Ahmet Akman 2442366
Middle East Technical University
Electrical and Electronics Engineering

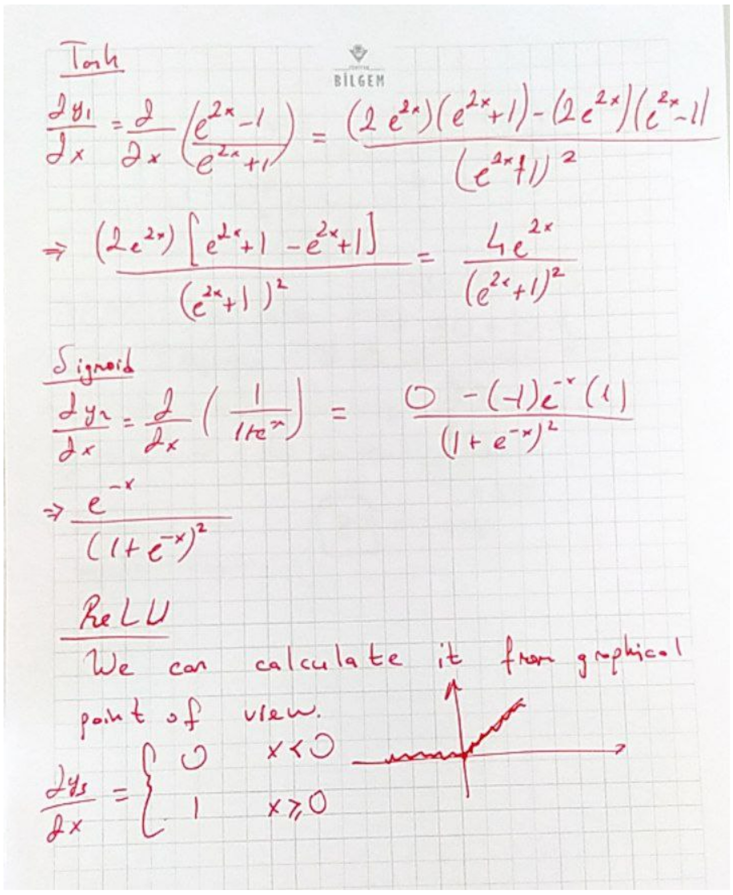
April 4, 2024

HOMEWORK 1 — Report

1 Question 1

1.1 Question 1.1 - Preliminaries

The partial derivaive calculation steps for Tanh, Sigmoid and ReLU activation functions are shown in Figure 1.



Tanh

$$\frac{\partial y_1}{\partial x} = \frac{\partial}{\partial x}\left(\frac{e^{2x}-1}{e^{2x}+1}\right) = \frac{(2e^{2x})(e^{2x}+1)-(2e^{2x})(e^{2x}-1)}{(e^{2x}+1)^2}$$

$$\Rightarrow \frac{(2e^{2x})\left[e^{2x}+1-e^{2x}+1\right]}{(e^{2x}+1)^2} = \frac{4e^{2x}}{(e^{2x}+1)^2}$$

Sigmoid

$$\frac{\partial y_2}{\partial x} = \frac{\partial}{\partial x}\left(\frac{1}{1+e^{-x}}\right) = \frac{0-(-1)e^{-x}(1)}{(1+e^{-x})^2}$$

$$\Rightarrow \frac{e^{-x}}{(1+e^{-x})^2}$$

ReLU

We can calculate it from graphical point of view.

$$\frac{\partial y_3}{\partial x} = \begin{cases} 0 & x<0 \\ 1 & x\geq 0 \end{cases}$$

Figure 1: Partial derivative calculation steps for Tanh, Sigmoid and ReLU activation functions.

Figure 2 illustrates the activation functions' response between -2 and 2. The plots are obtained using matplotlib library as instucted.

Figure 3 indicates the gradients of those functions in the same range. The gradients are calculated using the partial derivatives derived in Figure 1.

1.2 Question 1.2

In this part, utilizing the given template of code, MLP with one hidden layer is implemented. The input-output pairs are fetched from the XOR data provided in utils.py. Note that for all networks learning rate

