Embedded Machine Learning for Edge Computing

An Intuitive Introduction to Machine Learning

Pamuditha Somarathne

June/2024

What This Week is About

- This course is about Embedded Machine Learning for Edge Computing.
- First, let's look into Embedded Machine Learning.
- This week we only focus on Machine Learning as it is easier to learn Machine Learning when you are not constrained to embedded applications.

Your Instructor

- Pamuditha Somarathne

- Fields: Machine Learning, Human-Computer Interaction
- 2018 - 2023: BSc. Engineering (Electronics and Telecommunication Engineering), University of Moratuwa, Sri Lanka
- 2022 - 2023: Research Affiliate, School of Computer Science, The University of Sydney, Australia
- 2024 – Present: PhD Candidate, School of Computer Science, The University of Sydney, Australia

Content

- Background of Machine Learning (ML)
  - What is learning?
  - Traditional programming vs machine learning
  - Why machine learning?
  - Machine learning terminology
- Neural Networks – How do they work?
- Neural Networks – How do they learn?
- History of Machine Learning
- Latest Machine Learning Models
- Before Using Machine Learning

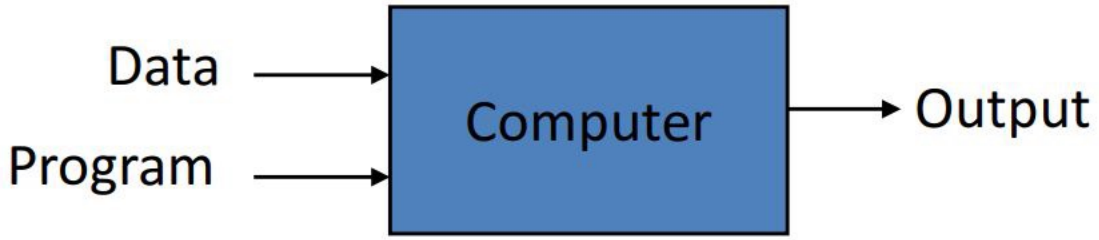
What is “Learning”?

- The activity or process of gaining knowledge or skill by studying, practising, being taught, or experiencing something.
    - Merriam Webster Dictionary

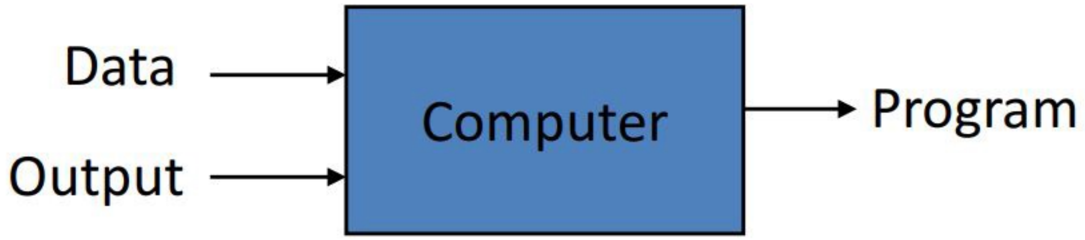
- A computer program is said to learn from ‘experience (E)’ with respect to some class of ‘tasks (T)’ and ‘performance measure (P)’, if its performance at tasks in T, as measured by P, improves with experience E.
    - Tom Mitchell

Traditional vs ML

- Traditional Programming



- Machine Learning



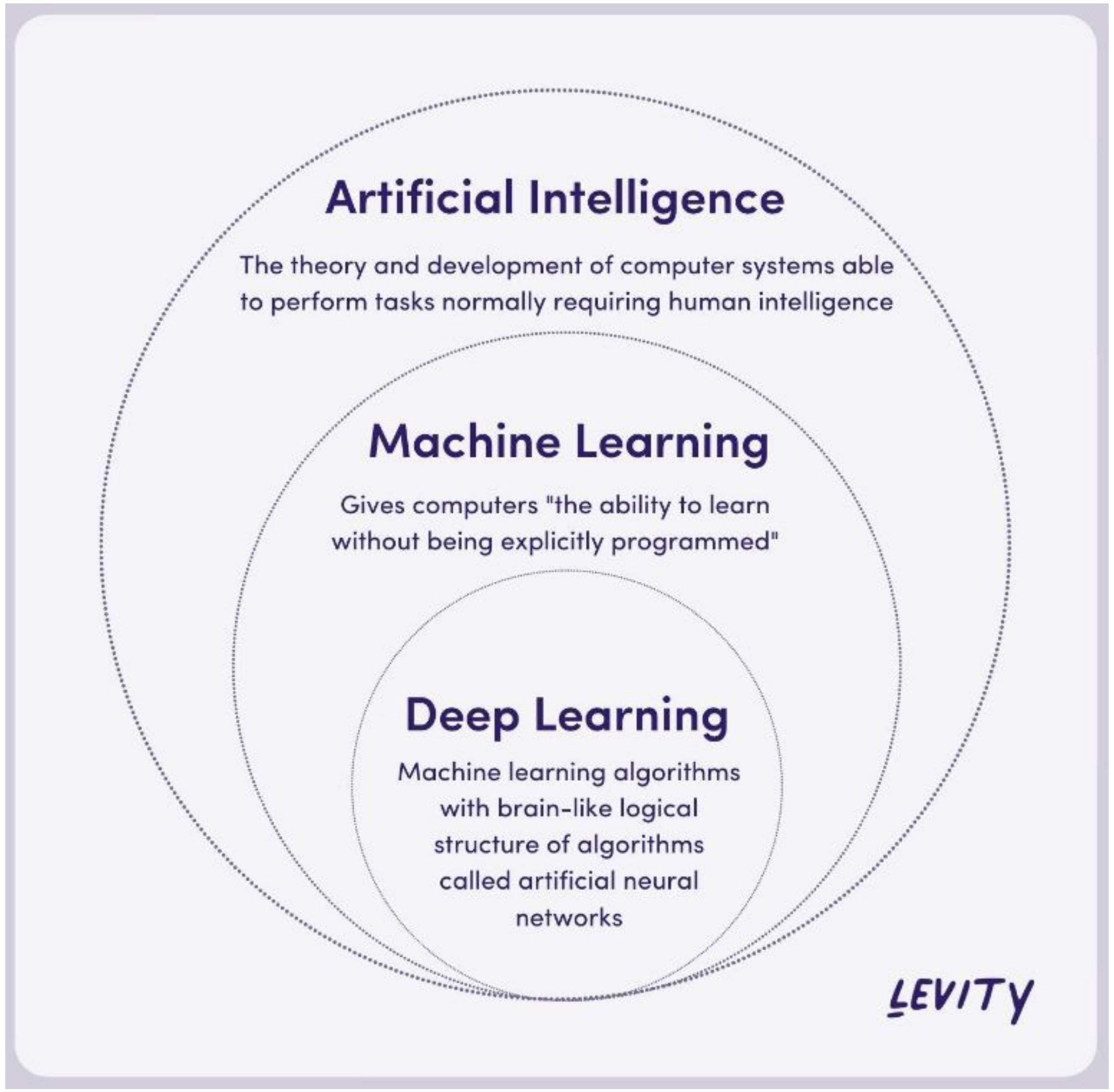
Concept of Learning in ML

- Learning = Improving with experience at some task
  - Task (T)
  - Performance metric (P)
  - Experience (E)

