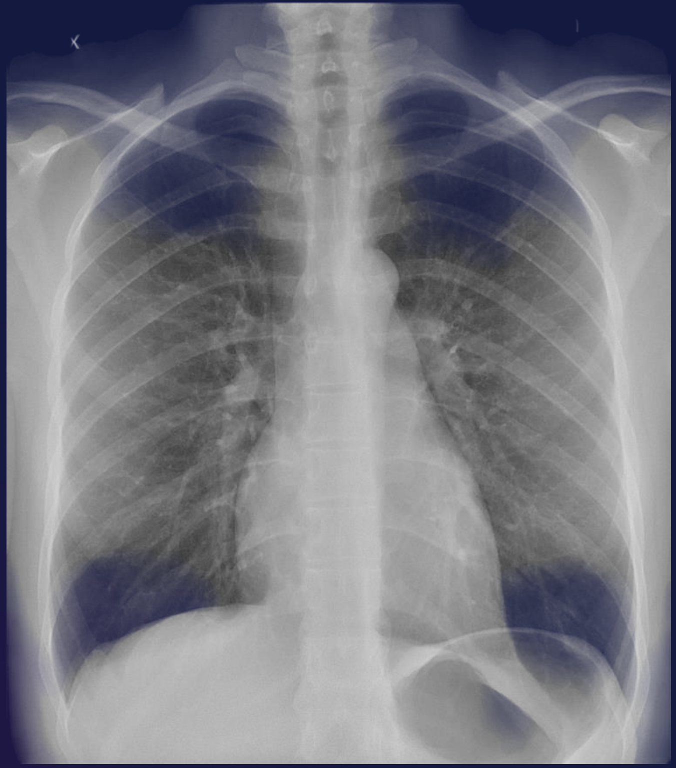
Custom Neural Network Framework

By Johnny Lee

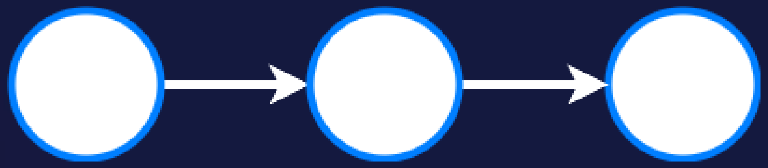
Why Neural Networks Matter for Health?

- Neural Networks can act as radiologist assistants.
- Example: trained a network on 19,500 chest X-rays, validated/tested on 8,500 (viral and bacterial pneumonia, COVID-19, TB, normal, and undiagnosed opaque lungs).
- Achieved 94% accuracy in disease classification.



What is a Neural Network?

- A neural network is inspired by the human brain.
- Both have **neurons** that take in signals, process them, and pass them on.
- By connecting many neurons together, networks can learn complex patterns.



Inputs and Outputs of a Neural Network

Inputs

- Inputs (often called features) are the data given to the network to make a prediction.

