INTRODUCTION TO NEURAL NETWORKS

Motivation – 1/3

AMAZON ALEXA SOUND CLIP

Motivation – 2/3

APPLE IPHONE X FACE ID

Motivation – 3/3

GOOGLE AUTONOMOUS DRIVING CAR

What is Artificial Intelligence?

" A branch of computer science dealing with the simulation of intelligent behavior in computers"
– Webster dictionary

" A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E"
– Tom Mitchell

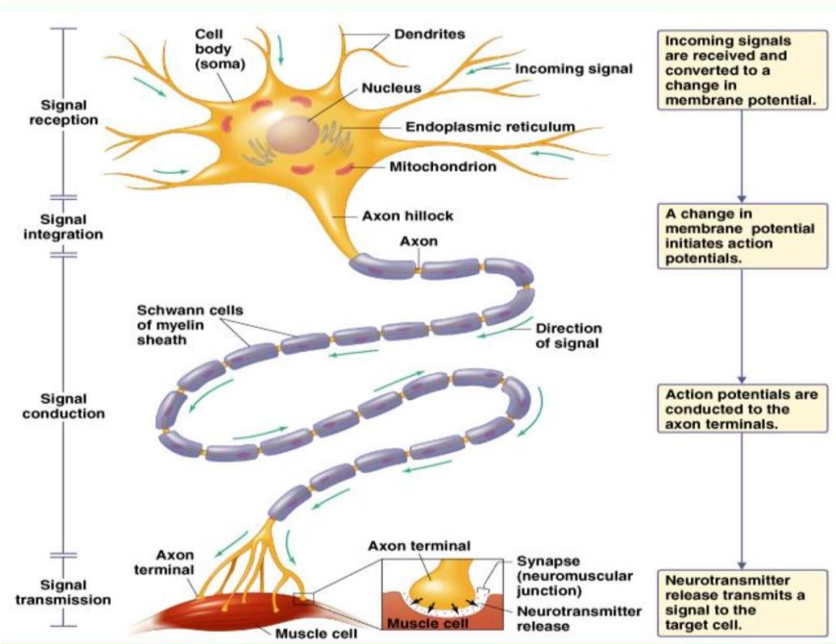
" The theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages."
– Oxford dictionary

" The modern definition of artificial intelligence (or AI) is "the study and design of intelligent agents" where an intelligent agent is a system that perceives its environment and takes actions which maximizes its chances of success."
– Science Daily

Inspiration for Artificial Intelligence

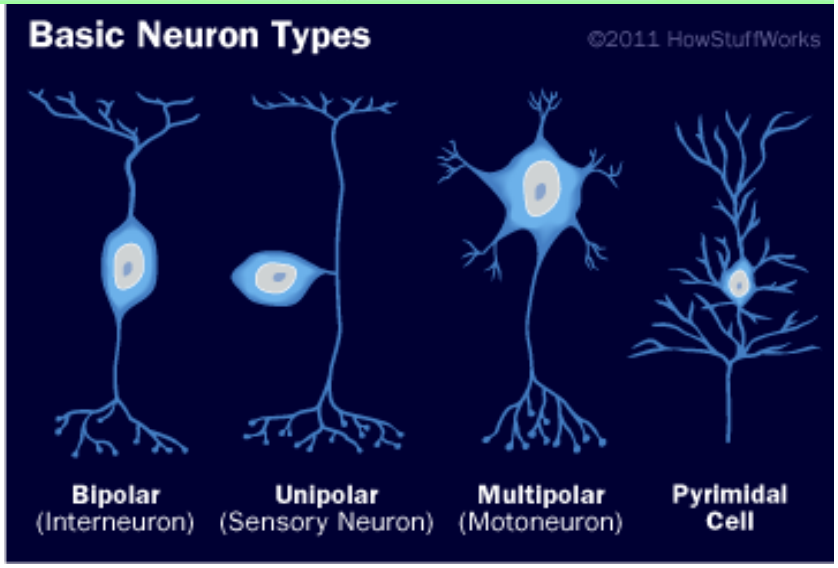
Inspired by the human mind

- A neuron is the basic component of the nervous system
- Appx. 86 billion neurons in the human brain
- Neurons transmit electric signals and enable the brain to understand and control the human body



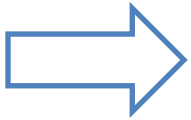
The evolution of AI

- McCulloch-Pitts neuron – one of the first artifical neurons created in 1943
- 1960 – 70 is considered the golden age of AI with developments at Cornell, Stanford & MIT
- The AI winter – limitations exposed, limited computing power, combinatorial explosion
- 1986 – The backpropagation paper to estimate gradients
- 2012 – AlexNet used GPUs to build deep learning model



Problem 1:

Write a program to allocate seats in the classroom to students based on their height.