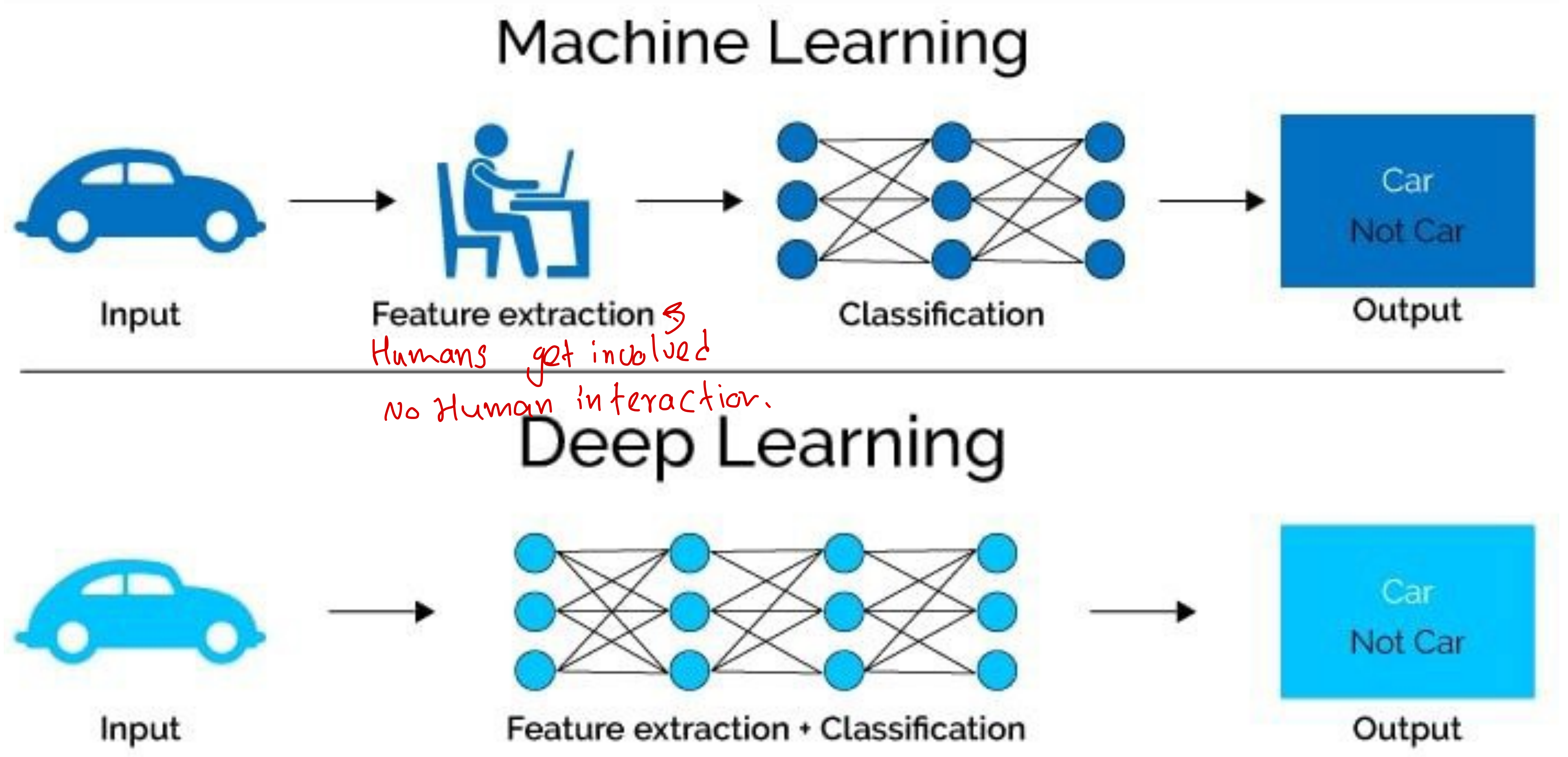
Why ML?

- Non-intuitive problems & solutions: we don't know how to instruct on solving these problems
  - Vision
  - Speech
- Highly complex/expensive/specific systems → Humans may not have enough knowledge
- Adaptability, Customizability, Personalization
- Mimic/replace humans in repetitive tasks
- Perform better than human counterparts and human programmers

AI, ML, DL


Artificial Intelligence
The theory and development of computer systems able to perform tasks normally requiring human intelligence
Machine Learning
Gives computers "the ability to learn without being explicitly programmed"
Deep Learning
Machine learning algorithms with brain-like logical structure of algorithms called artificial neural networks
LEVITY

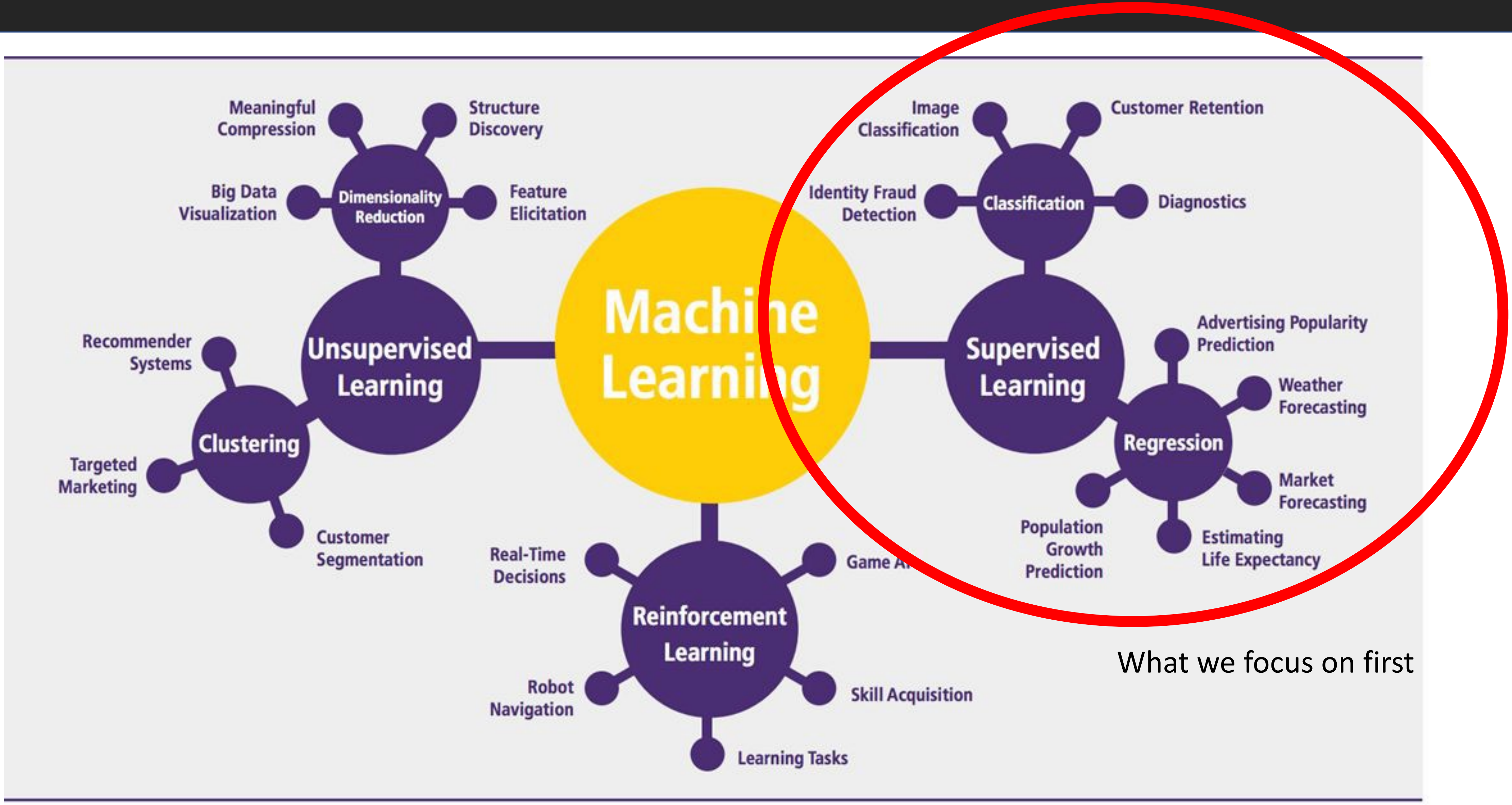
ML vs DL



Humans get involved

No Human interaction.