Task: Classification

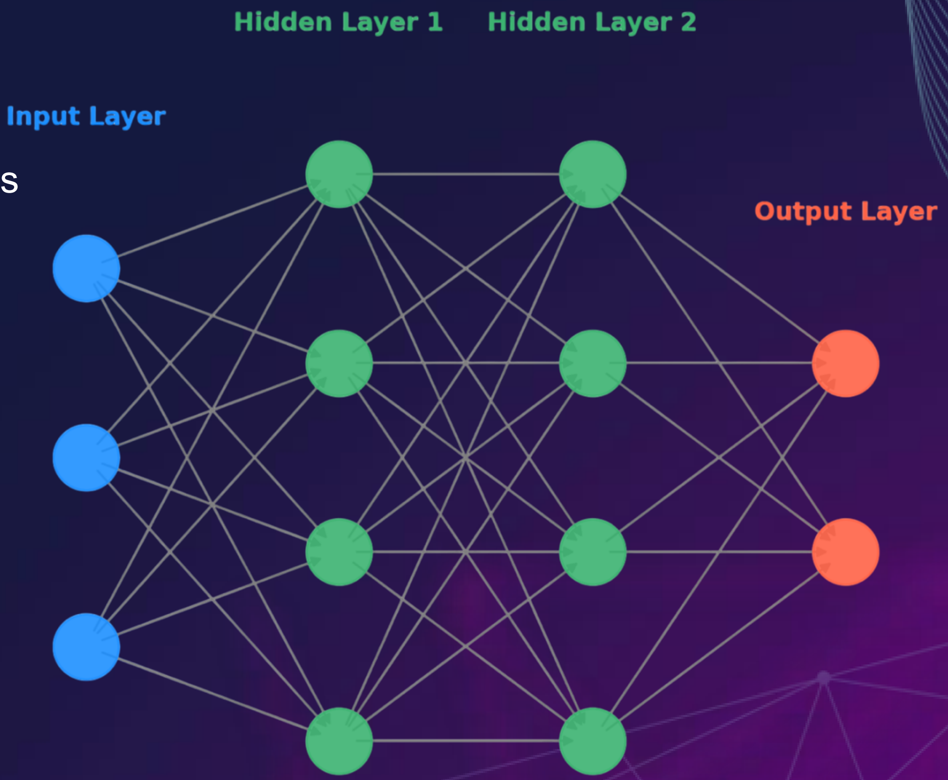
Outputs

- Outputs are the prediction(s) produced by the network.

Task: Regression

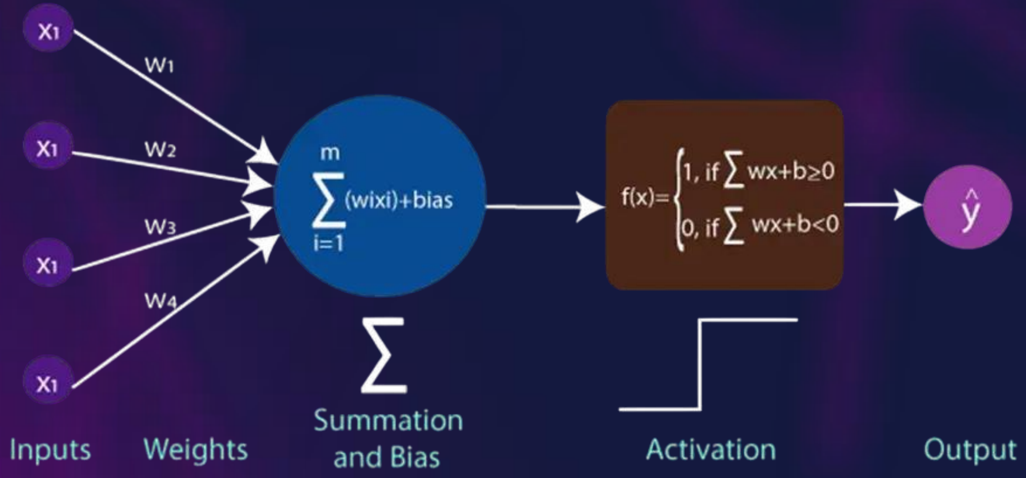
The Multilayer Perceptron (MLP)

- Simplest type of neural network
- Consists of layers of neurons (circles in the diagram) that process data from the input or previous layer and pass it to the next layer.
- Architecture:
  - **Input layer**: takes in features.
  - **Hidden layer(s)**: find patterns in the features.
  - **Output layer**: makes the prediction.



Inside a Neuron

- A neuron receives the output from each neuron in the previous layer as input.
- Each input has a weight, which represents its importance.
- The neuron adds these weighted inputs together.
- A bias term is added to shift the result.
- The result passes through an activation function, which decides if the neuron fires and introduces non-linearity to the network.



