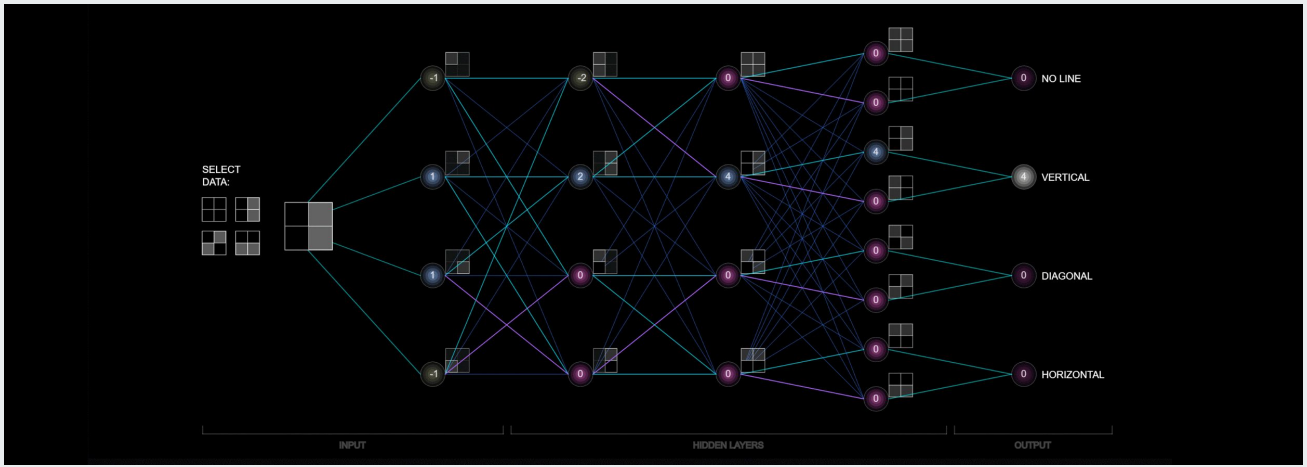
Machine/Deep Learning

By: Rohan Sikand

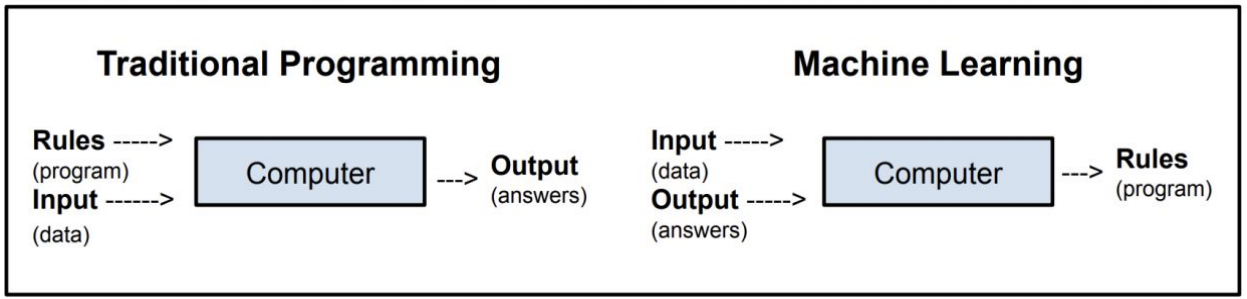


Machine Learning

- Subfield of Artificial Intelligence
- a field that brings together computer science, statistics, and mathematics.
  - In simple terms, machine learning focuses on pattern recognition and learning from data.
- Definitions:
  - ”A field of study that gives computers the ability to learn without being explicitly programmed” - Arthur Samuel
  - "A set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data" - Kevin Murphy

Difference Between Traditional Programming

- Learn patterns from data instead of "hard-coding" useful characteristics.



Impact

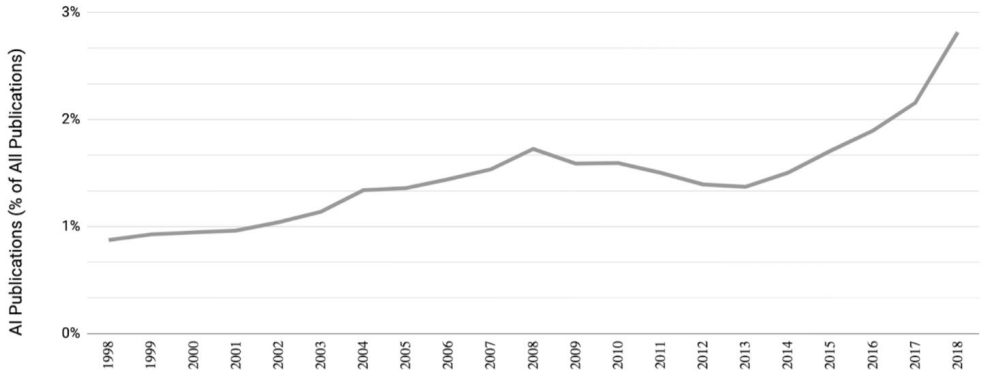
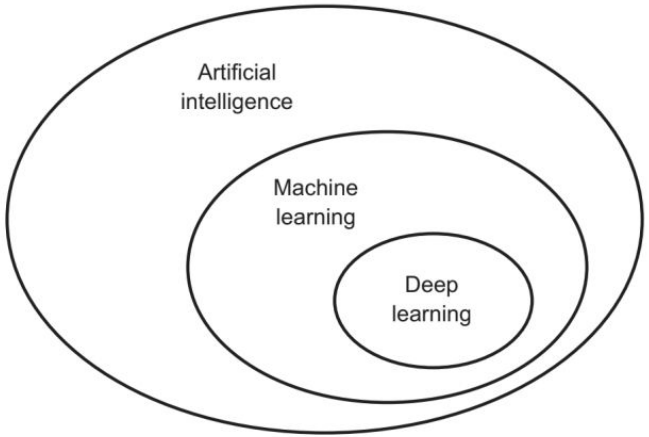


Figure 1: Since 1998, the total number of AI papers among all papers published worldwide has grown three-fold, now accounting for 3% of all journal publications. Source: [1]

Deep Learning

- Subfield of Machine Learning (which is a subfield of AI). Deep Learning, an end-to-end process, focuses on one specific algorithm: a **Neural Network**.