Overview of ML

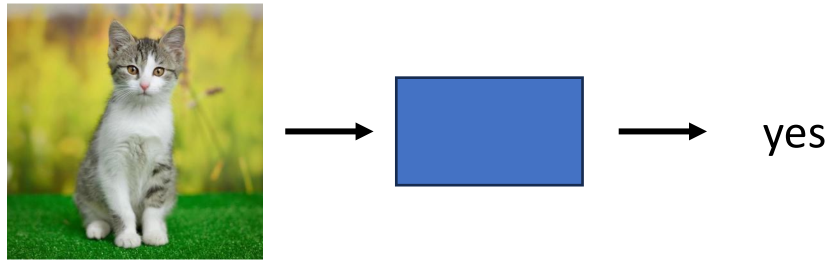


What we focus on first

Supervised Learning

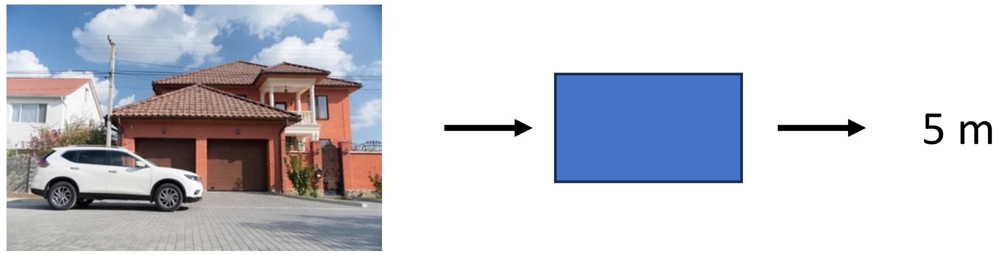
- In supervised learning, we provide the neural network with both inputs and targets (correct results). The model learns to predict outputs that are closer to the target.
- Two types of supervised learning
  - Classifications: picking from a finite number of options

    "Does this figure contain a cat?"



  - Regression: predictions can take any value in a range (range could be infinite as well)

    "How far is the house from the car?"



Content

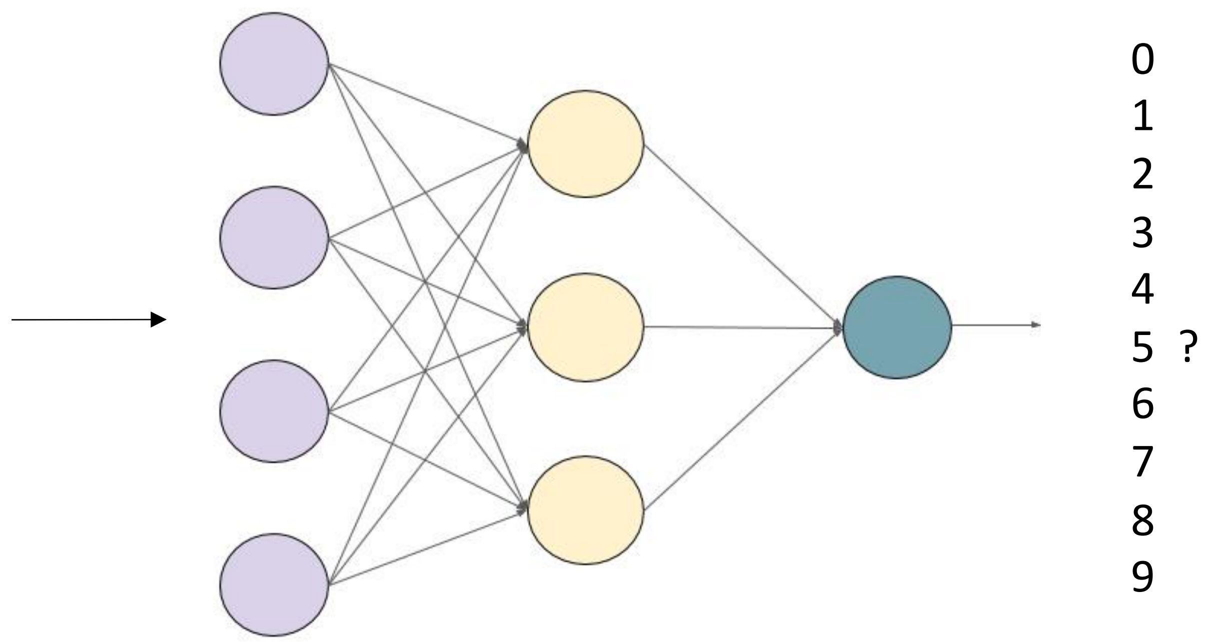
- Background of Machine Learning (ML)
- Neural Networks – How do they work?
  - Different types
  - An example problem
  - Inside a neuron
  - Multi-layer neural networks
- Neural Networks – How do they learn?
- History of Machine Learning
- Latest Machine Learning Models
- Before Using Machine Learning

Neural Networks

- Neural Network – Mimicing Human Brain
- Neural
  - Because we use "Neurons"
  - A neuron holds a number
  - This number depends on its inputs and internal parameters : Each neuron specialized is specific task.
- Network
  - How multiple neurons are connected
  - Different structures
- Different types of neural networks (ways connecting Neurons in Network)
  - Feedforward neural networks
  - Convolutional neural networks
  - Recurrent neural networks
  - Generative adversarial networks
  - Auto-encoders
  - Transformers