ANSWER 1

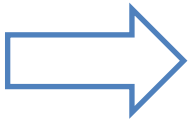
```
studentList = {s1,s2,s3,…..sn}

orderedList = studentList.orderBy(height)

seatAllocation = {p1,p2,p3…}
while orderedList.hasMoreElements():
        p1 = orderedList.pop()
        …
        …
```

Problem 2:

Write a program to determine if a student will do well in this course.



ANSWER 2

```
Features = {previousGrades,
            attendanceRecords,
            classParticipation,
            …
            …
}

//Train a model with selected features
model m = RandomForestClassifier()
m.fit(X[Features],y[Target])

//Use model to predict
Result = m.predict(X_test[Features])
```

Problem 3:

Write a program that will grant access to the classroom by checking the facial identity of registered students

Problem 1:

Write a program to allocate seats in the classroom to students based on their height.

This kind of problem can be referred to as Computer Programming.

Someone needs to tell the computer what to do, how to do it and the expected result.

Problem 2:

Write a program to determine if a student will do well in this course.

This kind of problem can be referred to as **Machine Learning**.

Someone needs to provide the computer with a set of features and the computer then identifies the relationship between the features and the result.

Problem 3:

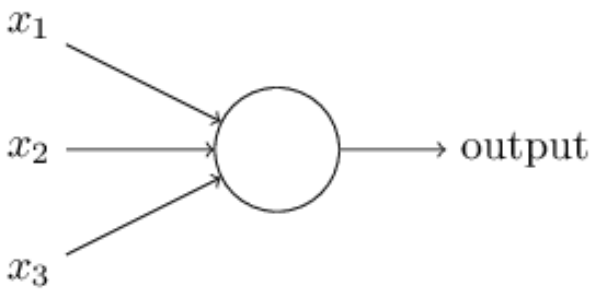
Write a program that will grant access to the classroom by checking the facial identity of registered students

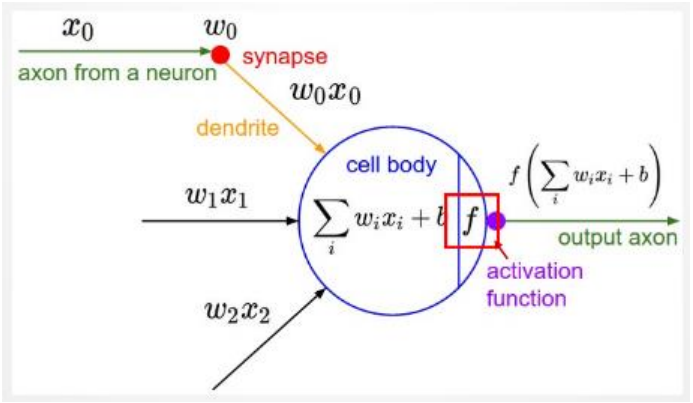
This kind of problem can be referred to as **Artificial Intelligence.**

Someone needs to provide the computer an input and output and the computer needs to identify the relationships between them.

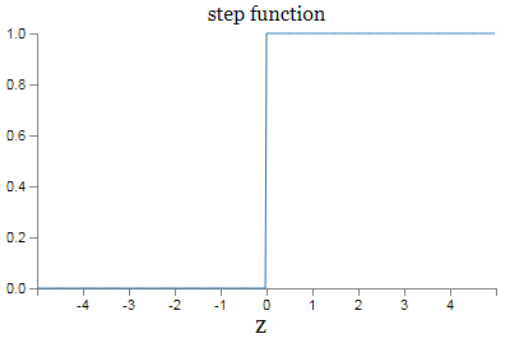
Introducing the Perceptron

- A perceptron is the primary building block of any neural network
- It provides us with a simple input/output operation with an activation function