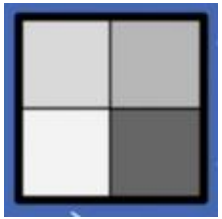


Two main topics within neural networks and deep learning:

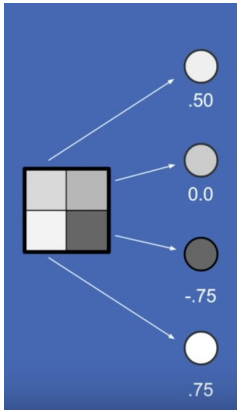
- **Building** a neural network
- **Training** a neural network (also known as **learning**)

Neural Network: Input Layer

1. Take the following sample image, consisting of 4 pixels (2 x 2), as an example input:

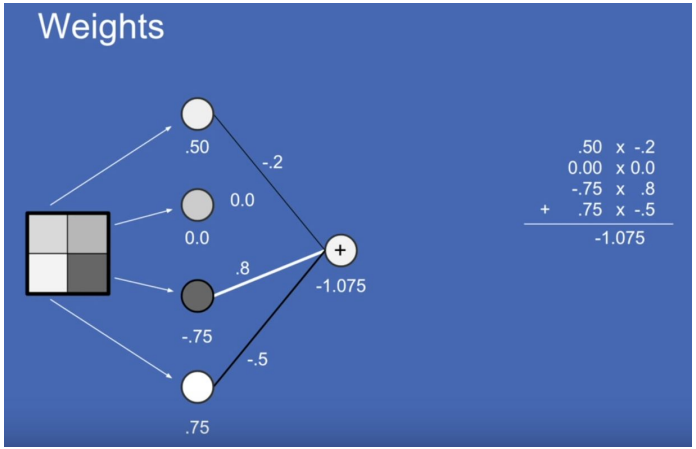


2. Each pixel is defined as a neuron which contains a scalar value between -1 (black) and 1 (white). This forms the input layer:



Neural Network: Next Neuron

3. From this, a new neuron is formed with a connection to each input neuron. "**Weights**" are multiplied to each scalar input neuron as shown in the figure. These results are then summed. A **bias** value is added to the sum to give the model more flexibility. This value constitutes the next neuron.



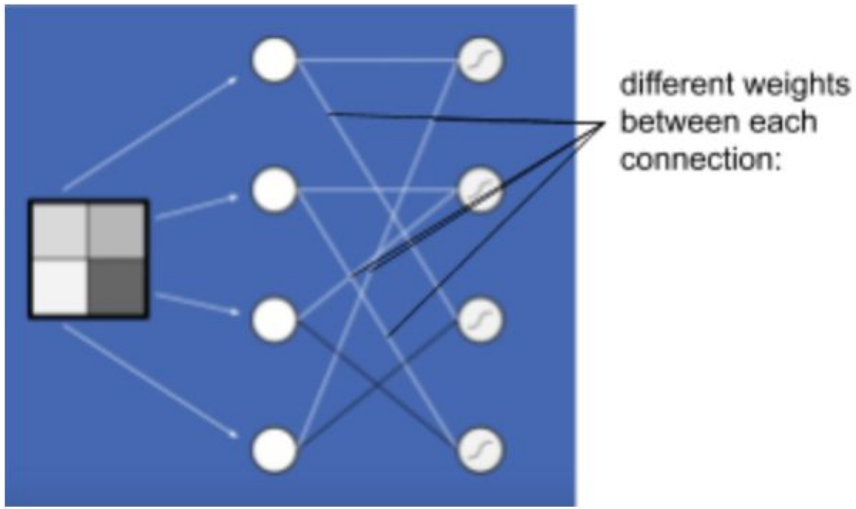
In mathematical notation, the weighted sum, s, is calculated as follows:

$$s = \sum_{i=1}^{n} w_i x_i + b$$

where s the weighted sum of all the inputs, n is the total number of inputs, x_i is all of the inputs, w_i is all of the weights, and b is the bias.