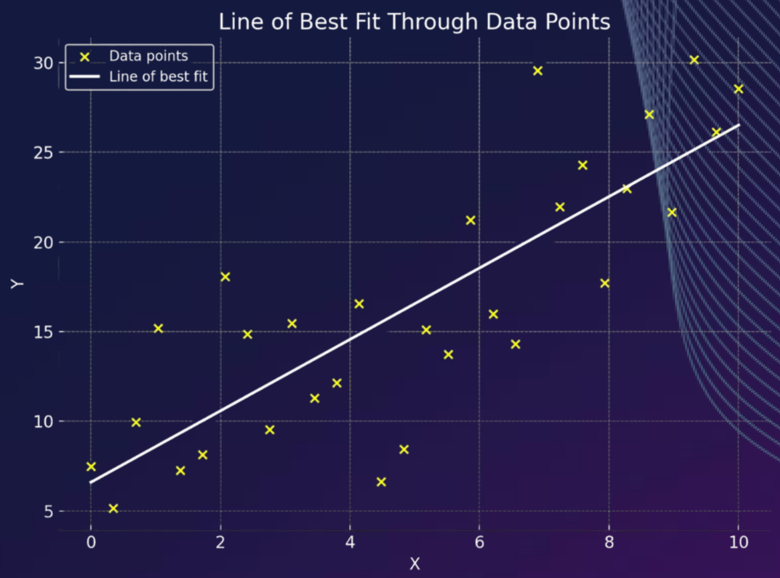
How Neural Networks learn

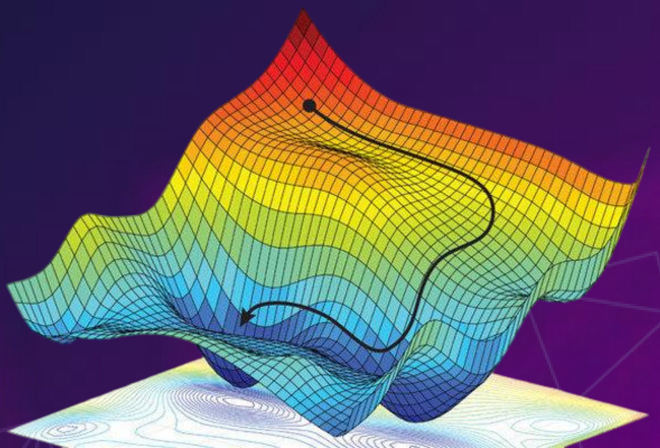
- During training, predictions are passed to a loss function, which measures how wrong they are.

- The network learns by adjusting the weights and biases to minimize this loss through a process called backpropagation.

- This process is identical to finding a line of best fit in 2D regression.

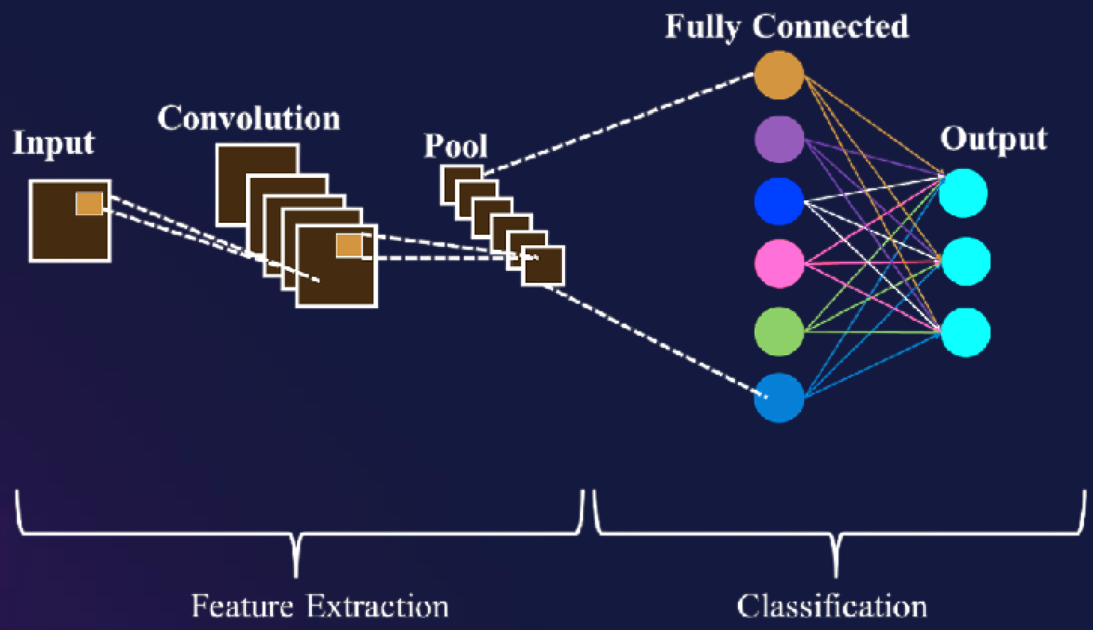
- It can also be pictured as a ball rolling downhill, settling at the lowest point of error.