An Example

- How do we teach a feedforward neural network to detect handwritten digits



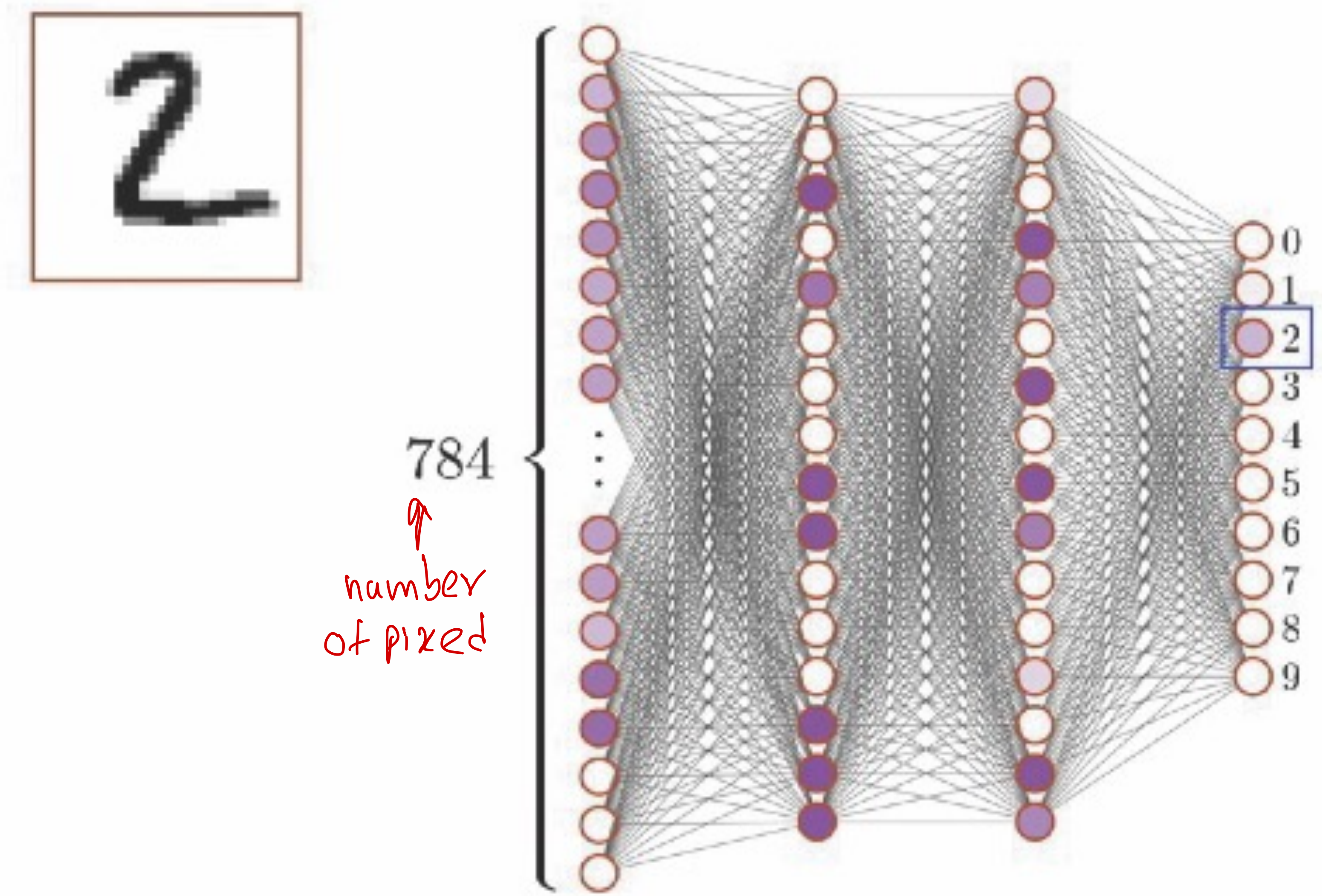
Data

Neural Network

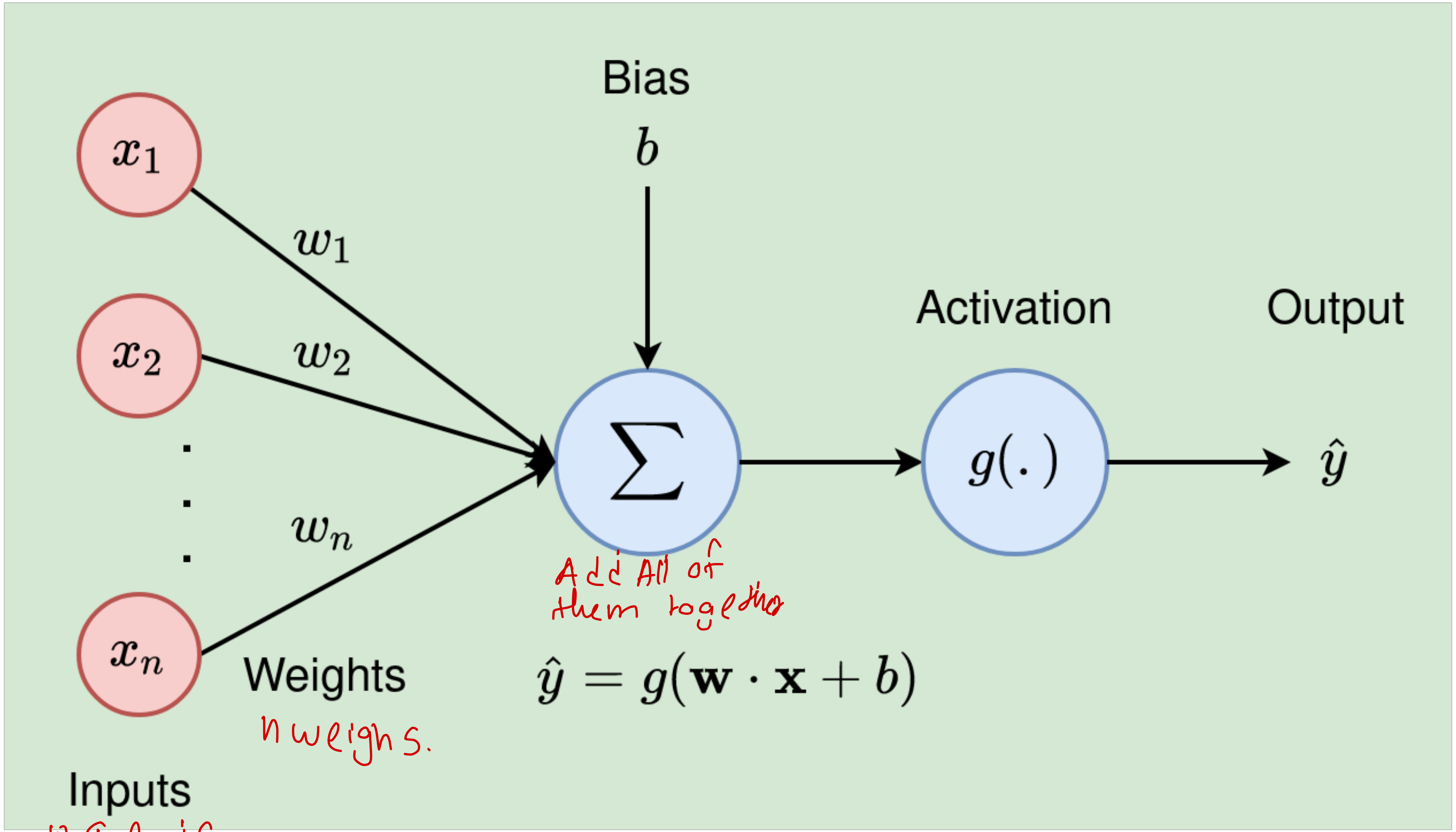
Output

- This is the task (T)