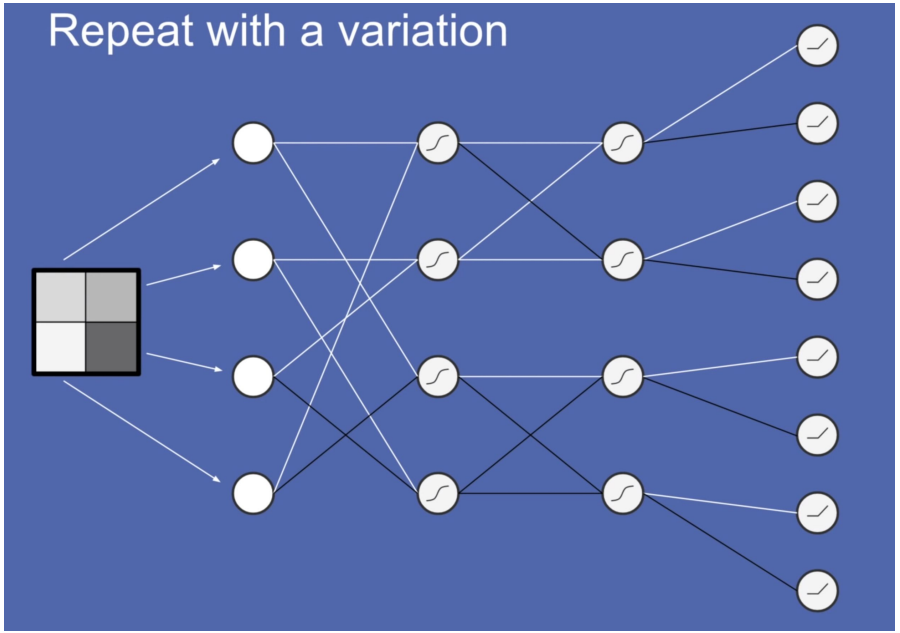
Neural Network: Next Layer

5. The new neuron value is formed. This process of creating a neuron can be repeated through different connections (usually millions). The process is identical except with different connections and different weights. These connections form the next layer of the neural network.



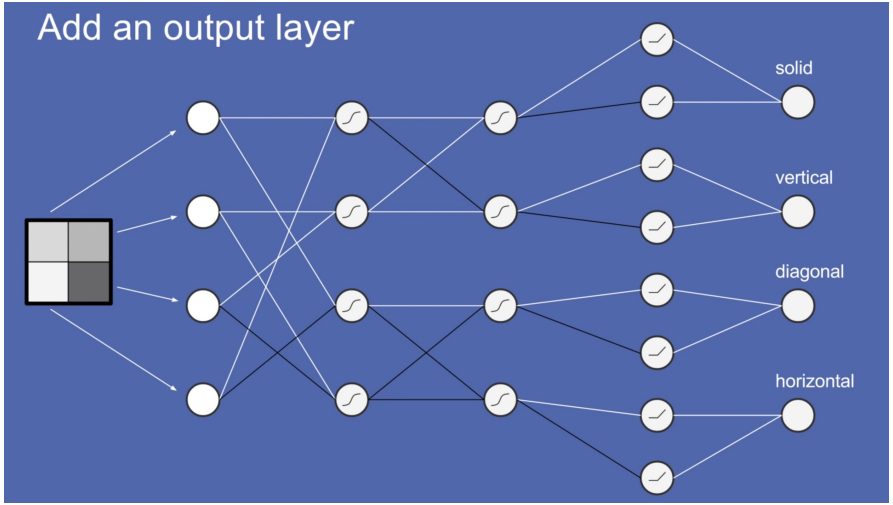
Add More Layers

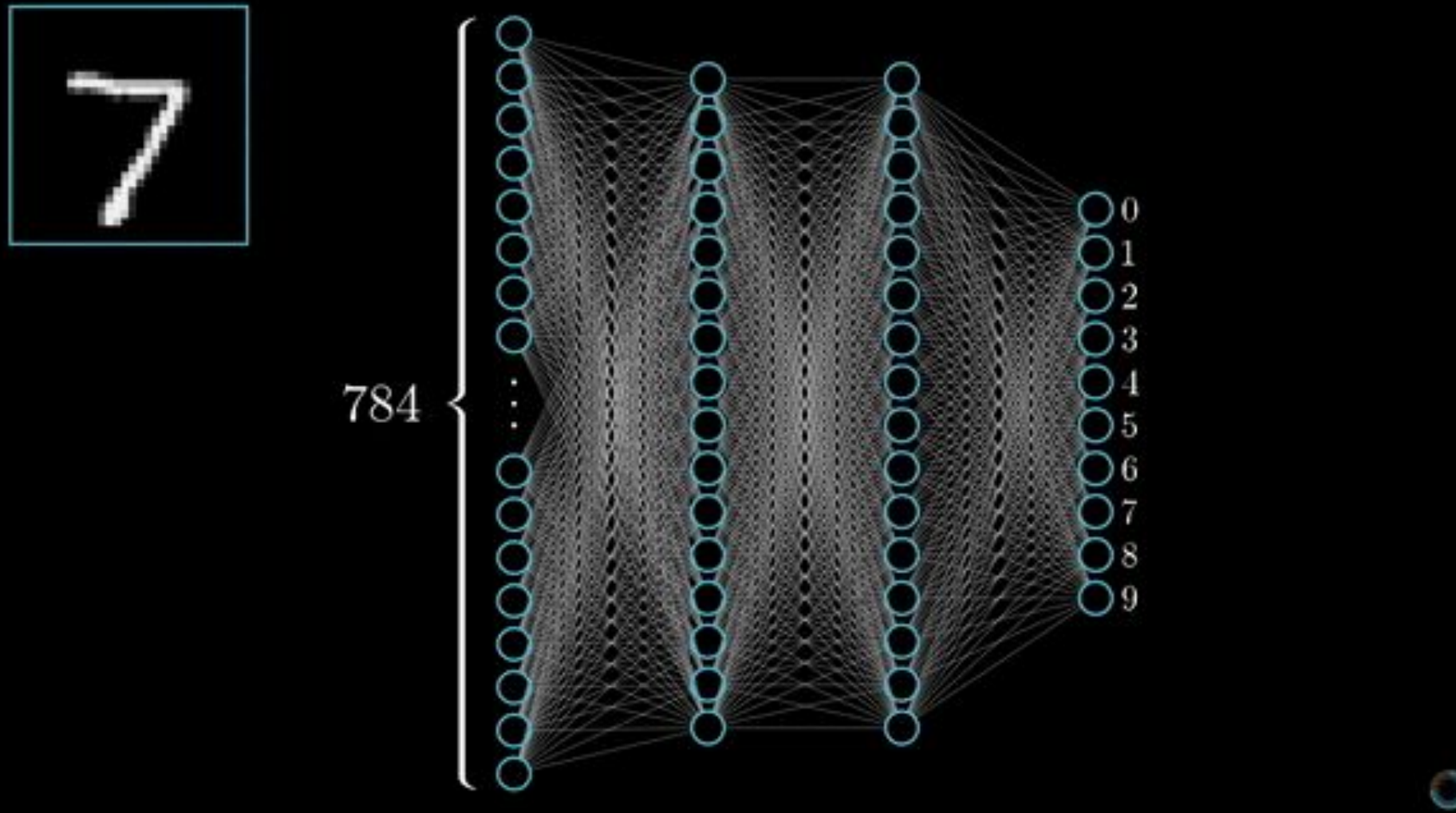
6. Add more layers with different amount of neurons, activation functions etc.