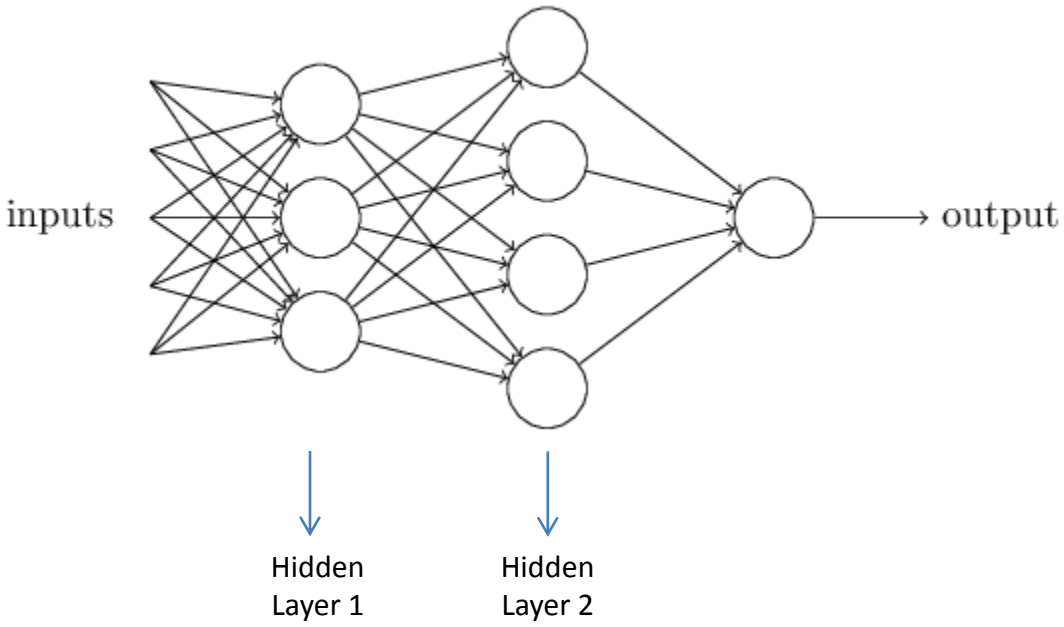


- **A simple neuron can accept one or more inputs**
- **May or may not have a bias variable**
- **Output is weighted combination of inputs -**
  Output = w1*x1 + w2*x2 + w3*x3 + ….. + bias
- **Activation function applied to output for final output**
- **Different activation functions – step function**



A simple Neural Network

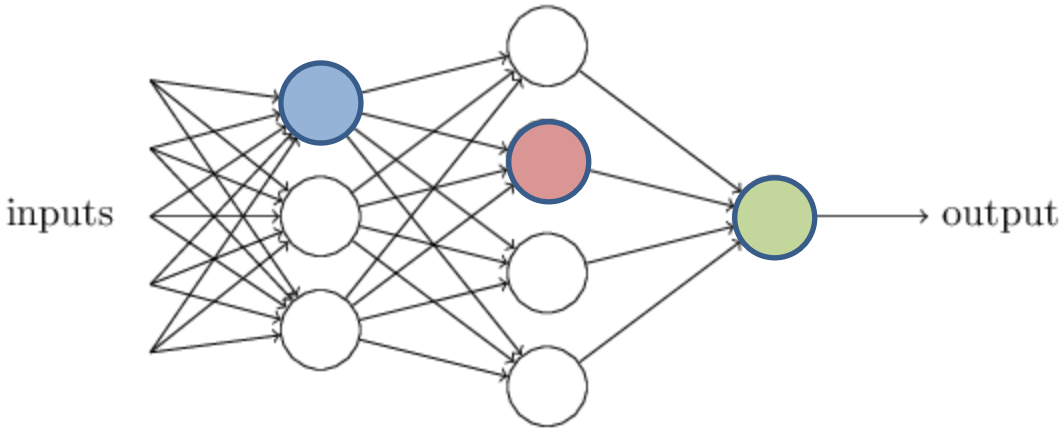
- A single layer can be created by stacking a number of perceptrons together
- Each layer that is not an input or output layer is called a hidden layer
- The output of every perceptron in a layer is used as the input for every perceptron in the next layer

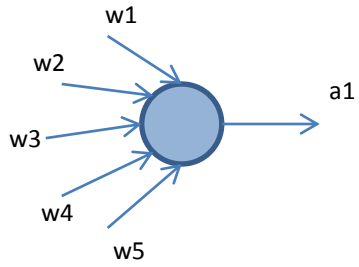


- Inspired by the human brain where different kinds of neurons interact in specific ways to achieve different functions of the human body
- These network architectures allow the neural network to learn complex functions and relations between the inputs and outputs

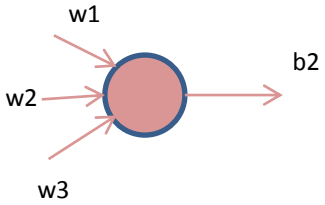
Let's understand these complex relations

- Let's follow one of the paths through this network and see what kind of a function is learnt

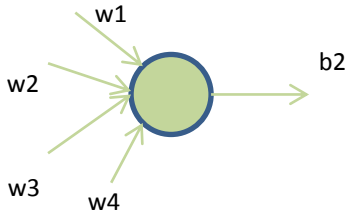




a1 = f(w1*x1 + w2*x2 + w3*x3 + w4*x4 + w5*x5)



b2 = f(w1*(f(w1*x1 + w2*x2 + w3*x3 + w4*x4 + w5*x5)
+ w2*a2 + w3*a3



y = f(w1*b1 + w2*(f(w1*(f(w1*x1 + w2*x2 + w3*x3 +
w4*x4 + w5*x5) +
w2*a2 + w3*a3)) + w3*b3 + w4*b4

- As you can see that while each neuron has a very simple function, because the input to a neuron is from multiple neurons, the functions becomes more complex as we move through the neural network
- This allows the neural network to learn complex relationships

Intuition of a Neural Network – 1/2

How do we visualize a neural network?

