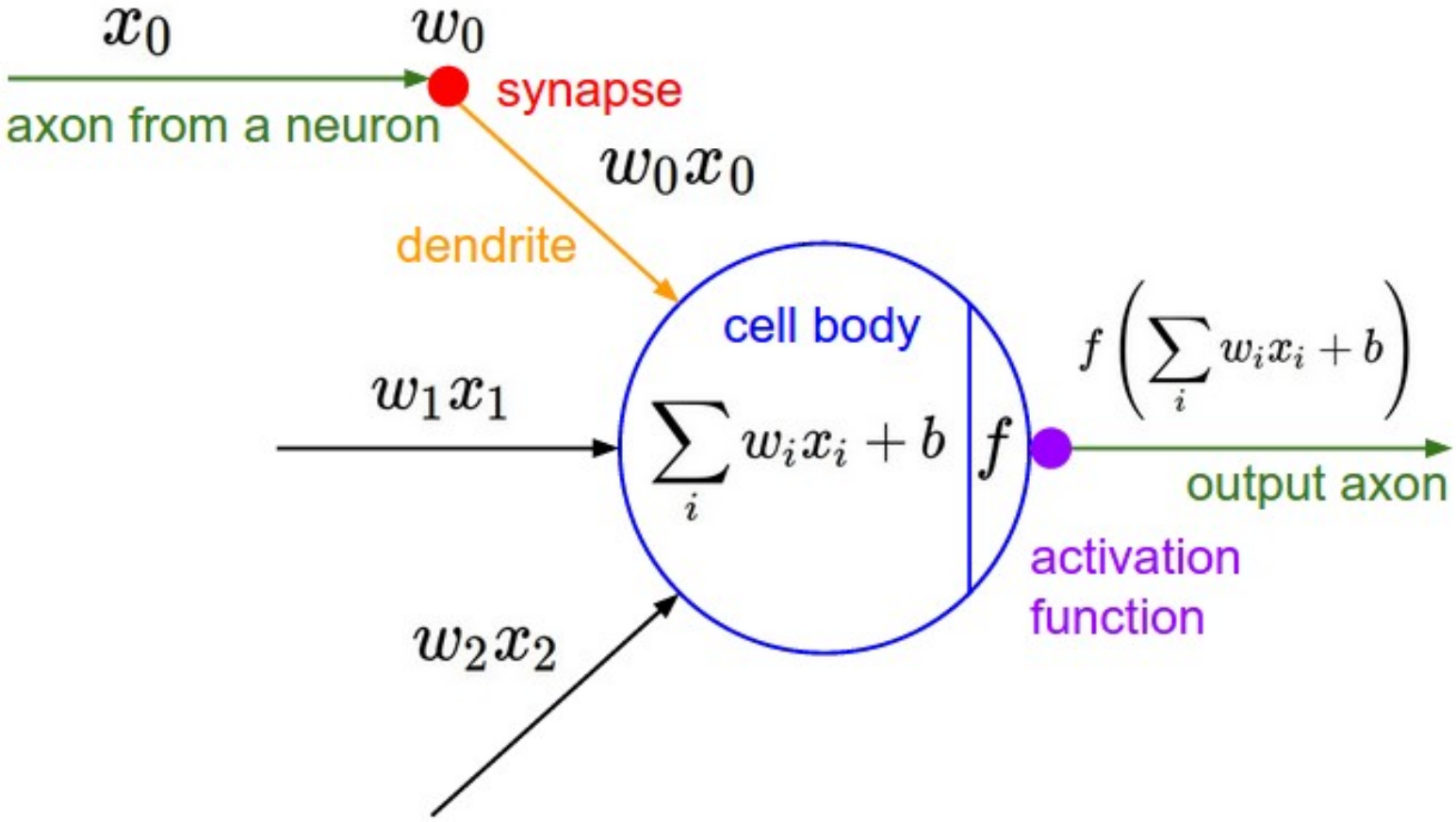
Sigmoid Activation



Tanh Activation



Parameters



The **parameters** of a network are the numbers that can be tuned to train the network. The parameters include the **weights** and **biases**.

We often use **weights** and **parameters** synonymously.

The **number of parameters** of a network is a measure of its size.

Neural Network Architecture

An **architecture** of a neural network describes the neurons and their connectivity in the network.