What is a Convolutional Neural Network (CNN)

- CNNs are a specialized neural network for image, video, and audio data.
- They scan small patches of data at a time, which helps them detect edges, shapes, and patterns efficiently.
- The scanners are called filters (or kernels) and each one of them learns a pattern from the data.



Training Features of The Framework

- Supports both tabular and image data.
- Handles categorical and numerical inputs.
- Built-in data loaders for easy input.
- Provides data splitters to divide data into training, testing, and validation sets.
- Provides data transformers to normalize, scale, or resize data.
- Supports multiple layers, losses, and activations.
- Supports classification and regression tasks.

Testing & Monitoring Features of The Framework

Monitoring Features:

- Command-line interface (CLI) shows each step of data preparation and training.
- Progress bar displaying progress through each epoch.
- Tracks running loss, evaluation metric, and time per epoch in seconds.
- Prints validation loss and performance metric after each epoch.
- Early stopping halts training if validation performance stops improving and keeps the best model.

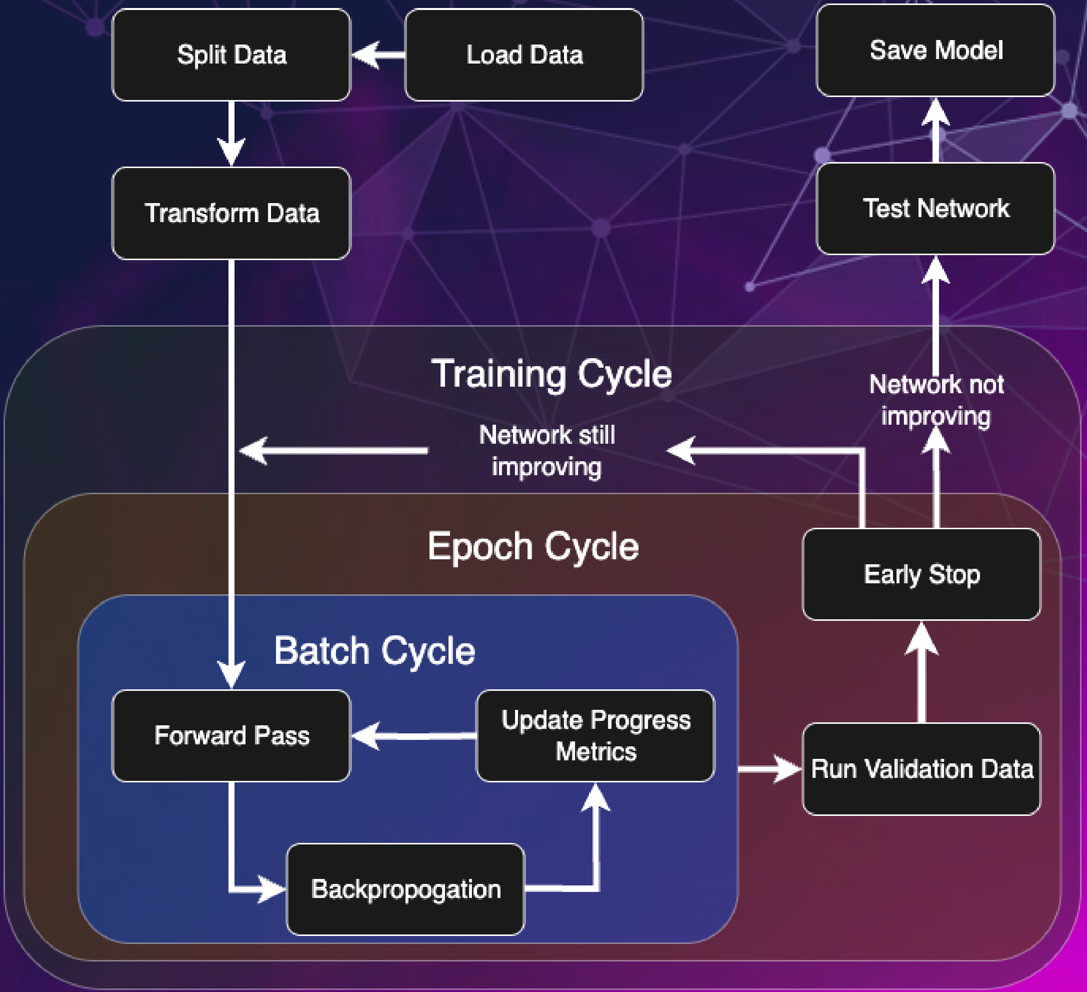
Testing Features:

- Easily evaluate a trained network on a test set.
- Supports multiple performance metrics for classification and regression.