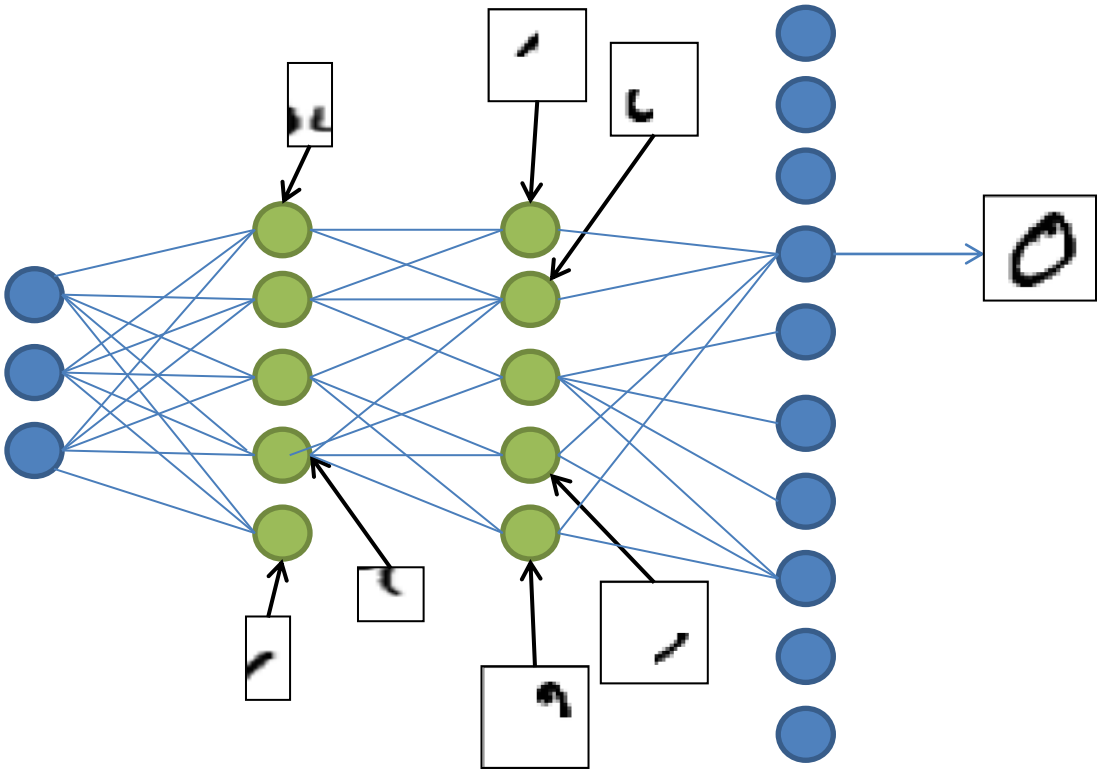
- What happens at a node?
- What are these activation functions at each node?
- How are they combined in subsequent layers?

HANDWRITING RECOGNITION

- Can we create some rules to identify specific digits, say zero?
  - **It should be a curve that meets itself**
    In contrast, this rule would also be met by the digit 8, with two such curves
  - **It should have a black curve that acts as a barrier between two white areas**
    In contrast, this rule may also be satisfied by the digit 6
- Difficult to create rules for such digit identification but a neural network is very good at learning such rules