An Example



Inside a Neuron



$$\hat{y} = g(\mathbf{w} \cdot \mathbf{x} + b)$$

n Inputs

Weights & Bias

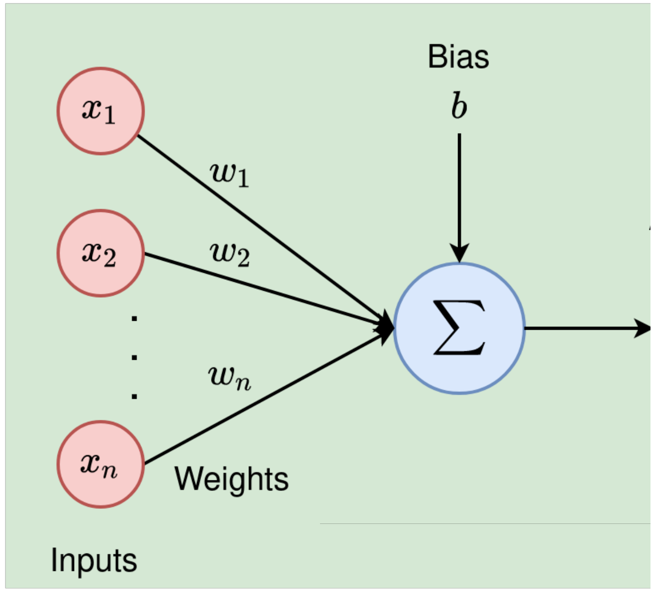
- The neuron sees its inputs through the weights
- Each input path has a weight
  - $[w_1, w_2, w_3, \dots, w_n]$ are the weights
  - Effect of weights and inputs: $w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n$
  - If we combine the weights $[w_1, w_2, w_3, \dots, w_n]$ into $\boldsymbol{w}$ vector and inputs in $\boldsymbol{x}$ vector,

$$w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n \rightarrow \boldsymbol{w} \cdot \boldsymbol{x}$$

weighted sum.

- Sometimes, it is good to add a bias
  - A learnable constant (b) - Doesnt depend on input
- We sometimes call weights and biases

  as 'parameters' or just 'weights'



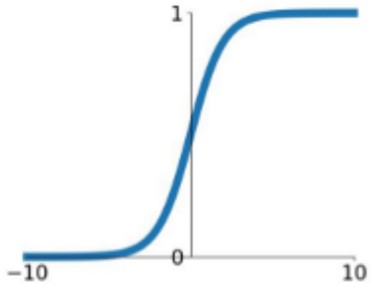
Activations

Non linear functions.

- A non-linear function
  - Sigmoid
  - Tanh
  - ReLU
- Limits the range of the neuron's output
  - Helps following neurons with their input's size
- Adds extra complexity to the neuron
  - Helps to make more complex decisions

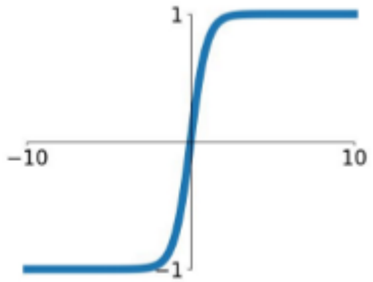
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



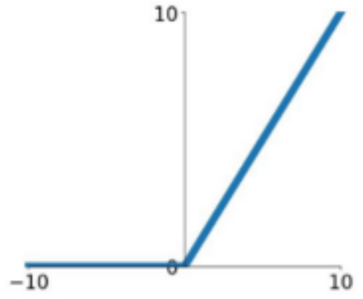
tanh

$$\tanh(x)$$



ReLU

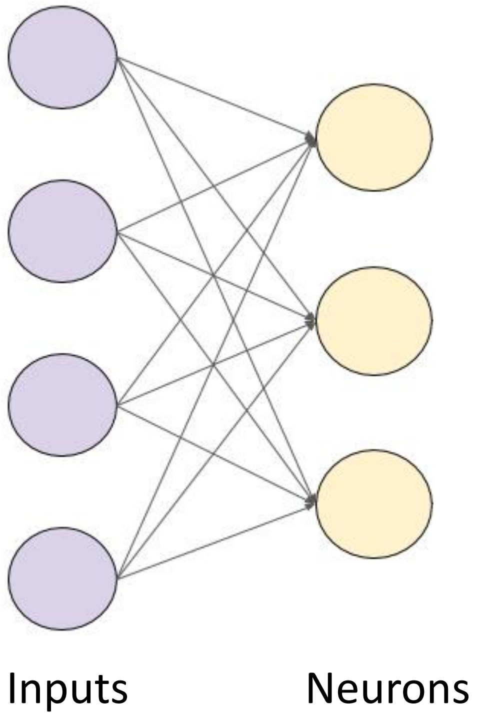
$$\max(0, x)$$



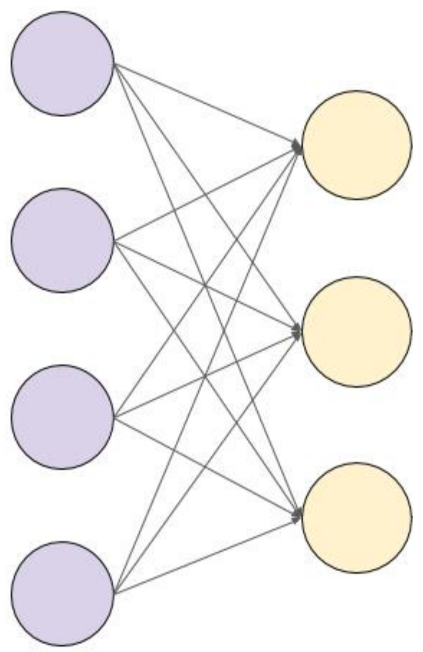
Layer of Neurons

- Giving the same input to multiple neurons helps as each neuron could learn something different.
- Each neuron has its own weights
- Let ' s say each neuron has weight vectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3$
- We can combine these into a weight matrix $\boldsymbol{W}$
- We combine biases b_1, b_2, b_3 from each neuron into $\boldsymbol{b}$
- Then we can write the whole layer's operation as

$$\hat{\boldsymbol{y}} = g(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$$



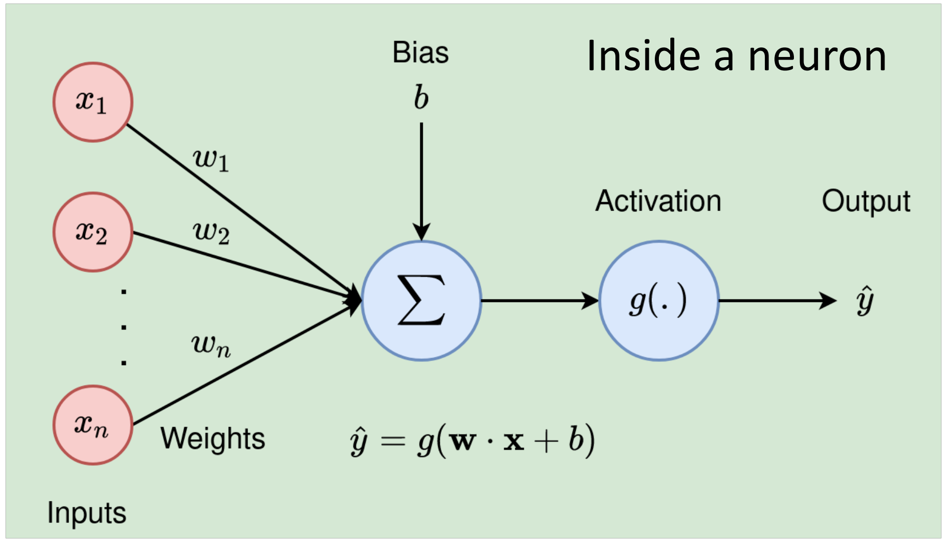
Layer of Neurons



$$\hat{\boldsymbol{y}} = g(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$$

$$\begin{bmatrix} \hat{y}_0 \\ \hat{y}_1 \\ \vdots \\ \hat{y}_{k-1} \end{bmatrix} = g\left(\begin{bmatrix} w_{0,0} & w_{0,1} & \cdots & w_{0,n-1} \\ w_{1,0} & w_{1,1} & \cdots & w_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ w_{k-1,0} & w_{k-1,1} & \cdots & w_{k-1,n-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{n-1} \end{bmatrix} + \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{k-1} \end{bmatrix} \right)$$