Output Layer

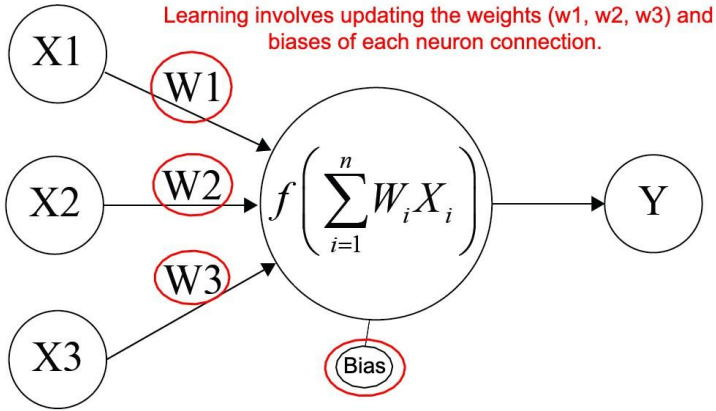
7. After more layers are added, an output layer is used to form the classification prediction. In the context of this problem, we have four possible classes for the image. Thus, the output layer will have for four neurons--each holding a scalar value.




784
0
1
2
3
4
5
6
7
8
9

Learning

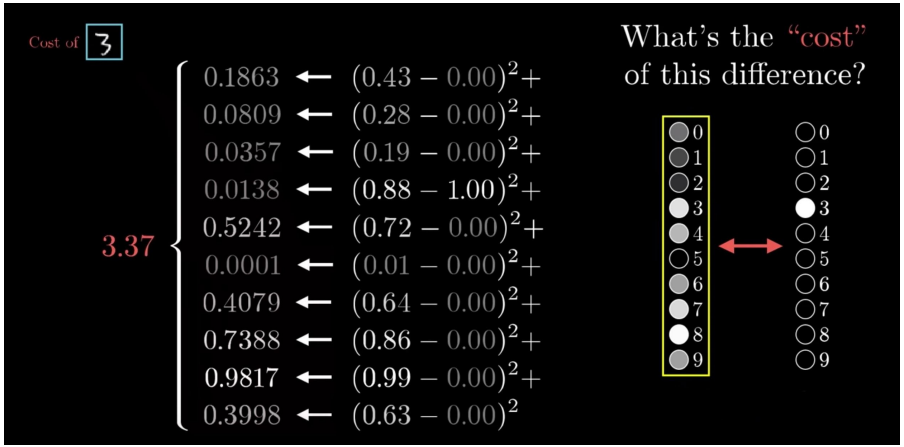
- What exactly is "learning" in a neural network?
  - The adjustment of weights and biases through optimizing a loss function.



Learning: Loss/Cost

- The loss or "cost" is calculated by summing the squares of the differences between the predicted activations (scalars) in the output layer neurons and the activation you wanted it to have.

$$C(w, b) \equiv \sum_x \|y(x) - a\|^2.$$

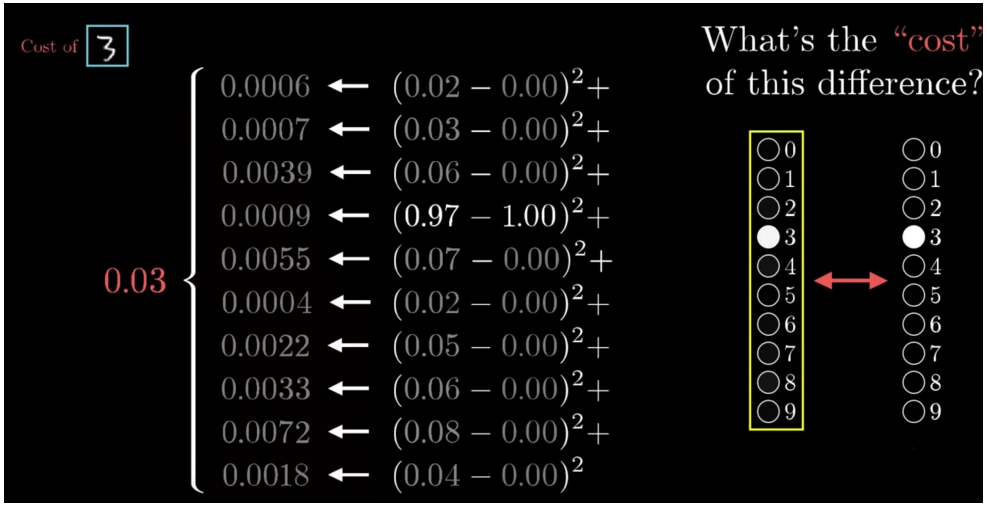


Learning: Loss/Cost

Notice that the lower the loss value, the more accurate the network is:

...