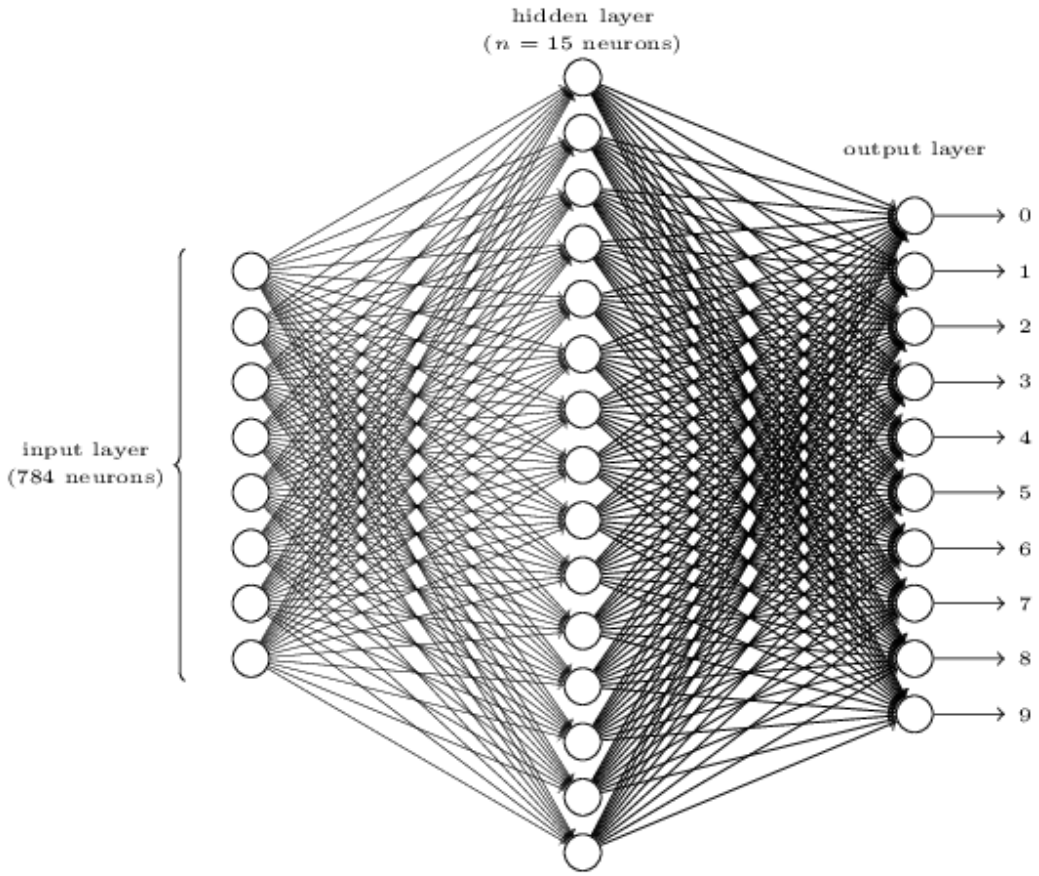
Intuition of a Neural Network – 2/2

Recognizing a handwritten ZERO



- More complex features are learnt as we move from one layer to the next
- Initial layer identifies line segments, differentiates dark from light areas
- Subsequent layer uses a combination of previous layers to identify more complex features like curves, wrap-arounds
- Final layer is able to synthesize all information to recognize final digit

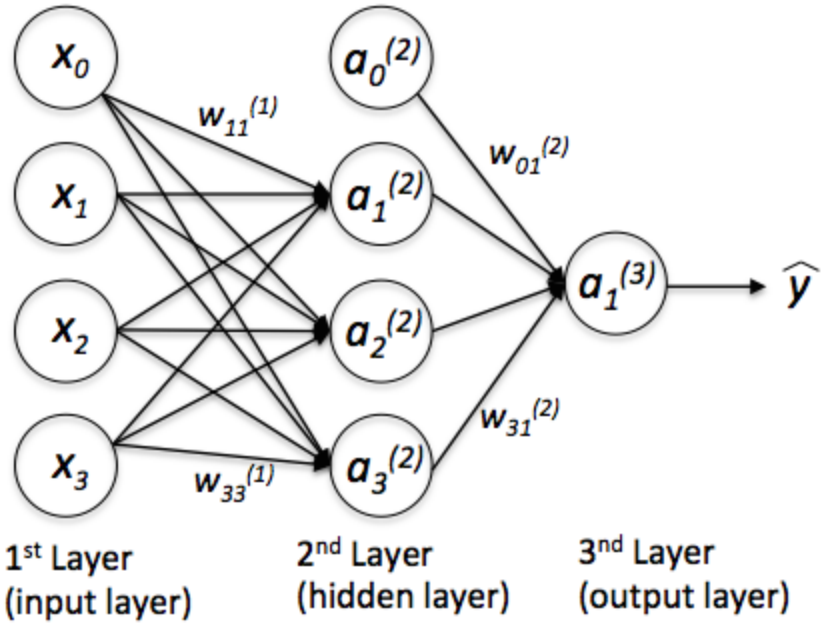
Fully Connected Neural Network



- Input layer – 784 neurons = 28 X 28 pixel image. Each input refers to one pixel in the image
- Hidden layer – 15 neurons, fully connected to Input layer and output layer
- Output layer – 10 neurons, with a different neuron firing depending on the actual digit recognized

Feed Forward

Process of traversing through the network starting from the input, calculating outputs and applying activation function for each neuron right till the output layer



When we do this traversal for one image –

- The input image used is known
- The weights and biases for each neuron are randomly initialized
- An output is calculated using the above weights and the input

However, we already know the value that the output must have. Therefore we only need to determine the appropriate values of weights and biases that will take us from input to output.

