HW2 Tutorial

Georgia Tech

Logistics

Logistics

Deliverables and Due Date:

HW2 Part 1: Implementing CNN from Scratch [7 points]
HW2 Part 2: PyTorch - Implement Two-Layer Network, Vanilla CNN and Your own model [5+2 points]
Report: [6 points]

Due Date: 11:59 PM Feb 19, 2024
Grace Period: 11:59 PM Feb 21, 2024

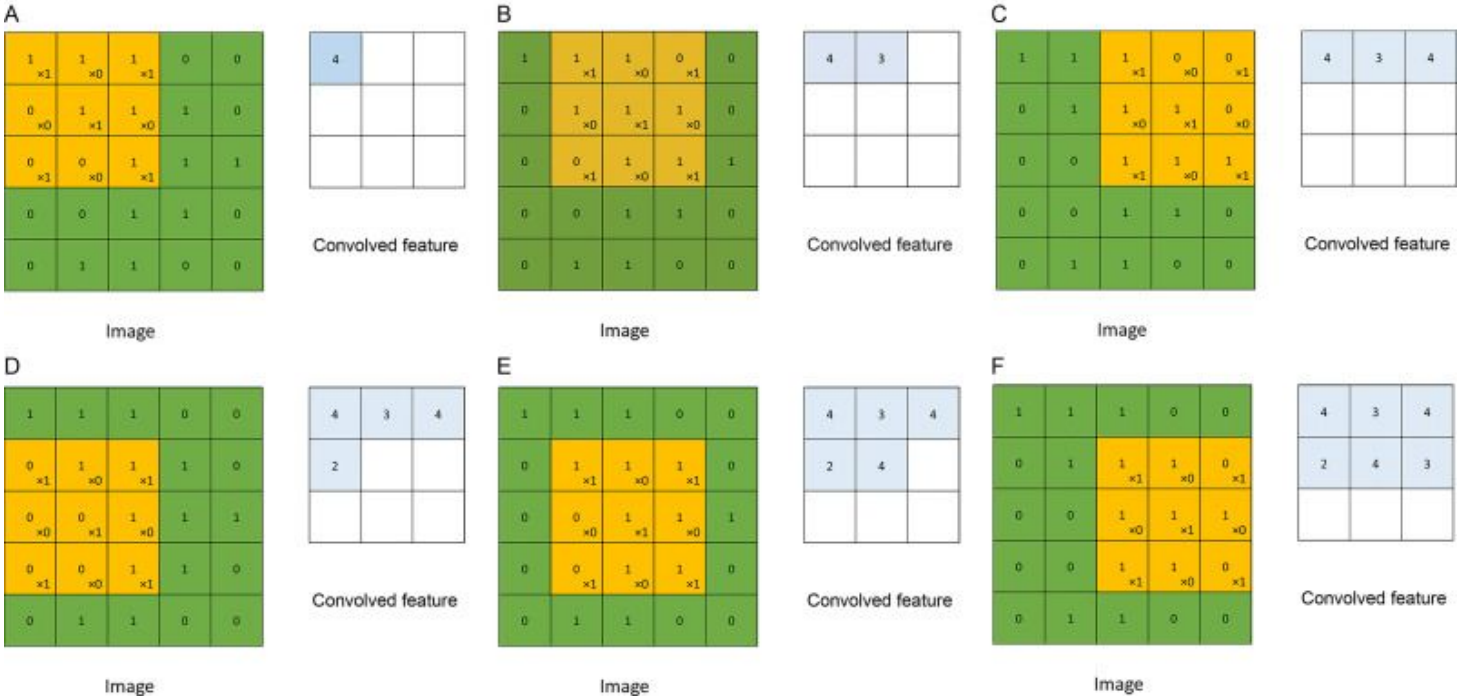
Coding

Part - 1: CNN from Scratch

- Modules
  - Convolution
  - MaxPooling
  - ReLU
  - Linear
  - Tie everything together: Convolutional Classifier
- Implement Optimizer:
  - SGD with momentum