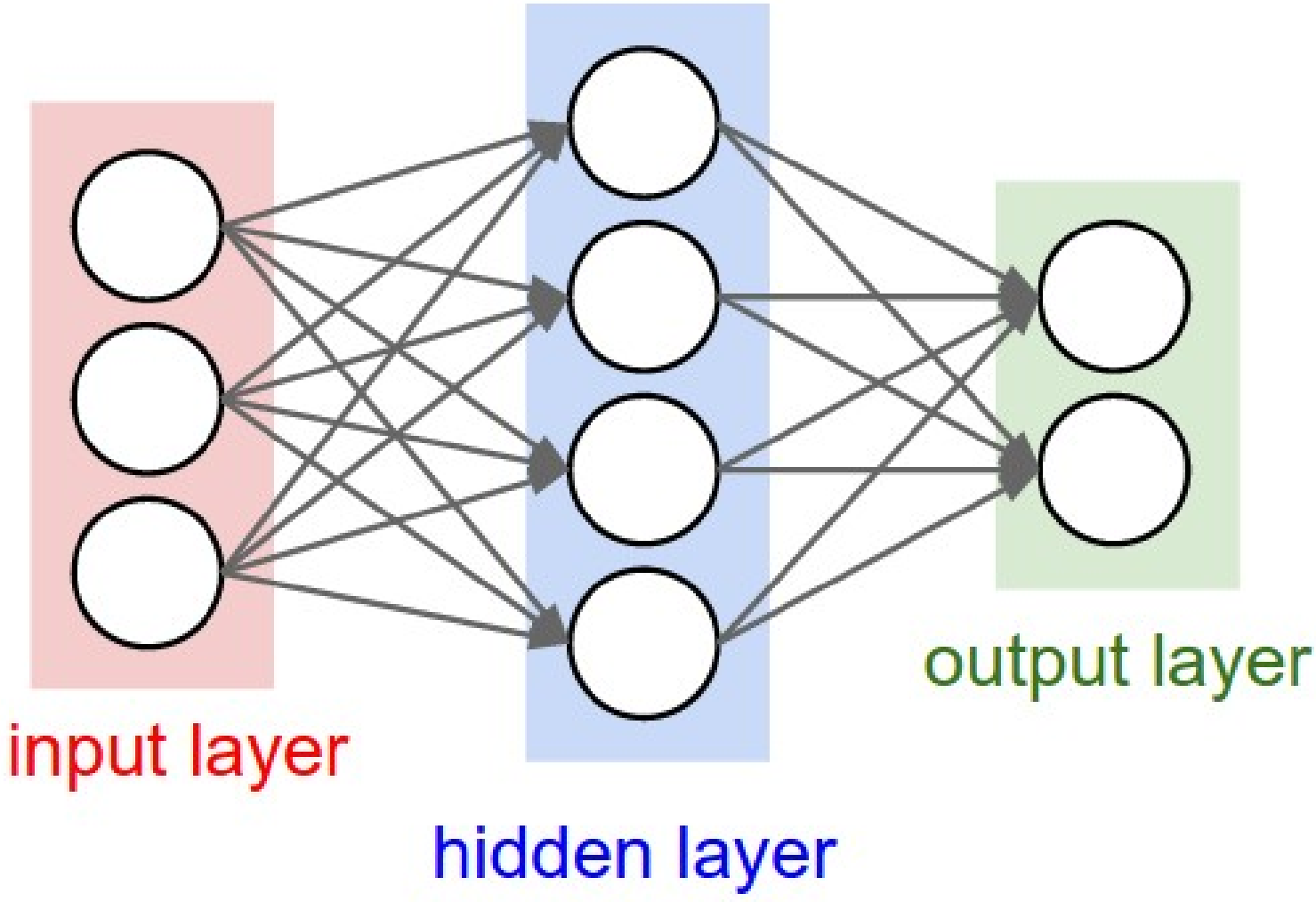
Feed-forward network

Information only flows from one layer to a later layer, from the input to the output.