Cost Function

Infinite Possibilities	• Select weights and biases that ensure mapping between the input vector to annotated output • Large combination of values – hundreds of parameters with possibility to choose multiple values
Cost Function	• Define a cost function that quantifies error between the expected output and the calculated output • Measure of deviation from the expected solution • Adjust parameter values to minimize error i.e. cost function • Examples include quadratic error, cross-entropy error etc.
Objective	• Find an algorithm that will allow the parameters to be continuously modified to minimize cost function • This will result in the correct mapping of the inputs to the outputs through layers of neurons - the perfect **neural network**

ALGORITHM TO ADJUST PARAMETERS & MINIMIZE COST FUNCTION

ASSUMPTION

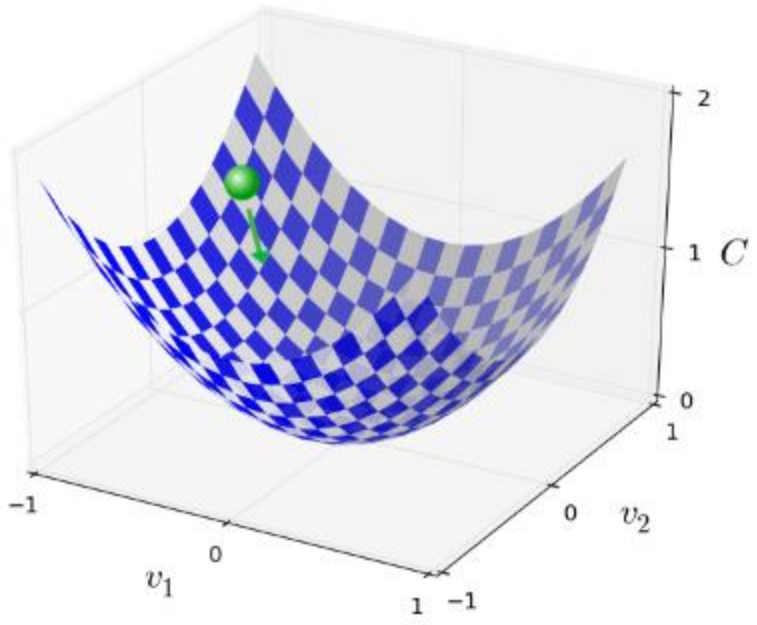
- Assume a cost function is selected as shown on the right
- This is a cost function with two parameters - v1 & v2

STEP 1

- Randomly select values of v1 and v2
- Calculate value of cost function - shown by green ball

STEP 2

- This value is not the lowest value of cost function
- Now adjust values of v1 & v2 to move the ball in the direction of lowest value - T*he Green Ball rolls downwards*



How to modify the values of v1 & v2 to move the ball downwards?

ADJUSTING PARAMETER VALUES

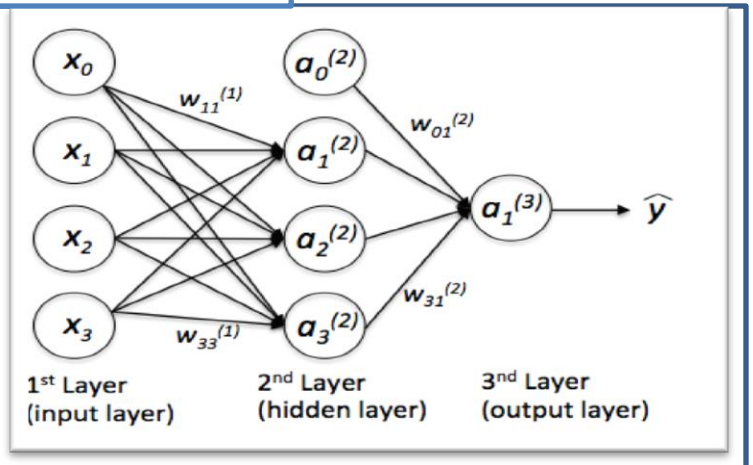
$$\Delta v = -\eta \nabla C,$$

η - Learning rate; amount by which we modify the parameter value

∇C, - Gradient descent of the cost function (provides the slope of the function)

GRADIENT DESCENT IN NEURAL NETWORKS

- Adjusting parameters using gradient descent work well for small number of parameters
- Simple Neural Network = 16 parameters
- One forward pass for a change in each parameter
- Neural networks typically have 1000s of parameters