```
============================================================
              🧠 C++ NEURAL NETWORK TOOLKIT
         MLP & CNN For Classification & Regression
============================================================

📥 Loading training data from: "mnist_png".
[✓] Scanning Image Directories.
[✓] Extracting Images.
[✓] Extracting Targets.
------------------------------------------------------------

🎨 Transforming 68992 images.
[✓] Resizing & Normalizing images.
------------------------------------------------------------

✂ Splitting 68992 samples: 55197 | 13795
[✓] Splitting data with stratification.
------------------------------------------------------------

✂ Splitting 55197 samples: 49682 | 5515
[✓] Splitting data with stratification.
------------------------------------------------------------

Epoch: 1/2
49682/49682 |██████████████████████████████████████████████████| Accuracy: 81.76%| Avg Loss: 0.73 | Elapsed: 0.87s
Avg Validation Loss: 0.38 | Validation Accuracy: 89.86%

Epoch: 2/2
49682/49682 |██████████████████████████████████████████████████| Accuracy: 90.66%| Avg Loss: 0.34 | Elapsed: 0.76s
Avg Validation Loss: 0.30 | Validation Accuracy: 91.57%
------------------------------------------------------------

[✓] Model saved successfully as "models/test.nn".
------------------------------------------------------------

Test Accuracy 91.71%.
```