Fully-connected layer

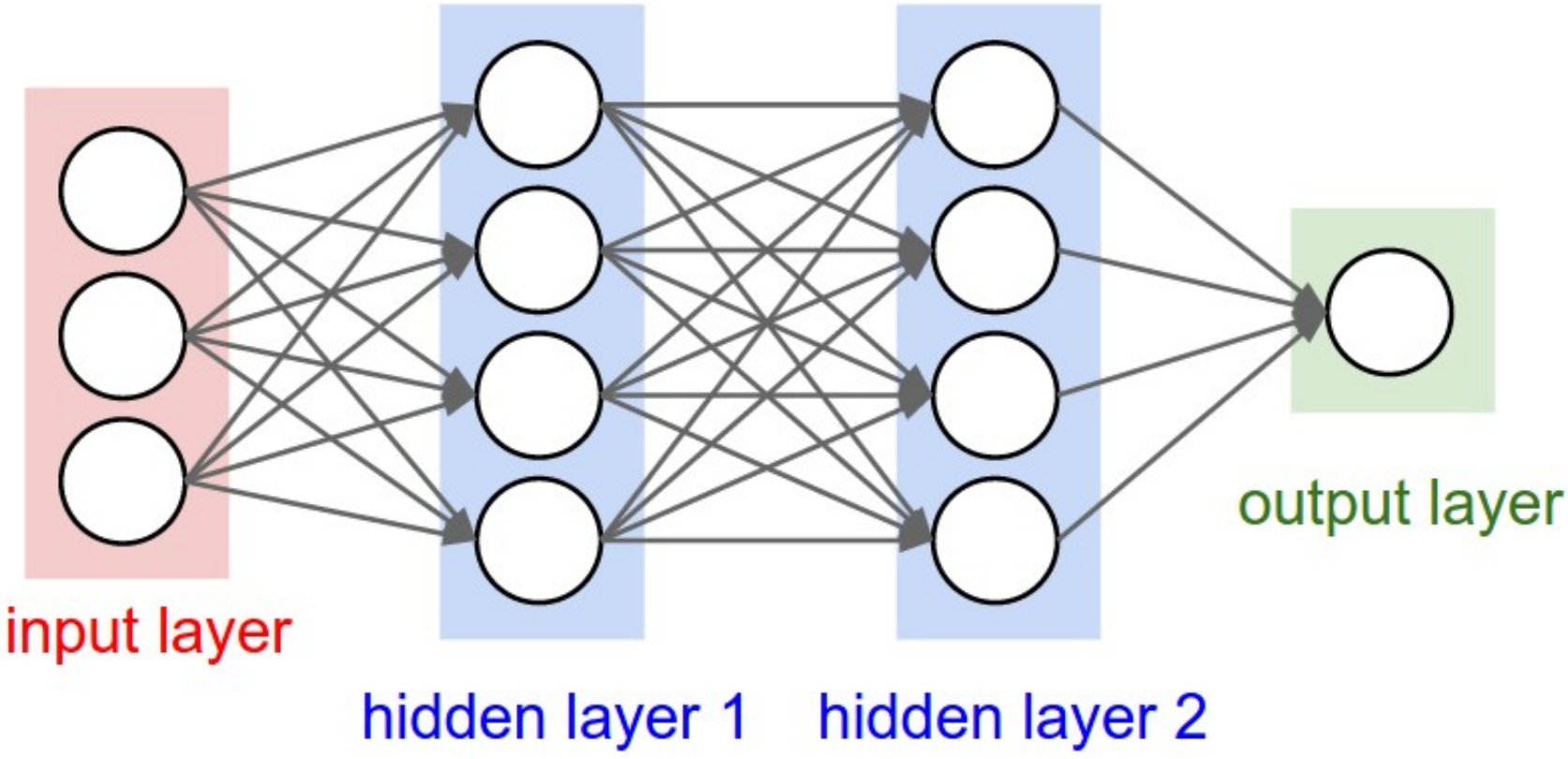
Neurons between adjacent layers are fully pairwise connected.

Number of Layers



This is a 2-layer neural network. We do not count the *input layer*, so the number of layers equal number of sets of weights and baises.

Number of Layers



This is a 3-layer neural network.

Training

We **train** a neural network to adjust its *weights*

Loss (Loss Function)

A **loss function** $L(\textit{actual}, \textit{predicted})$ computes how "bad" a set of predictions was, compared to the ground truth.

- Large loss = the network's prediction differs from the ground truth
- Small loss = the network's prediction matches the ground truth

Q: What are the inputs to the loss function?

Optimizer

An optimizer determines, based on the value of the **loss function**, how each parameter should change.

The optimizer solves the **credit assignment problem**: how do we assign credit (blame) to the parameters when the network performs poorly?

Optimize Step

We take **one step** towards solving the optimization problem:

$min_{weights} L(actual, predicted, weights)$

How do we do this?

Optimize Step

We take **one step** towards solving the optimization problem:

$min_{weights} L(actual, predicted, weights)$

How do we do this?

Using an optimizer like **gradient descent**.

Optimizer: Gradient Descent

All neural network optimizers you see in this course will be based on **gradient descent**.

We use the derivative of the loss function at a training example, and take a step towards its negative gradient.