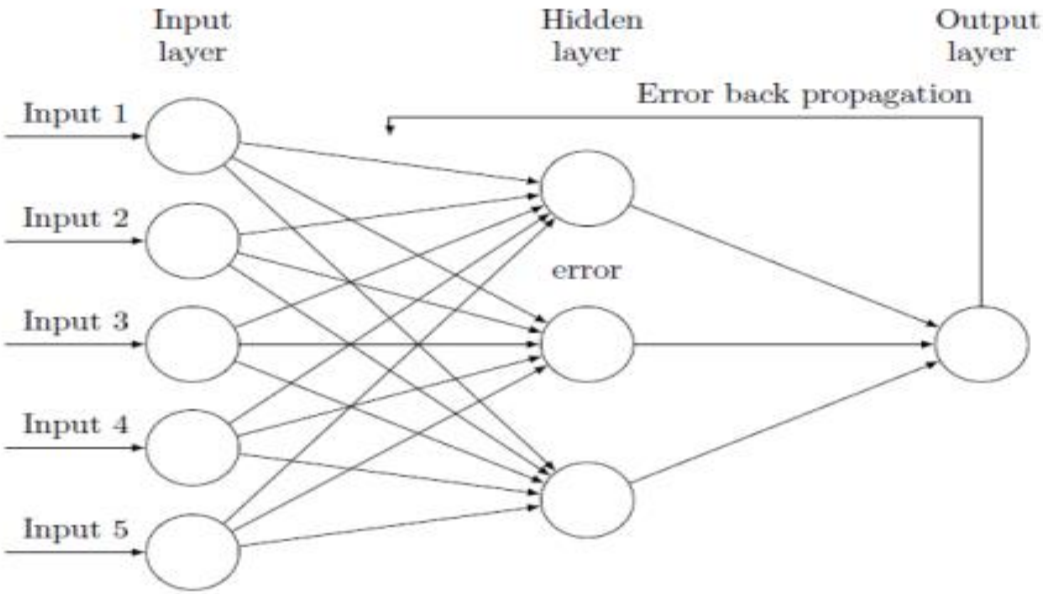


16 Forward passes for each gradient descent!

Backpropogation

TECHNIQUE THAT ALLOWS NEURAL NETWORKS LEARN PARAMETER VALUES USING GRADIENT DESCENT

1. Initial forward pass through the network using randomly assigned parameters
2. Calculated output and known output are used to determine the error
3. Backpropagate this error backwards through the network to obtain gradient at each neuron



The error at each neuron is a factor of error at previous neurons – using this and some standard equations, we can arrive at the gradient descent for each parameter and adjust those values for the next pass.

Steps to making Neural Networks work

1. Input Layer of the Neural Network is set to the pixels of the input image
2. Feedforward through the network by initially choosing parameters randomly and applying the activation function
3. Calculate output error using predicted output and actual output using cost function
4. Backpropagate the error backwards for each layer and neuron to calculate gradient descent for each parameter
5. Use the gradient for each weight and bias and multiply by learning rate to adjust the parameter value
6. Repeat steps 1-5 for a smaller set of images called the mini batch to approximate the Cost function for faster learning
7. Repeated for all mini batches till no more training examples are left. This is the end of an epoch. Repeat for multiple epochs.

Summarizing Neural Networks

- An artificial neuron is created to replicate a neuron in the human brain. It accepts one or more inputs and sums them up followed by applying an activation function

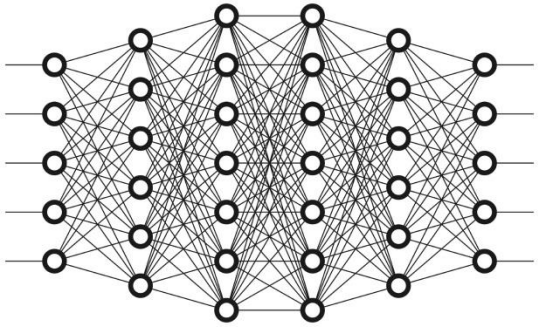
- Neurons can be stacked together to form layers and connected to other layers to form Neural Networks

- The purpose of a neural network is to find the optimum value of the parameters (weights and biases) so that a relation can be learnt between the input and output

- Gradient descent is used to continuously adjust the value of the parameters and backpropogation is used to determine the values by which to adjust each parameter

Artificial neural networks form the basis for deep neural networks

Deep Learning is the field of creating large, complex network architectures so that more complex relationships can be learnt.

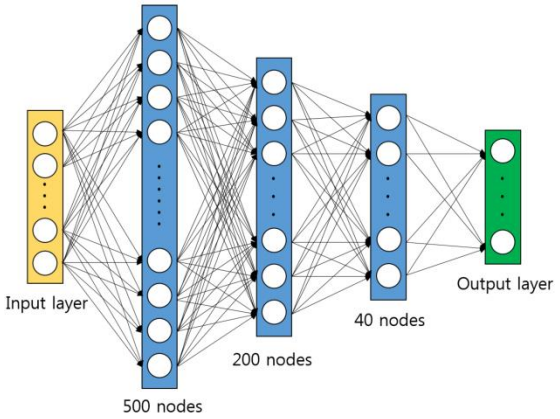


Complex Architectures created by –

- Adding multiple layers
- Adding more neurons in each layer
- Changing the type of layer
- Changing the type of activation function

Research Area This is a very active area of research and we will not go further into it today.