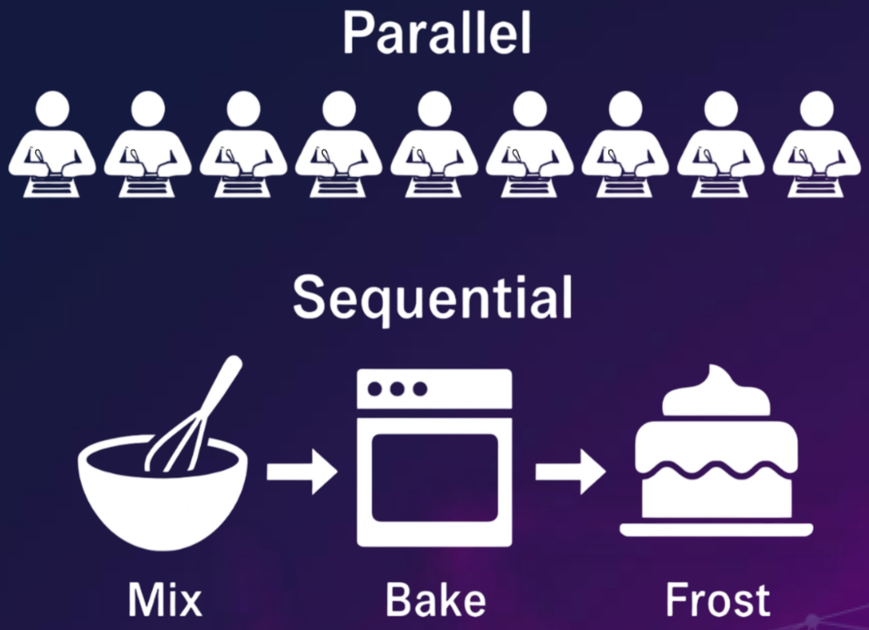
Data Flow Through A Neural Network

- Training data is split into batches, and an epoch ends once all batches are processed.
- After each epoch we run validation. If the network is not improving, we test and save the model. If it is improving, we continue training for another epoch.



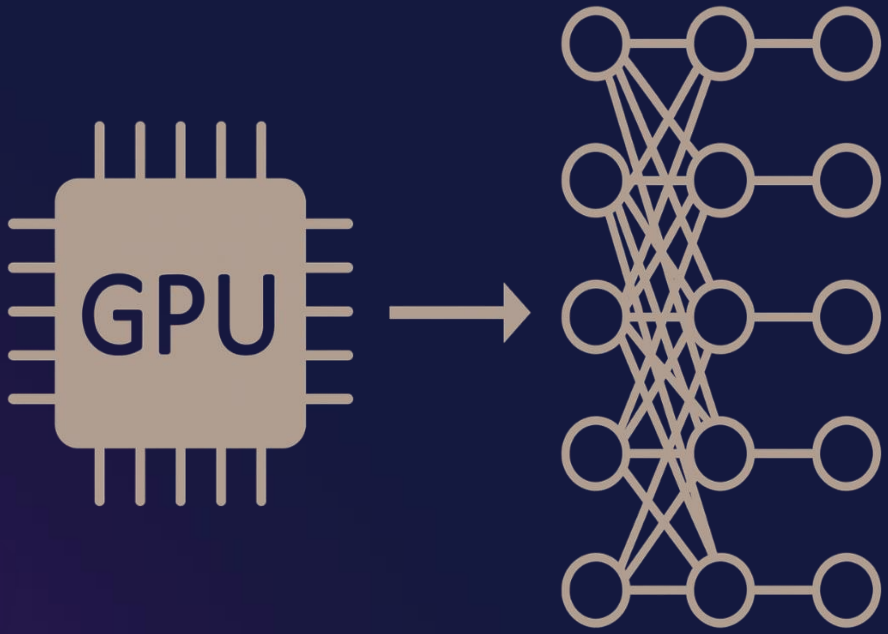
What is GPU Computing

- GPUs perform thousands of small calculations all in parallel.
- CPUs perform only a limited number of calculations in parallel.
- Parallelizable: grading 100 exams with 100 graders.
- Not parallelizable: baking a cake (must be done step by step).



GPU Computing for Neural Networks

- Neural networks have many independent neurons in each layer.
- Each neuron can be computed in parallel.
- But the layers themselves must be computed sequentially as each layer's output is another layer's input.