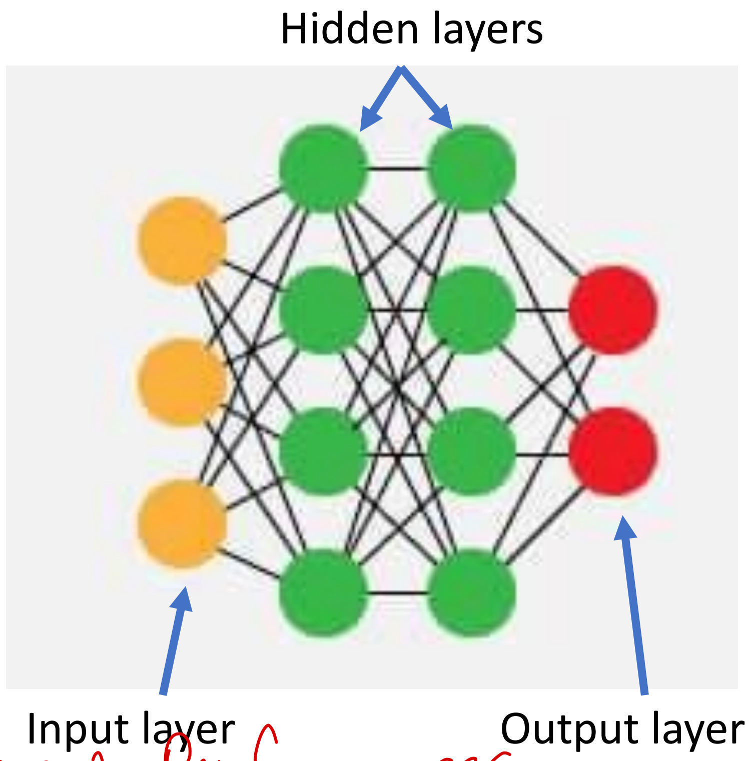
* vectorized calculations.



Multiple Neuron Layers

- We can join multiple layers one after the other
- We give output of each layer to the next layer
- Each layer can learn more complex patterns than the previous layer
- This is called a Multi-Layer Perceptron (MLP)
- We call the layers:
  - Input layer – take input (shaped Determied by size of input)
  - Hidden layers – hidden from the outside world
  - Output layer – gives output (Shape Determied by expected output)

*we can play around the number of Hidden layers and number of Neurons in each layers to get different versions of model with different Performances.



Content

- Background of Machine Learning (ML)
- Neural Networks – How do they work?
- Neural Networks – How do they learn?
  - What is ‘learning’ for a neural network
  - Dataset
  - Cost function
  - Gradient Descent
  - Evaluation
- History of Machine Learning
- Latest Machine Learning Models
- Before Using Machine Learning

A Learning Neural Network

- For a neural network, learning means finding the best weights and biases for the given task.
- The learning process is called ‘training the neural network’.
- During training, you provide the model with different input and target pairs.
- For each input, we compare the output of the model to the target and adjust the weights and biases to improve the output slightly.
- Doing this multiple times over many samples results in a good set of weights and biases for the task.

Dataset

Experience component of learning

- The dataset is all the data related to the task we have.
- It represents Experience (E) in
  - Learning = Improving with experience at some task
- We separate the available dataset into training, validation, and test sets
  - Train set: used for training
  - Validation set: used to compare between different versions of our model
  - Test set: used to evaluate the real-world performance of the model
- If we are developing a model for a real-world application, the best practice is to use the test only when we are confident that the model is good.

Back-propagation

- For each input in the dataset, we compare the output of the model to the target and adjust the weights and biases to improve the output slightly.
- The amount that we adjust each weight and bias is determined by an algorithm called backpropagation.
- Backpropagation has two steps,
  - Cost function/loss function
  - Gradient Descent

Cost Function

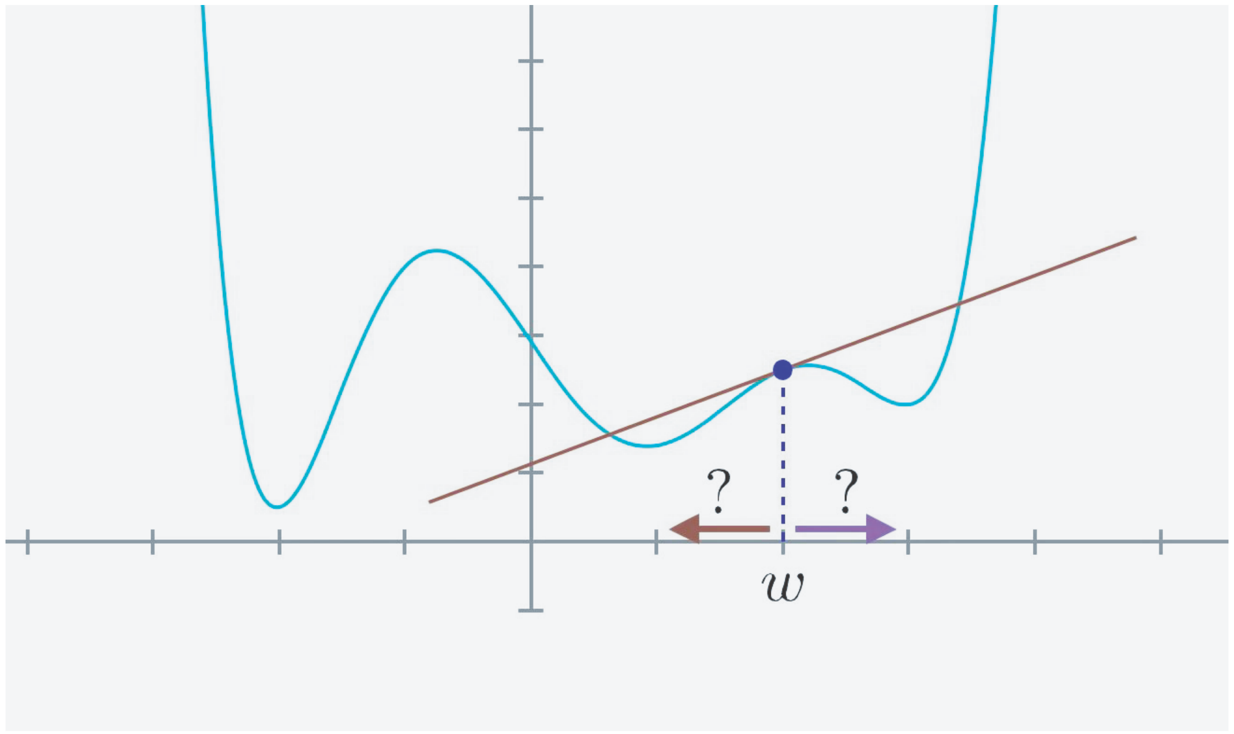
- The cost function/loss function is a function that is used to
  - measure how good or bad our model's output (also called prediction) is compared to the target (also called ground truth).
- The cost function takes in the output and target and gives us a single value that is a measure of the difference or error between the output and target.
- Usually, when the cost function result is higher, that means the model is not working properly.
- Therefore, we adjust the weights and biases in such a way that the cost function reduces.