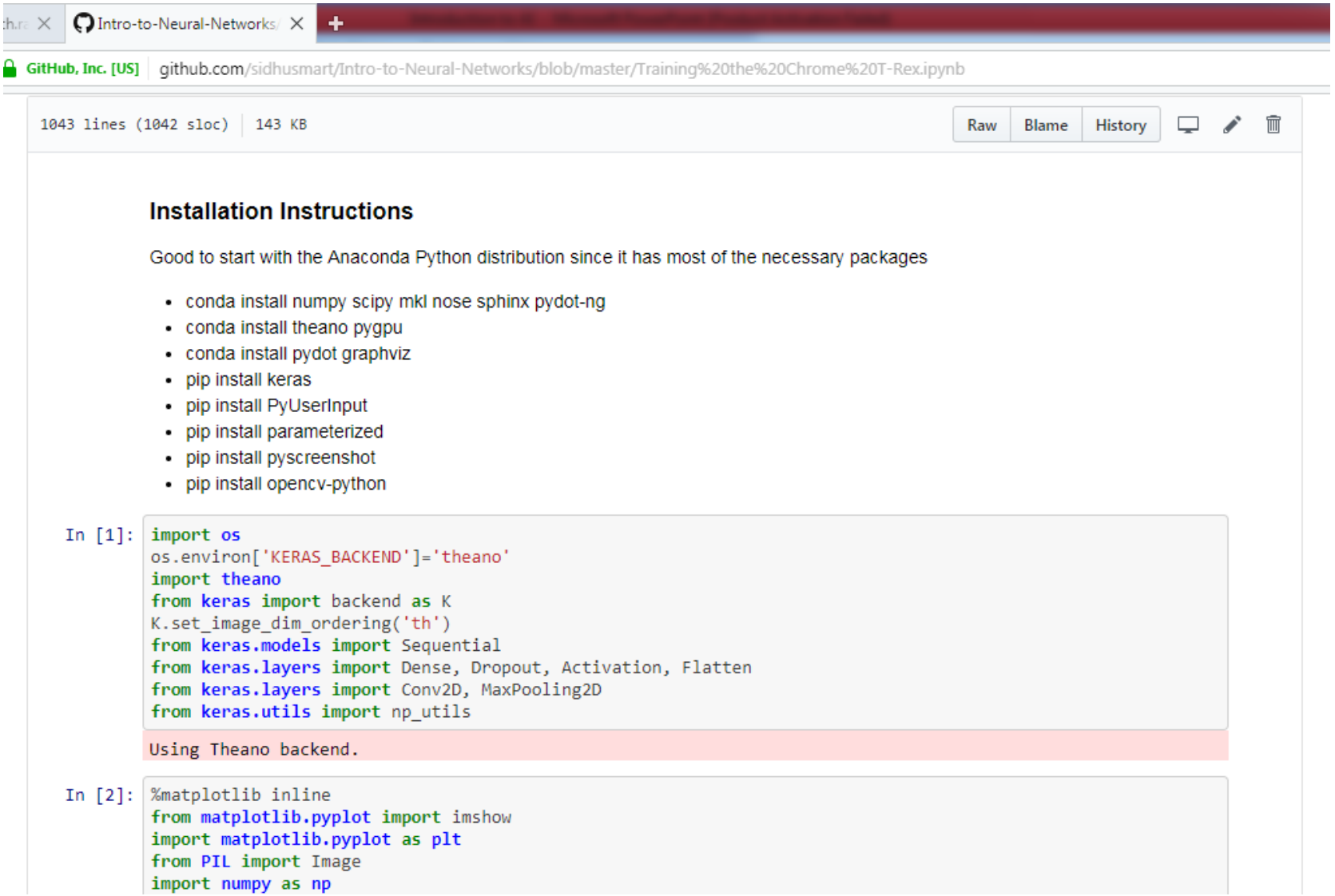
In the problem exercise that we will solve, we will use what are called Convolutional Neural Networks. These are commonly used in solving Image recognition problems.



Google Chrome T-Rex challenge

- We will create a program that can play this game autonomously
- Possible to solve this problem in a programmatic way by providing some simple rules
- Since we have learnt artificial neural networks we want to test and see if a computer can be taught to learn on it's own – **Train the T-Rex**!

Walkthrough Jupyter Notebook



THANK YOU

sidharth.ramachandran@gmail.com