Hence, the idea is to "**minimize**" the loss function



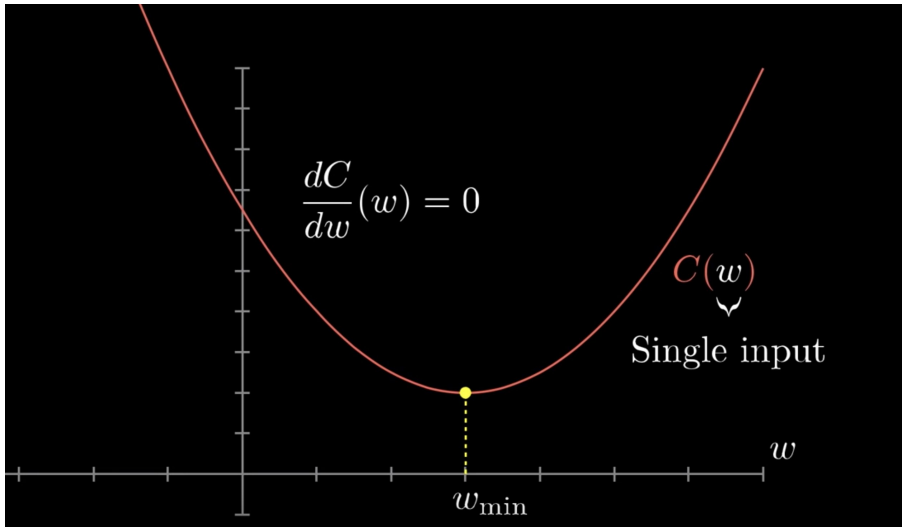
What constitutes the loss function?

- What are the inputs of the high-dimensional loss function?
  - The inputs (and dimensions) are all the weights and biases. Thus, finding the minimum requires adjusting the weights and biases.
- Remember, since this loss function is usually >3 dimensions, we cannot visualize this. However, in mathematical notation, we can use a matrix defined as follows:

$$loss(w_1, w_2, \ldots, w_i, b_1, b_{2,i}) = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_i \\ b_1 \\ b_2 \\ \vdots \\ b_i \end{bmatrix}$$

Learning: Finding the minimum (optimization)

In a simple two-dimensional world, we can plot this function and find its minimum through optimization--the process of taking the derivative and setting it equal to 0.

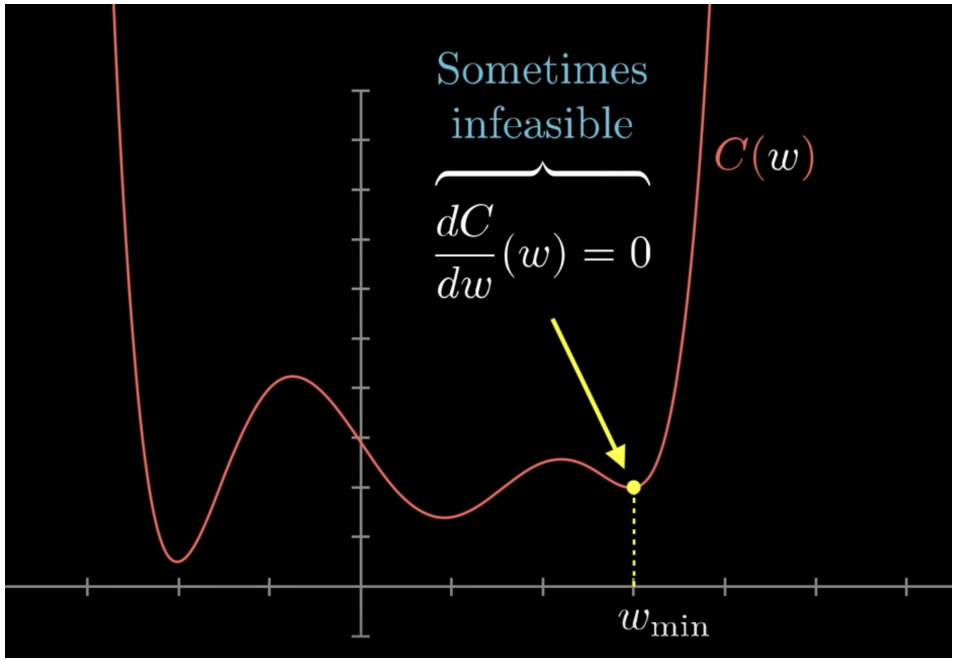


However...

It is not usually that easy...

especially with the crazy cost function (millions of dimensions) formed from a giant neural network. Furthermore, the computational power to do this is near impossible for large networks.

Since each parameter (weight or bias) represents a dimension, and since that would usually constitute > 1 dimensions, we need to find an alternative route to get to the minimum.