How the Framework Uses GPU Computing

- Forward pass for each layer is GPU accelerated by computing each neuron simultaneously.
- Backpropagation is also GPU-accelerated, with neurons updated in parallel.
- This parallelism massively speeds up the training process.

- GPU Training (65s/epoch)

```
Epoch: 2/50
4219/4219 |████████████████████████████████████████| Accuracy: 88.81%| Avg Loss: 0.28 | Elapsed: 65.18s
Avg Validation Loss: 0.21 | Validation Accuracy: 92.51%
```

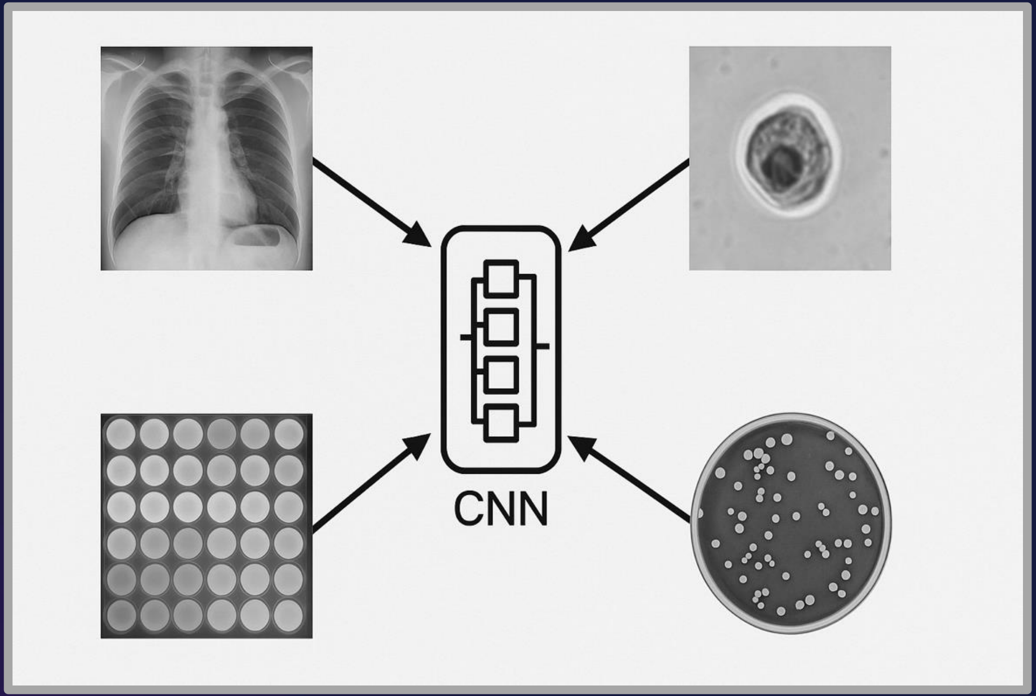
- CPU Training (10,011s/epoch)

```
Epoch: 1/50
4219/4219 |████████████████████████████████████████| Accuracy: 77.06%| Avg Loss: 0.50 | Elapsed: 10011.76s
Avg Validation Loss: 0.39 | Validation Accuracy: 83.51%
```

- GPU training is 154x faster than the CPU (~15,400% faster)

Applications of CNNs in Health and Microbiology

- Medical Imaging: detect diseases from X-rays, CT scans, MRI scans, and pathology slides.
- Diagnostics: analyze lab results, ELISA plate images, and gel electrophoresis (including Western blot).
- Microbiology: classify bacterial/viral colonies, identify healthy vs abnormal cells in microscopy.
- Surveillance: large-scale outbreak detection and pattern recognition.



Future Improvements

- Improve GPU acceleration.
- Reduce memory usage across framework.
- Expanded library of activations and losses.
- Support for advanced blocks.
- More Data scalars/transformers.
- Layer normalization.
- Python interface for easier usage.

Demo:
End-to-End Pipeline

(watch it train & run in real-time)

Thank you

<u>Code Available here</u>