Report

MCP261 Project

-Kartik Aggarwal (2018ME20690)

-Siddharth Dixit (2018ME20727)

Problem Statement:

Building of a Convolutional Neural Network (CNN) using NumPy from scratch and detection of edges in images using it.

Approach:

The major steps involved in the project are as follows:

1. Reading the input image – The image is turned into grey to convert 3-dimensional images into 2-dmensional images.

2. Preparing filters - The filters of the first conv layer are prepared according to the input image dimensions. The filter is created by specifying the following:

1) Number of filters (Taken as 2 in the project)

2) Size of first dimension of image

3) Size of second dimension of image

3. Conv layer - Convolving each filter with the input image.

4. ReLU layer - Applying ReLU activation function on the feature maps (output of conv layer).

5. Max Pooling layer - Applying the pooling operation on the output of ReLU layer.

Filter used for Vertical Line detection – **Sobel Filter** $\begin{bmatrix} \mathbf{-1} & \mathbf{0} & \mathbf{1} \\ \mathbf{-2} & \mathbf{0} & \mathbf{2} \\ \mathbf{-1} & \mathbf{0} & \mathbf{1} \end{bmatrix}$

Filter used for Horizontal Line detection – **Sobel Filter** $\begin{bmatrix} \mathbf{-1} & \mathbf{-2} & \mathbf{-1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{1} & \mathbf{2} & \mathbf{1} \end{bmatrix}$

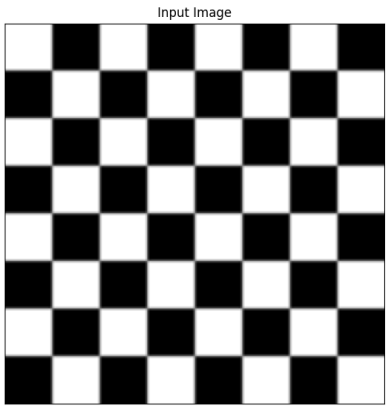
Filter used for Line detection at 135 degrees– **Sobel Filter** $\begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$

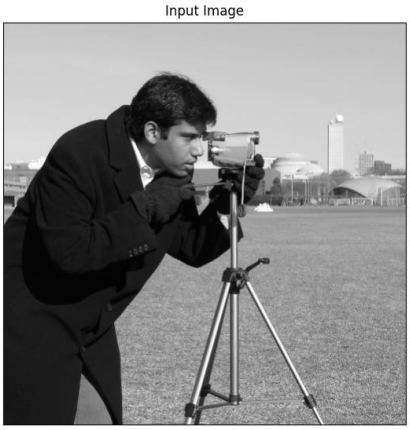
Filter used for Line detection at 45 degrees– **Sobel Filter** $\begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix}$

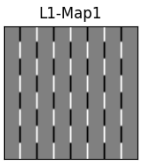
Filter used for all lines detection – $\begin{bmatrix} 0 & 1 & 0 \\ -1 & -4 & 1 \\ 0 & -1 & 0 \end{bmatrix}$

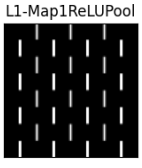
Results:

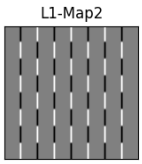
The following results were obtained by running the code:

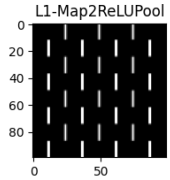


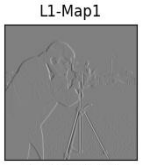


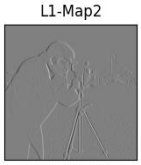


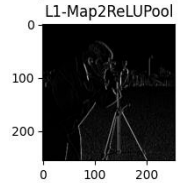




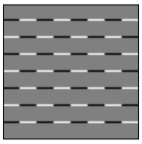




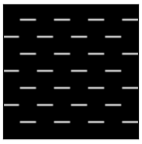




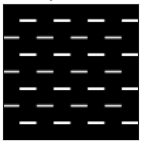
L2-Map1



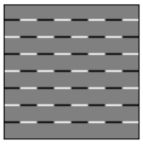
L2-Map1ReLU



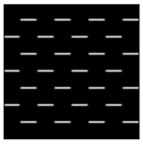
L2-Map1ReLUPool



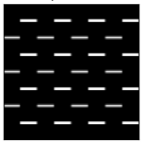
L2-Map2



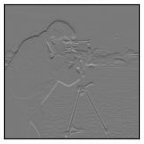
L2-Map2ReLU



L2-Map2ReLUPool



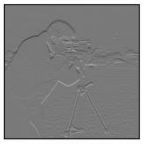
L2-Map1



L2-Map1ReLU

L2-Map1ReLUPool

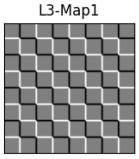
L2-Map2



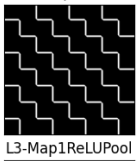
L2-Map2ReLU

L2-Map2ReLUPool

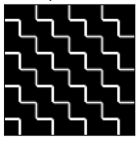
L3-Map1



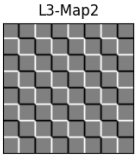
L3-Map1ReLU



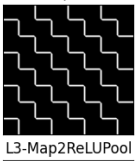
L3-Map1ReLUPool



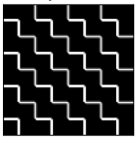
L3-Map2



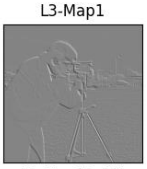
L3-Map2ReLU



L3-Map2ReLUPool



L3-Map1



L3-Map1ReLU

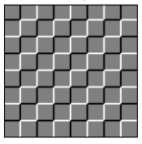
L3-Map1ReLUPool

L3-Map2

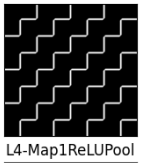
L3-Map2ReLU

L3-Map2ReLUPool

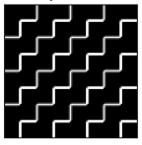
L4-Map1



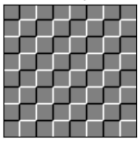
L4-Map1ReLU



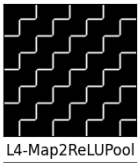
L4-Map1ReLUPool



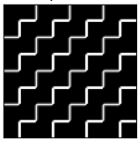
L4-Map2



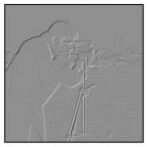
L4-Map2ReLU



L4-Map2ReLUPool



L4-Map1



L4-Map1ReLU

L4-Map1ReLUPool

L4-Map2

L4-Map2ReLU

L4-Map2ReLUPool

Final edge detected images -