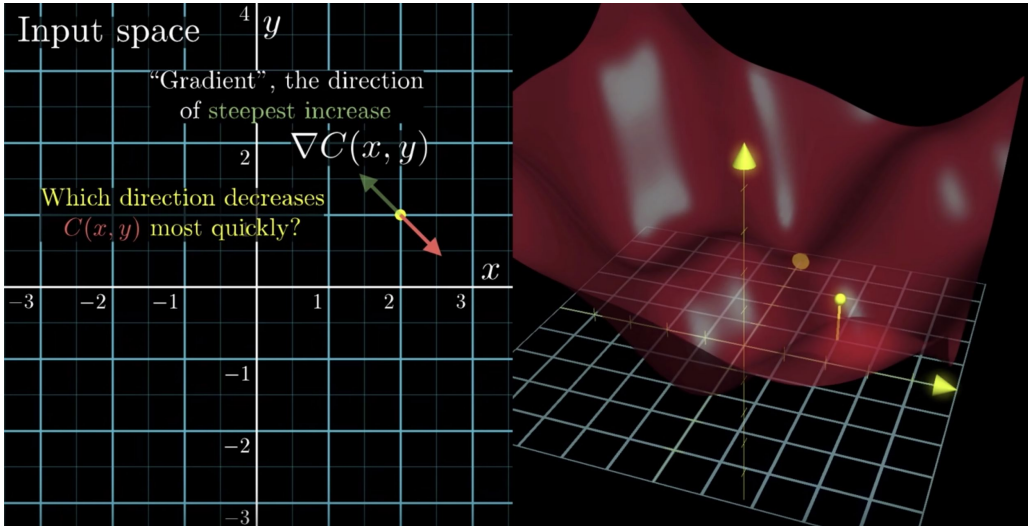
So... any ideas on how we can do this?

Hint:

Gradient Descent!!!

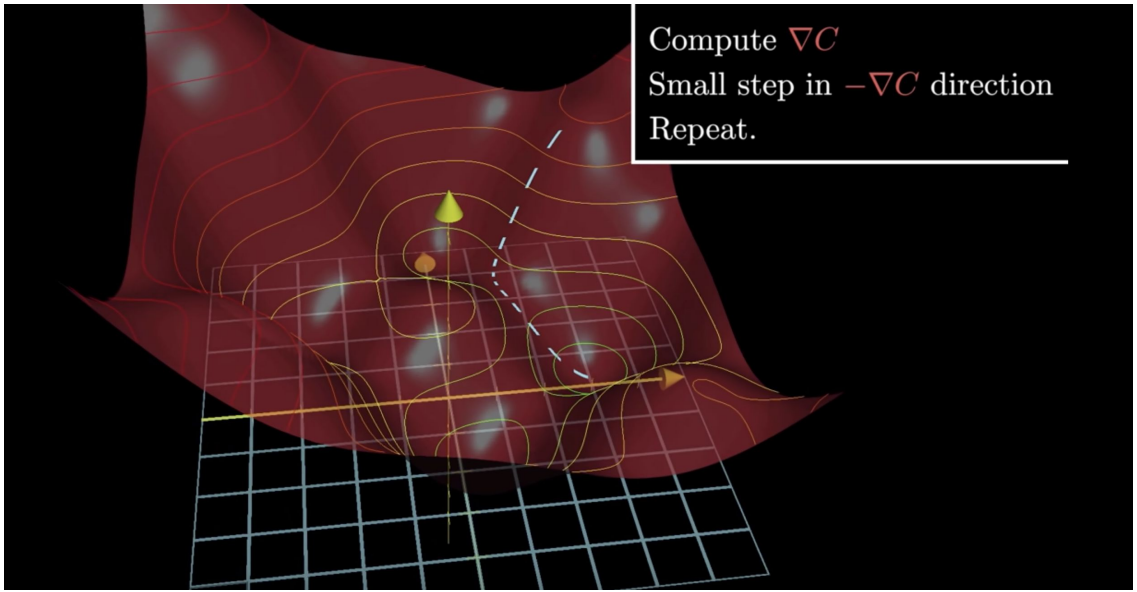
Through a **gradient** (14.6)!

To find what direction to "step" towards to find the minimum point, the negative **gradient** can be calculated which finds the **direction** in which the function decreases most quickly (most steep). This idea is termed "**gradient descent**".



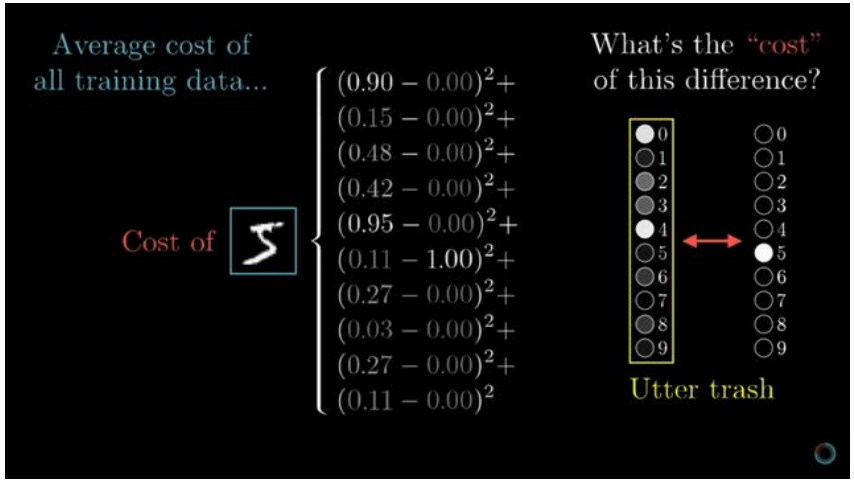
$$\nabla f = \langle f_x, f_y, f_z \rangle = \frac{\partial f}{\partial x}\mathbf{i} + \frac{\partial f}{\partial y}\mathbf{j} + \frac{\partial f}{\partial z}\mathbf{k}$$