Part - 1: CNN from Scratch

Convolution:

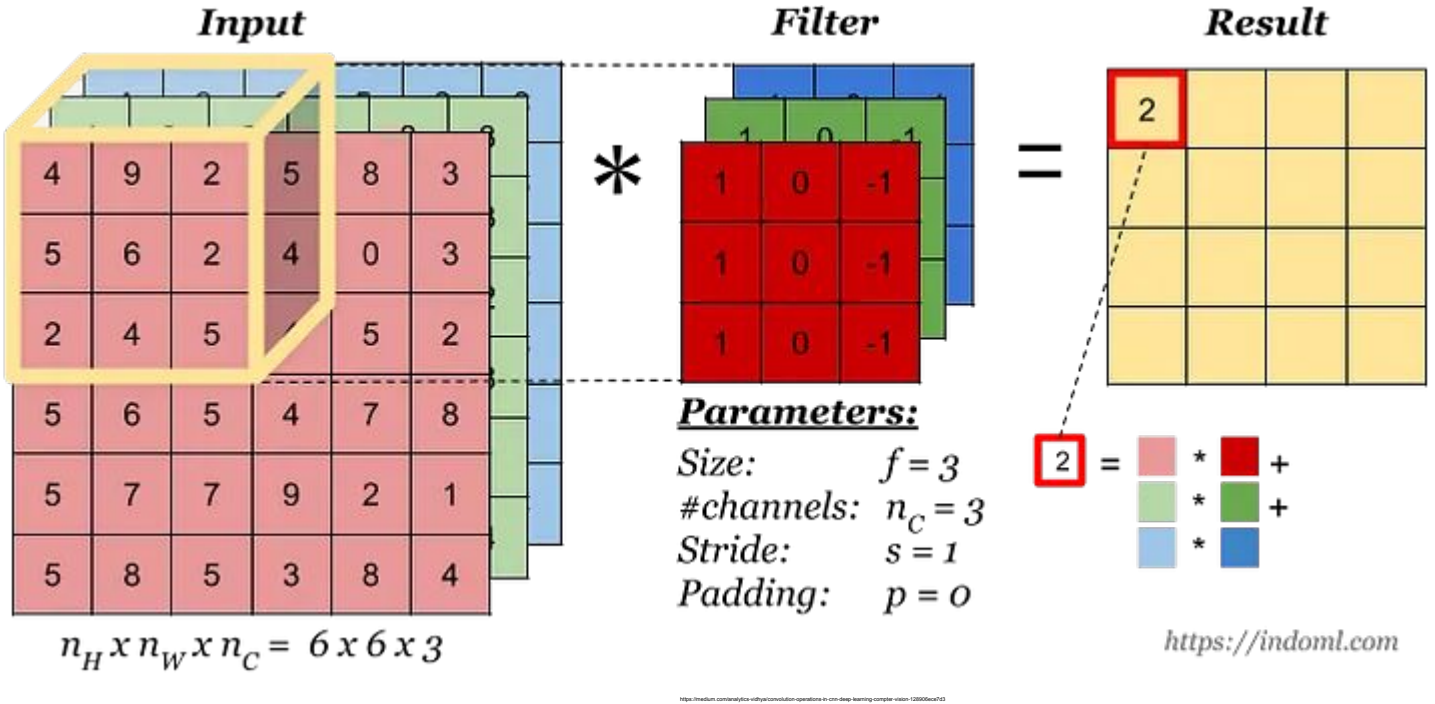


https://medium.com/analytics-vidhya/convolution-operations-in-cnn-deep-learning-computer-vision-128906eca7d3

Georgia Tech

Part - 1: CNN from Scratch

Convolution:



https://medium.com/analytics-vidhya/convolution-operations-in-cnn-deep-learning-compler-vision-126900ece7d3

Part - 1: CNN from Scratch

Convolution:

- Calculate final size of output
  - H = 1 +((input.H - kernel_size+2*padding)/stride))
  - W = 1 +((input.W- kernel_size+2*padding)/stride))

- Can use loops to iterate over batch, output_channels, row, and column
- Use np.pad, np.dot

https://medium.com/analytics-vidhya/convolution-operations-in-cnn-deep-learning-compter-vision-126906eca7d3

Georgia Tech

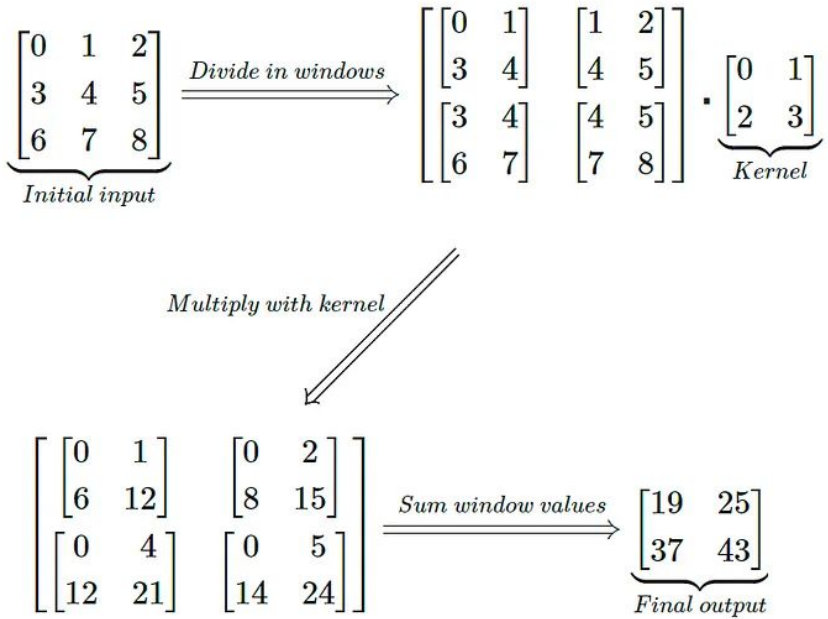
Part - 1: CNN from Scratch

Vectorization Tricks:

1. “Place” the kernel on the image
2. Multiply the kernel weights by the corresponding pixels
3. Sum everything together
4. “Stride” (or move) the kernel x amount of pixels
5. Repeat for the whole image
6. Repeat for all the kernels