Gradient Descent

- The gradient descent algorithm is used to find which way the weights and biases should be adjusted (i.e. whether to increase or decrease) in order to reduce the cost function.
- The gradient descent algorithm was developed using calculus in mathematics.
- Gradient Descent
  - Gradient refers to the slope/derivative of the cost function with respect to a weight.
  - Descent means that we step in the direction that gives maximum reduction in cost.
- This is similar to when you descend from a hill. If you wanted to descend faster, you would pick the direction with the maximum slope.

Gradient Descent

- Let’s consider a simplified situation. Imagine our model had only one weight, and the cost function changed like the following graph on the left when we changed that parameter.



- If we are currently at the blue point. We can apply gradient descent algorithm to decide that we should go to the left to minimize the cost function.

Gradient Descent

- When we consider a neural network that is usually used in real-world applications, it has a lot more than one parameter and calculating the direction that each parameter should be updated is quite difficult and time-consuming.
- Fortunately, there are libraries/frameworks that we can use to achieve this.
- PyTorch, Tensorflow, and JAX are some of those libraries that work with python.
- These libraries have implemented the gradient descent algorithm so we just have to give them a loss function and tell the library to do one step of gradient descent. The library automatically does the gradient calculation and weight update.

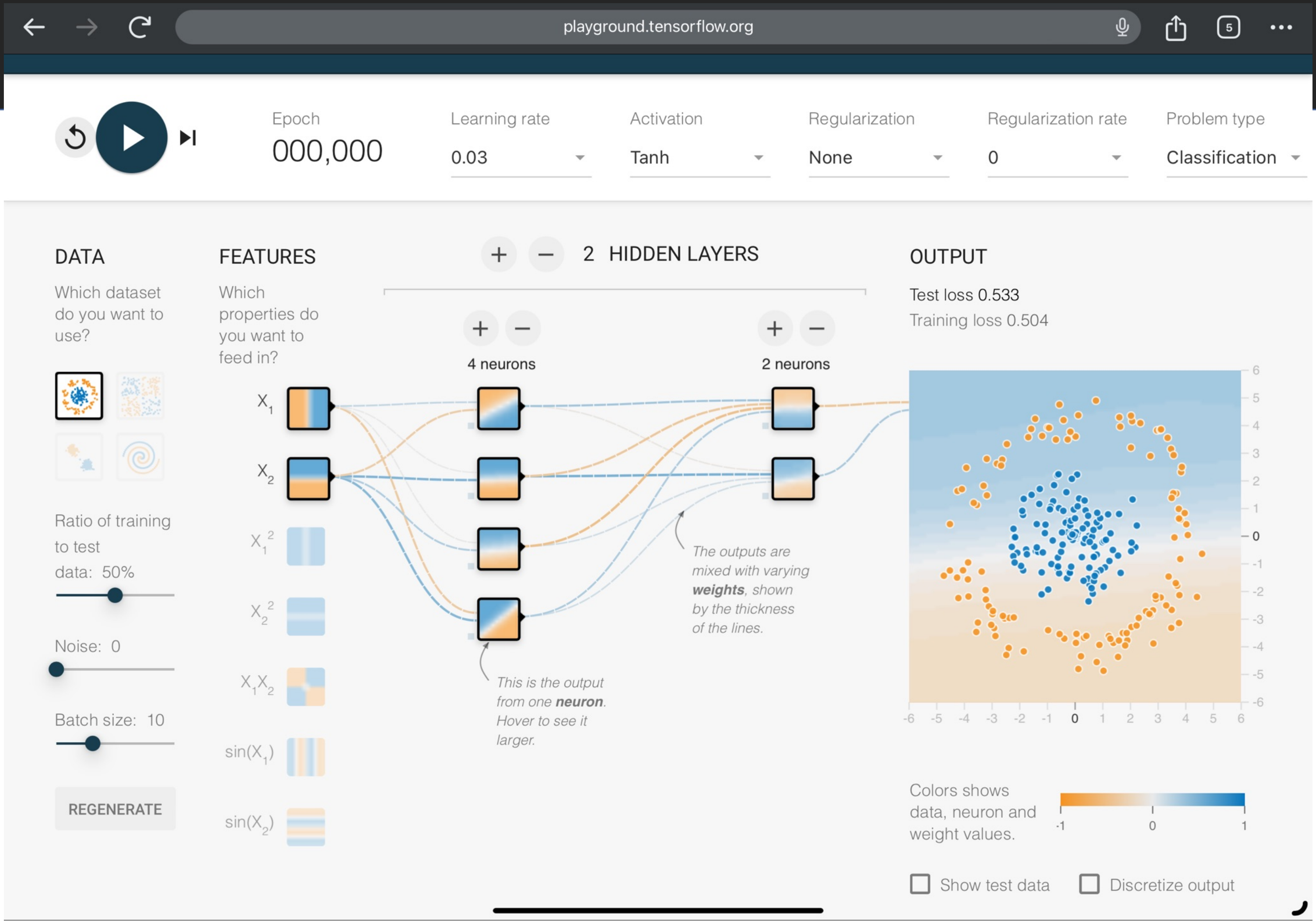
Measuring Performance

- During and after training, we can measure the neural network's performance using many metrics.
- There can be many metrics, including accuracy, precision, sensitivity, mean-square error, heat-maps, etc.
- The cost function's cost result can also be used as a way to measure performance.
- However, not all metrics can be used as a cost function.

Additional Resources

- This website has an interactive playground where you can build simple neural networks with properties.
  - https://playground.tensorflow.org/

- This YouTube video, "But what is a neural network?" and the website has a very visual and intuitive explanation of how a neural network works and learns. The channel is full of good mathematical content. Have a look if you have free time.
  - https://www.youtube.com/watch?v=aircAruvnKk
  - https://www.3blue1brown.com/



playground.tensorflow.org

Epoch 000,000 | Learning rate 0.03 | Activation Tanh | Regularization None | Regularization rate 0 | Problem type Classification

DATA

Which dataset do you want to use?

Ratio of training to test data: 50%

Noise: 0

Batch size: 10

REGENERATE

FEATURES

Which properties do you want to feed in?

2 HIDDEN LAYERS

4 neurons

2 neurons

*The outputs are mixed with varying **weights**, shown by the thickness of the lines.*

*This is the output from one **neuron**. Hover to see it larger.*

OUTPUT

Test loss 0.533

Training loss 0.504

Colors shows data, neuron and weight values.