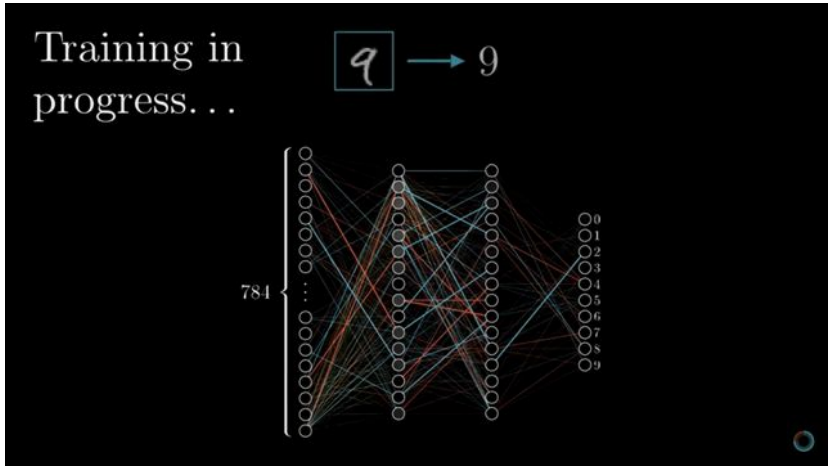
Repeat until minimum is found:



Now... on a Macro Scale:

This process involved only one data point (image). We train the neural network on thousands of data points. Thus, thousands of loss functions need to be minimized. Once this is done, the resulting **overall loss** is the average between all of the loss values.



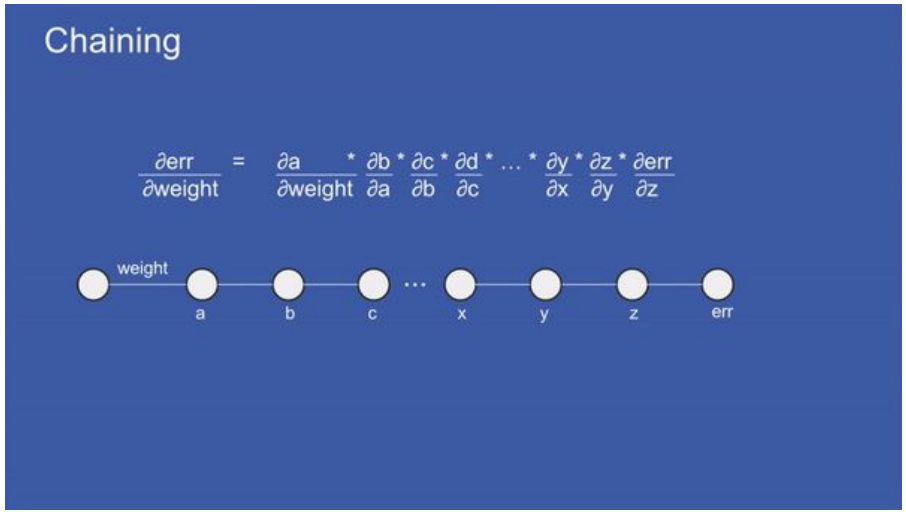


How much will each adjustment change this overall loss (cost)?

You can calculate the change in loss from one data point by "**chaining**" together partial derivatives. Find the change with respect to the weight:

Calculate loss changed up until specific weight (w_i):

$$\frac{\partial loss}{\partial weight} = \frac{\partial loss}{\partial w_i} * \dots \frac{\partial loss}{\partial w_3} * \frac{\partial loss}{\partial w_2} * \frac{\partial loss}{\partial w_1}$$

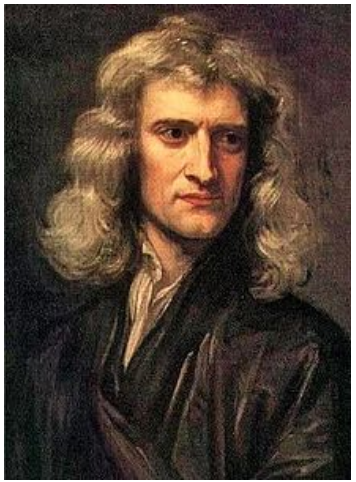


So... How is Calculus Used?

- Gradient descent involves calculating the gradient of a loss function.
- Partial derivatives are used to find how much the loss will change for a certain adjustment of a weight/bias.

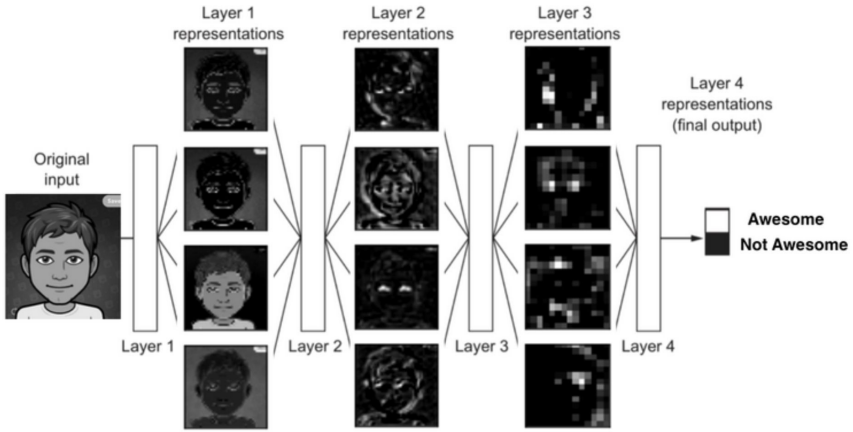
Math in general:

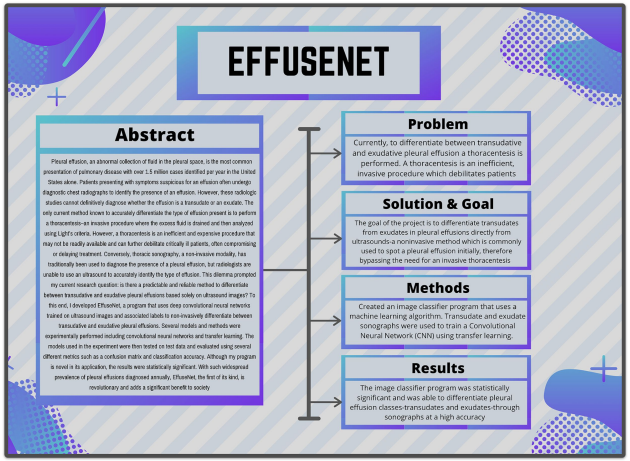
- Summation of weights and biases for the activation of each neuron.

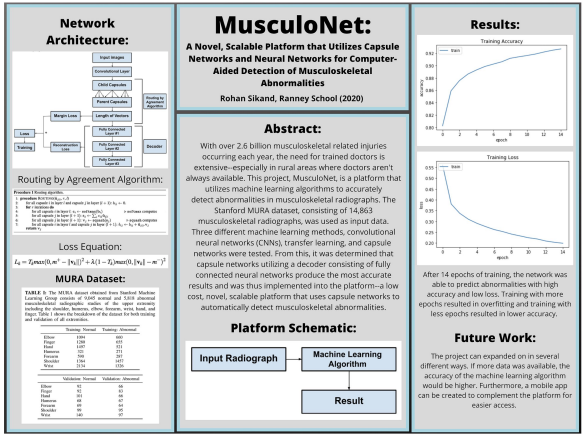


Person in field: Me!

- Applied Machine Learning Research
  - Several research projects
    - **EffuseNet**: Pleural Effusion classifier
    - **MusculoNet**: Computer detection of musculoskeletal abnormalities
- Hope to work on theoretical and applied ML.

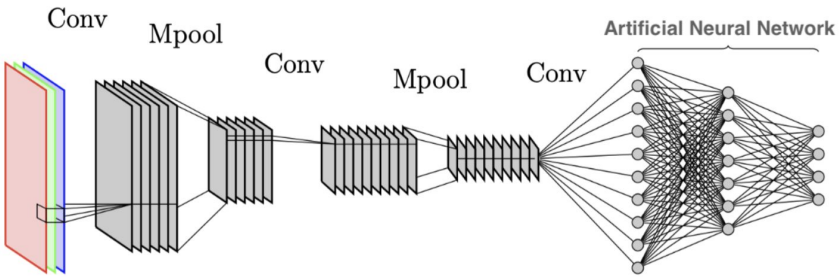






Ok ok... actual researcher in the field: Alex Krizhevsky

- First person to successfully apply a **Convolutional Neural Network** to image data.
  - Placed top 5 in ImageNet image classificatic competition.
  - This sparked a new revolution in computer vision.



Deep Learning Researcher

- **Average salary:** $144,281
- **Level of experience:**
  - PhD in mathematically related field is usually required.
  - Several positions within company/university. More experience = bigger roles.
- **Job description**:
  - Research oriented. Perform theoretical and experimental research. Work closely with other team members.
- **Projects:**
  - Algorithmic
  - Applied to a different field

Company: Zebra Medical Vision

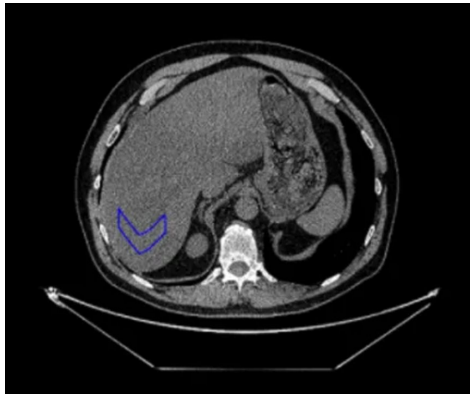
- "Transforming patient care with the power of AI"
- Uses deep learning algorithms for medical image analysis
- Aim is to "provide humanity with automated, accurate, and timely medical image diagnosis and analysis".
- Founded in Israel in 2014

Zebra's Emphysema Algorithm

- Deep learning algorithm analyzes CT Chest studies.
- Detects emphysematous regions in the lungs and quantifies the volume of emphysema.

Zebra's Fatty Liver Algorithm

- Algorithm that analyzes a CT chest/abdomen scan to automatically segment the liver



Resources to learn more:

- **3Blue1Brown** YouTube series
- **Stanford's** CS 230 (Deep Learning) course: lectures, course notes, exams etc.
- Michael Nielson's online e-book: ***Neural Networks and Deep Learning***
- Ian Goodfellow's textbook: ***Deep Learning***
- **arXiv** open research paper repository

Any Questions?

References:

[1] R. Sikand, “Effusenet - deep learning for differentiating between exudative and transudative pleural effusion through ultrasounds,” 2020.

[2] *3Blue1Brown: Gradient descent, how neural networks learn — Deep learning, chapter 2.*

[3] *3Blue1Brown: Backpropagation calculus — Deep learning, chapter 4.*

[4] *Brandon Rohrer: How Deep Neural Networks Work.*

[5] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. USA: Prentice Hall Press, 3rd ed., 2009.

[6] M. A. Nielsen, “Neural networks and deep learning,” 2015. `http://neuralnetworksanddeeplearning.com/`.

[7] Y. LeCun and C. Cortes, “MNIST handwritten digit database,” 2010.