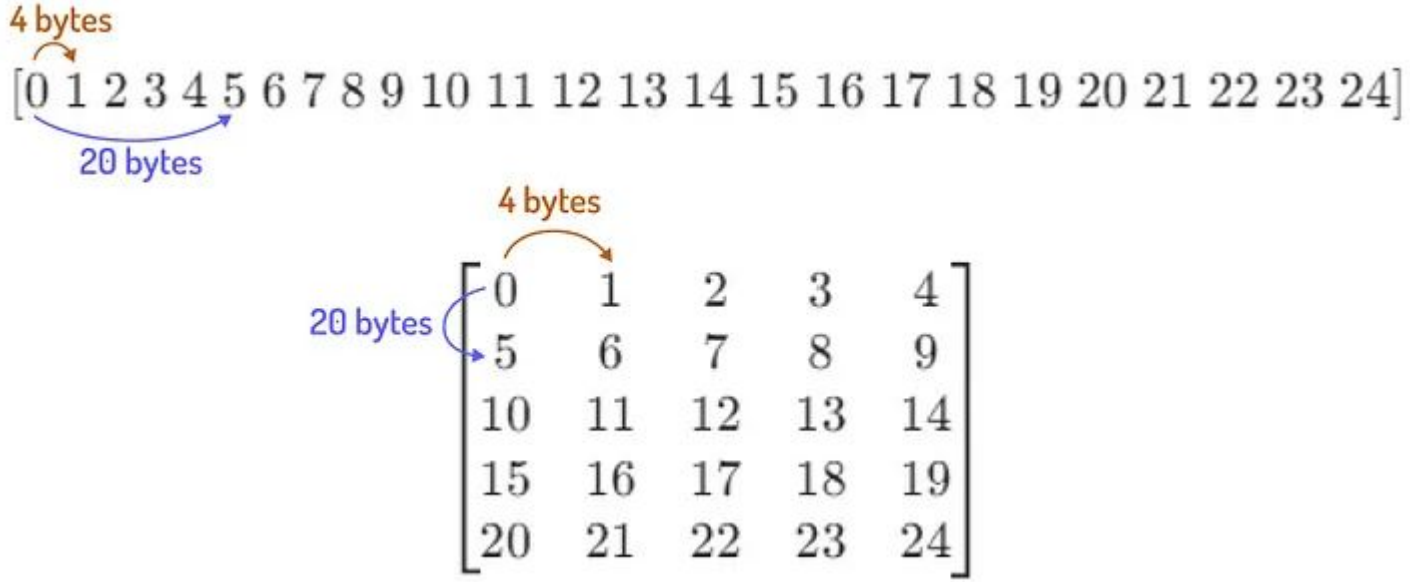
Part - 1: CNN from Scratch

Vectorization Tricks:



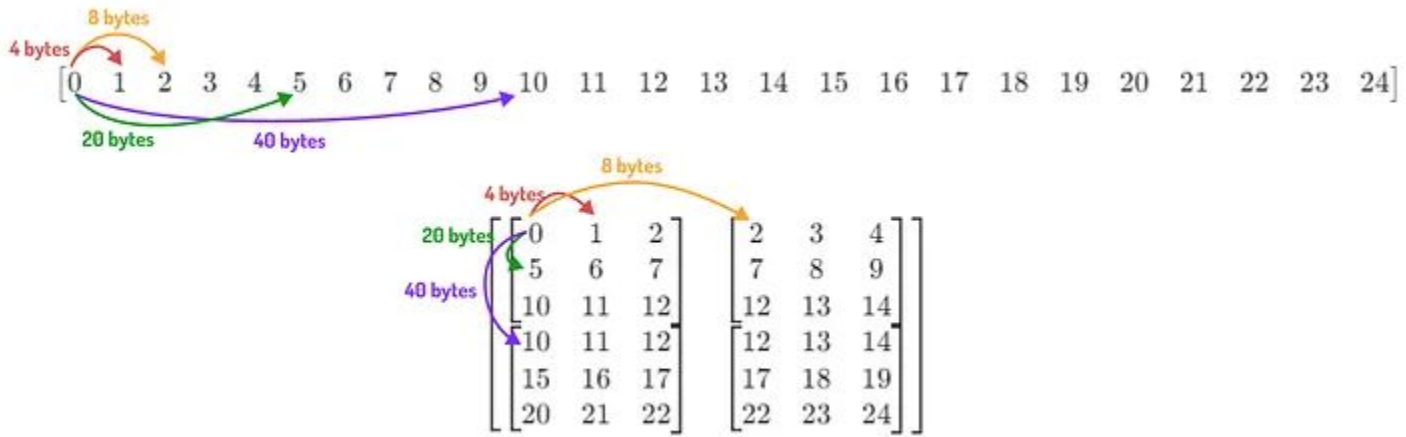
Part - 1: CNN from Scratch

Vectorization Tricks: **np.lib.stride_tricks.as_strided**



Part - 1: CNN from Scratch

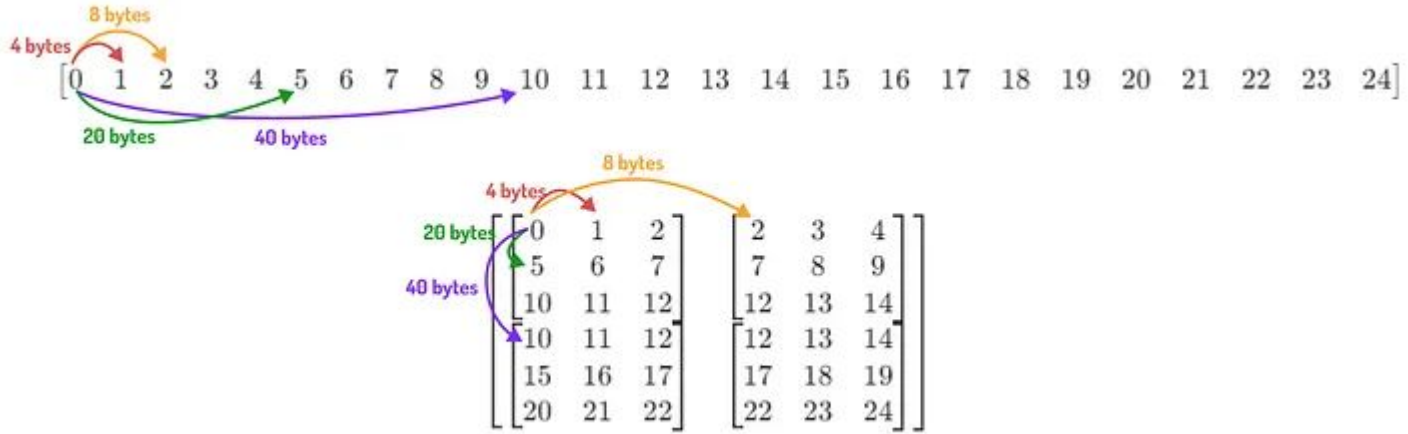
Vectorization Tricks:



Part - 1: CNN from Scratch

Vectorization Tricks:

(original_row_stride * layer_stride,
 original_column_stride * layer_stride,
 original_row_stride,
 original_column_stride)



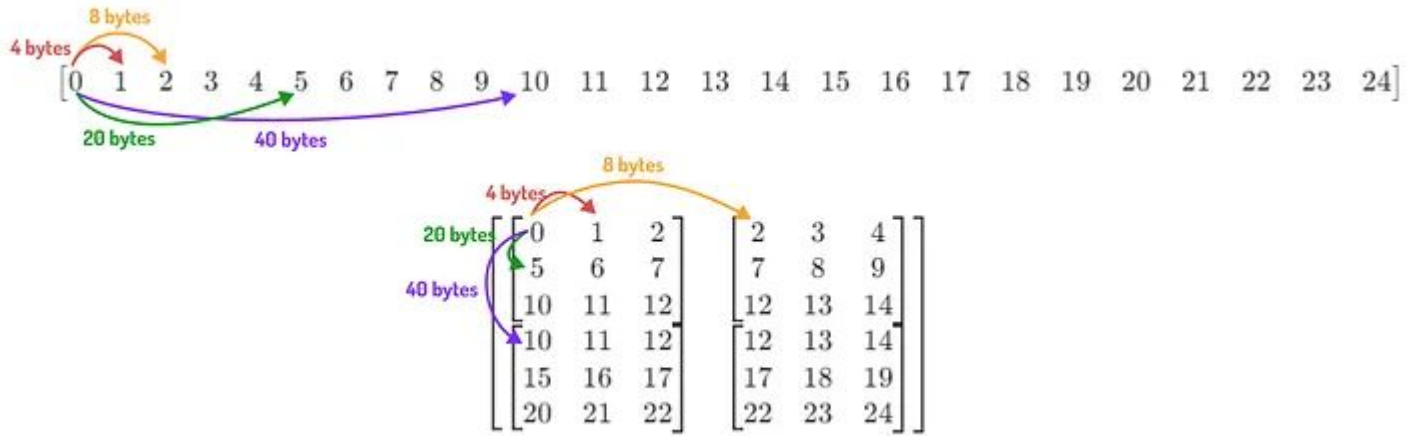
Georgia Tech

Part - 1: CNN from Scratch

Vectorization Tricks:

new_shape = (out_height, out_width, kernel_size, kernel_size)
new_strides = (rows_stride * layer_stride, columns_strides * layer_stride, rows_stride, columns_strides)

windowed_input = **np.lib.stride_tricks.as_strided**(input, new_shape, new_strides)