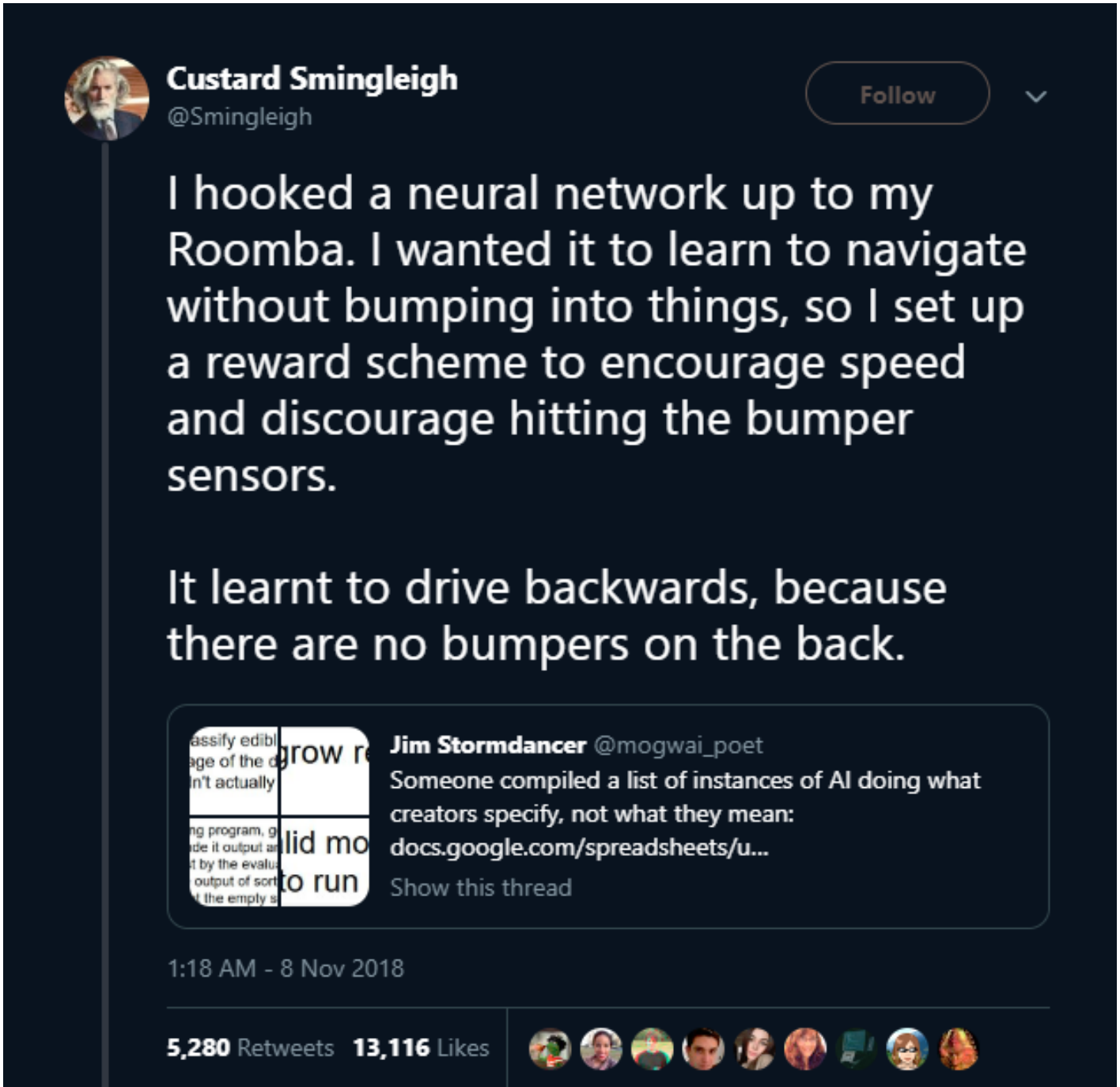
You don't need to know how optimizers work for this course.

From learning to optimization

Defining a loss function turned a **learning problem** into an **optimization problem**.

- Recurrent theme in Machine Learning

Caveats


Custard Smingleigh
@Smingleigh
Follow
I hooked a neural network up to my Roomba. I wanted it to learn to navigate without bumping into things, so I set up a reward scheme to encourage speed and discourage hitting the bumper sensors.
It learnt to drive backwards, because there are no bumpers on the back.
Jim Stormdancer @mogwai_poet
Someone compiled a list of instances of AI doing what creators specify, not what they mean:
docs.google.com/spreadsheets/u...
Show this thread
1:18 AM - 8 Nov 2018
5,280 Retweets 13,116 Likes

Train, and Test Set

- **Training Set**: Used to tune parameters
- **Test Set**: Used to measure network accuracy

Training and Test Splits

For standard data sets, there are standard train/test splits:

```
mnist_train = datasets.MNIST('data', train=True)
mnist_test = datasets.MNIST('data', train=False)
```

Why?

Neural Network Training

Next class!!