☐ Show test data ☐ Discretize output

Epoch – How many steps in training

Content

How It All Started

Turing Machine

Perceptron

Backpropagation

TD-Gammon
First computer to play Backgammon

Deep Blue
First computer to beat a world champion at chess

ConvNets & GAN

Popularisation of Deep Learning

1940 **1950** **1960** **1970** **1980** **1990** **2000** **2010** **2020**

A. Turing

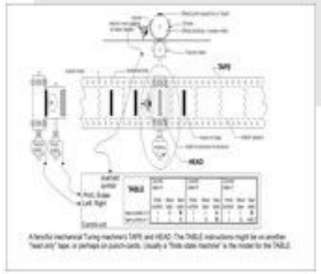
F. Rosenblatt

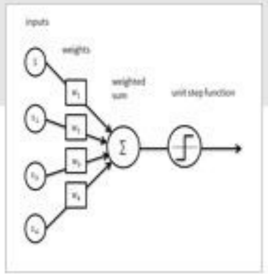
D. Rumelhart – G. Hinton – R. Williams

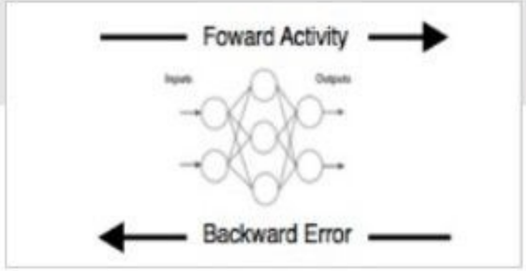
R. Sutton – C. Watkins

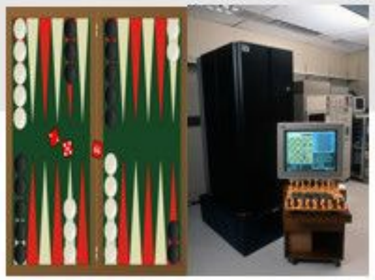
Y. Lecun

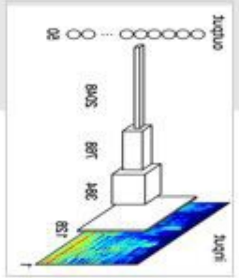
Y. Bengio - I. GoodFellow – A. Courville

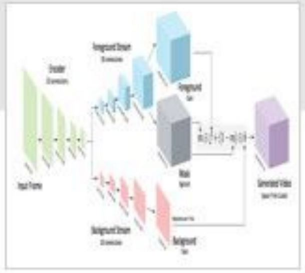


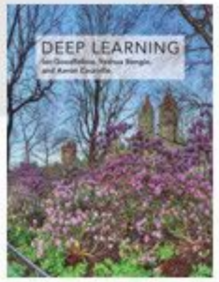












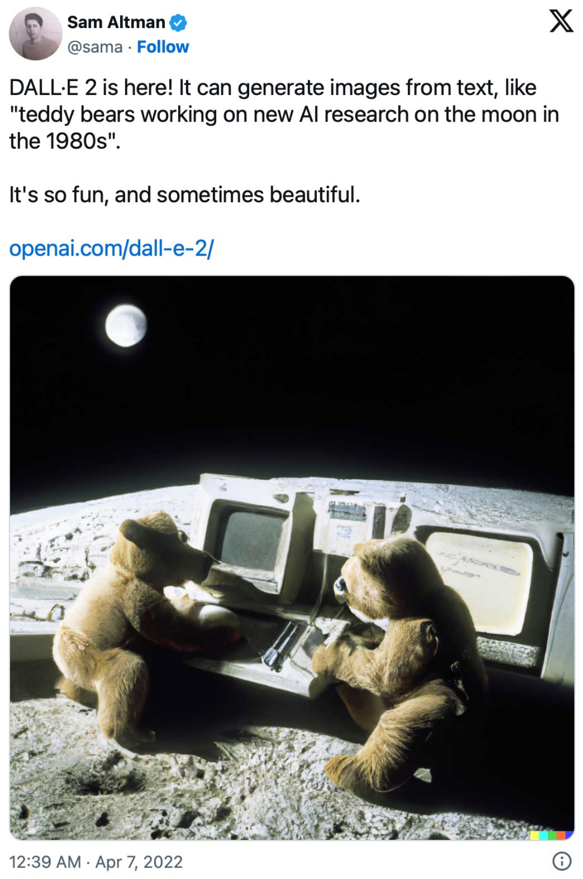
Content

State-of-the-Art Models: ChatGPT

- ChatGPT is one of the most popular machine learning models in the current world. It was developed by OpenAI.
- ChatGPT models and GPT backbone models are text generation models.
- They can understand the text, image, and audio data that we give and provide text feedback.
- ChatGPT can be accessed using its website or its python library.
- ChatGPT can be used for embedded applications as well if the device has access to the internet.
- ChatGPT is an amazing technology, but it is not perfect. It still makes mistakes and incorrect results.

State-of-the-Art Models: DALL-E

- Dall-E is an image generation model developed by OpenAI.
- We can input text prompts and it will generate images according to the provided text.
- It is pretty good at creating images.


Sam Altman
@sama · Follow

DALL·E 2 is here! It can generate images from text, like "teddy bears working on new AI research on the moon in the 1980s".

It's so fun, and sometimes beautiful.

openai.com/dall-e-2/

12:39 AM · Apr 7, 2022

Content

Before Using ML

- Limitations of machine learning
  - Ethics
  - Data
  - Interpretability – blackbox models (explainable AI)
  - Technical debt
    - In software development and other information technology fields, technical debt is the implied cost of future reworking required when choosing an easy but limited solution instead of a better approach that could take more time.
  - Better alternatives
    - Simpler = Better
    - A simple solution that is 90% accurate but can be built in a week is preferred by everyone over a ML model that takes months to build but achieves 95% accuracy.
- https://www.springboard.com/blog/data-science/when-not-to-use-ml/
- https://towardsdatascience.com/4-reasons-why-you-shouldnt-use-machine-learning-639d1d99fe11

A Recipe for ML

1. Should I use ML on this problem? Is there a pattern to detect? Can I solve it using mathematics or traditional programming? Do I have enough data?
2. Gather and organise data: preprocessing, cleaning, visualising, analysing
3. Establish a baseline to compare your progress
4. Choose a model, cost function, etc.
5. Optimise the model
6. Hyperparameter search $\rightarrow$ Number of Layers, Number of Neurons - - .
7. Analyse performance & mistakes, and iterate back to step 4 or 2.

Conclusion

- This week, we learned the background of machine learning.
- We looked into how a simple machine learning model, “the neural network”, learns from its data.
- We also learned a bit about the history and current best models.
- Finally, we learned what we should consider when using machine learning.

- This week’s hands-on session includes building a simple neural network using the theory we learned. We will be using the PyTorch library and Google Colab platform.
- Looking forward to seeing you there.

Conclusion

- This week, we learned the background of machine learning.
- We looked into how a simple machine learning model, “the neural network”, learns from its data.
- We also learned a bit about the history and current best models.
- Finally, we learned what we should consider when using machine learning.

- This week’s hands-on session includes building a simple neural network using the theory we learned. We will be using the PyTorch library and Google Colab platform.
- Looking forward to seeing you there.

Conclusion

- This week, we learned the background of machine learning.
- We looked into how a simple machine learning model, “the neural network”, learns from its data.
- We also learned a bit about the history and current best models.
- Finally, we learned what we should consider when using machine learning.

- This week’s hands-on session includes building a simple neural network using the theory we learned. We will be using the PyTorch library and Google Colab platform.
- Looking forward to seeing you there.

Conclusion

- This week, we learned the background of machine learning.
- We looked into how a simple machine learning model, “the neural network”, learns from its data.
- We also learned a bit about the history and current best models.
- Finally, we learned what we should consider when using machine learning.

- This week’s hands-on session includes building a simple neural network using the theory we learned. We will be using the PyTorch library and Google Colab platform.
- Looking forward to seeing you there.