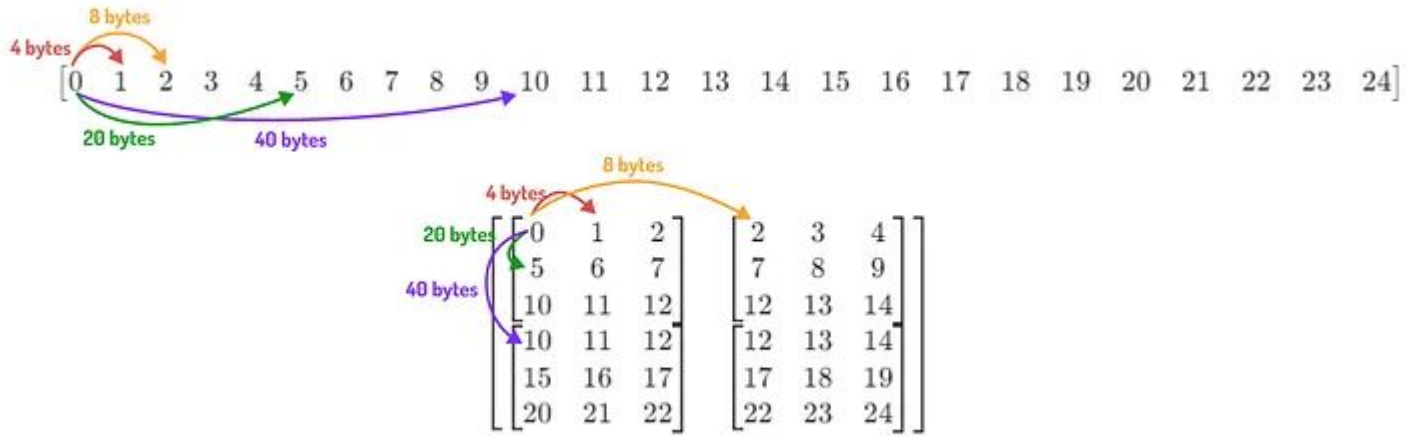
Georgia Tech

Part - 1: CNN from Scratch

Vectorization Tricks:

new_shape = (out_height, out_width, kernel_size, kernel_size)
new_strides = (rows_stride * layer_stride, columns_strides * layer_stride, rows_stride, columns_strides)

windowed_input = **np.lib.stride_tricks.as_strided**(input, new_shape, new_strides)



Georgia Tech

Part - 1: CNN from Scratch

Vectorization Tricks:

np.einsum:
windowed_input = np.random.rand(batch_size, channels, height, width, kernel_size, kernel_size)
weights = np.random.rand(num_filters, channels, kernel_size, kernel_size)

np.einsum(? , windowed_input, weights)

Broadcasting

A = (a,b,c) B=(b,)

A+B[:,np.newaxis]

A+B[:,None]

Broadcast across 1st and 3rd dimensions before adding

Unravel Index

```
arr_2d = np.array([[1, 2, 3],
                   [4, 5, 6],
                   [7, 8, 9]])

flat_index = 5
multi_index = np.unravel_index(flat_index, arr_2d.shape)
```

(1,2)

Part - 1: CNN from Scratch

- Max-Pool
- Use loops to calculate the maximum value
- np.unravel_index

https://medium.com/analytics-vidhya/convolution-operations-in-cnn-deep-learning-compiler-vision-126906eca7d3

Georgia Tech

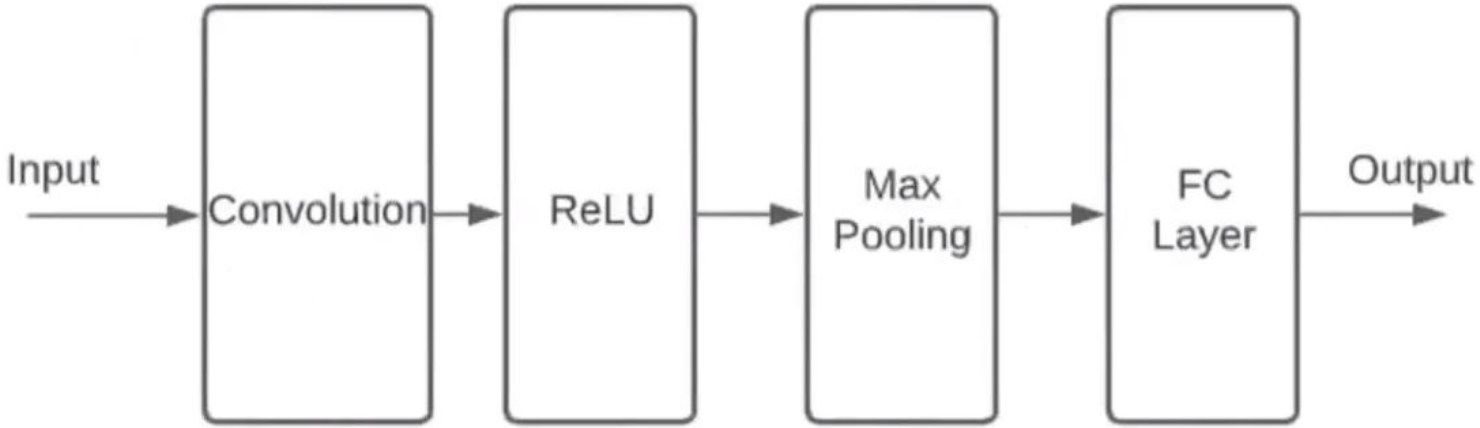
Part-2

Complete the training loop of PyTorch

- Implement the training and validation step utilizing PyTorch
- Key concept is that in validation you won't be updating the gradients

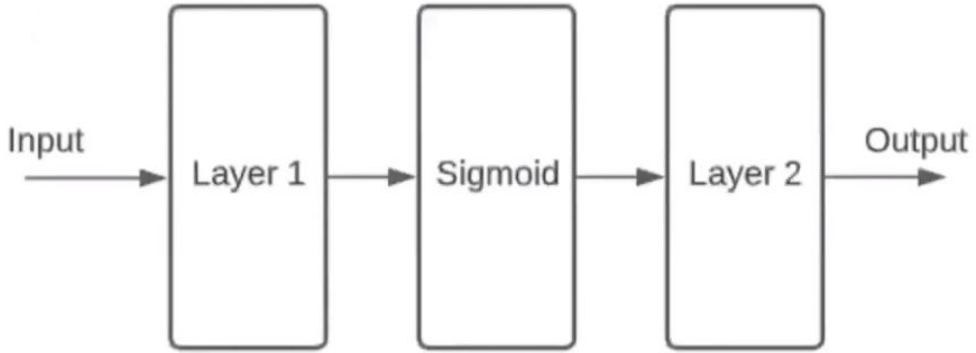
Part-2

- Implement a basic CNN utilizing PyTorch



Part-2

- Two fully connected layers with a sigmoid activation function in between



Part-2: Your Model

- Your accuracy on test data needs to be above 0.5 to receive partial credits. For full credit you need above 0.8.
- You must upload the checkpoint because gradescope won't be able to train your model
  - The script will create and save the checkpoints for you.
- If you make your model too deep then gradescope might timeout as there is a limit for runtime + size (10 min for all tests, 100 mb filesize)
- You can utilize your own model selection. Take inspiration from architecture design discussed in lecture.