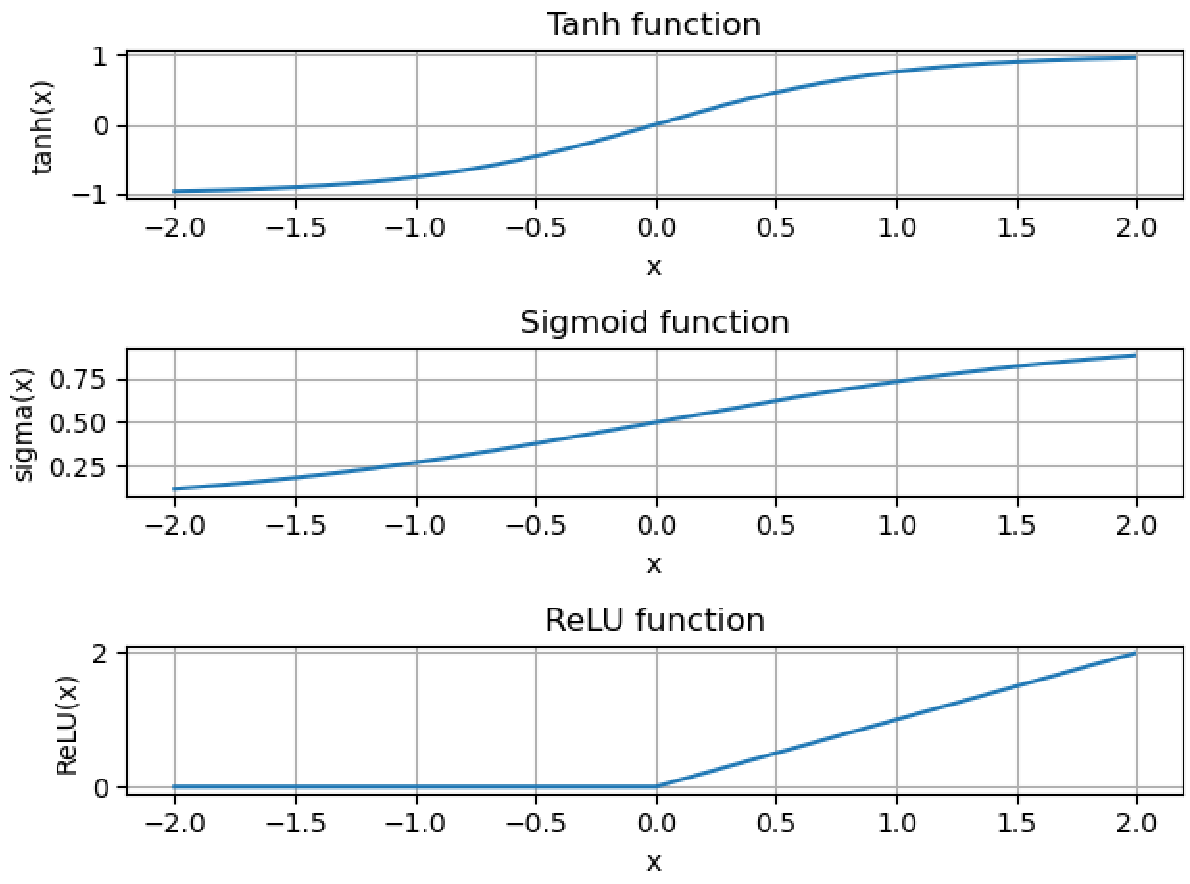


Figure 2: Activation functions plot.

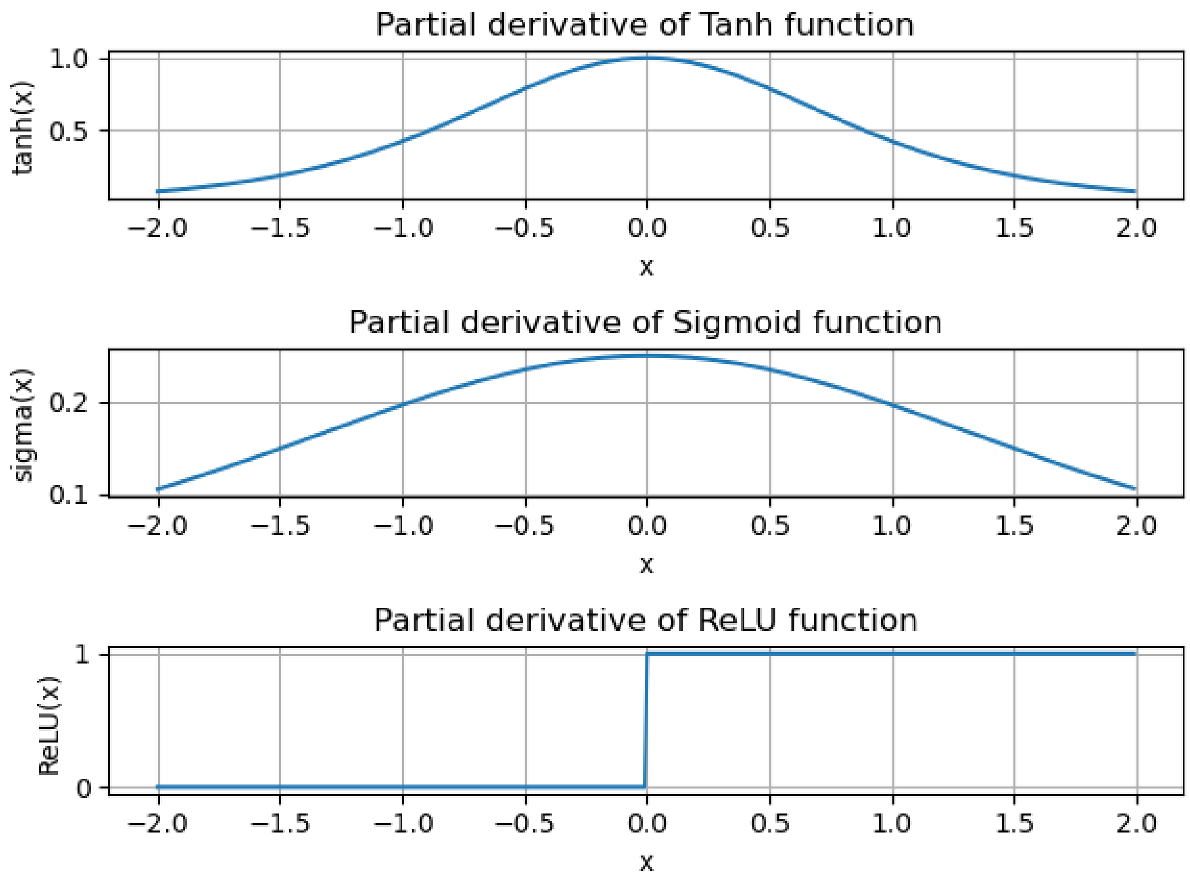


Figure 3: Gradients of the activation functions plot.

is fixed to 0.00001 and seed is utilized. Figure 4 shows the decision boundary for the sigmoid activated network. Similarly, Figure 5 is the decision boundary for the tanh activated network and Figure 6 is the

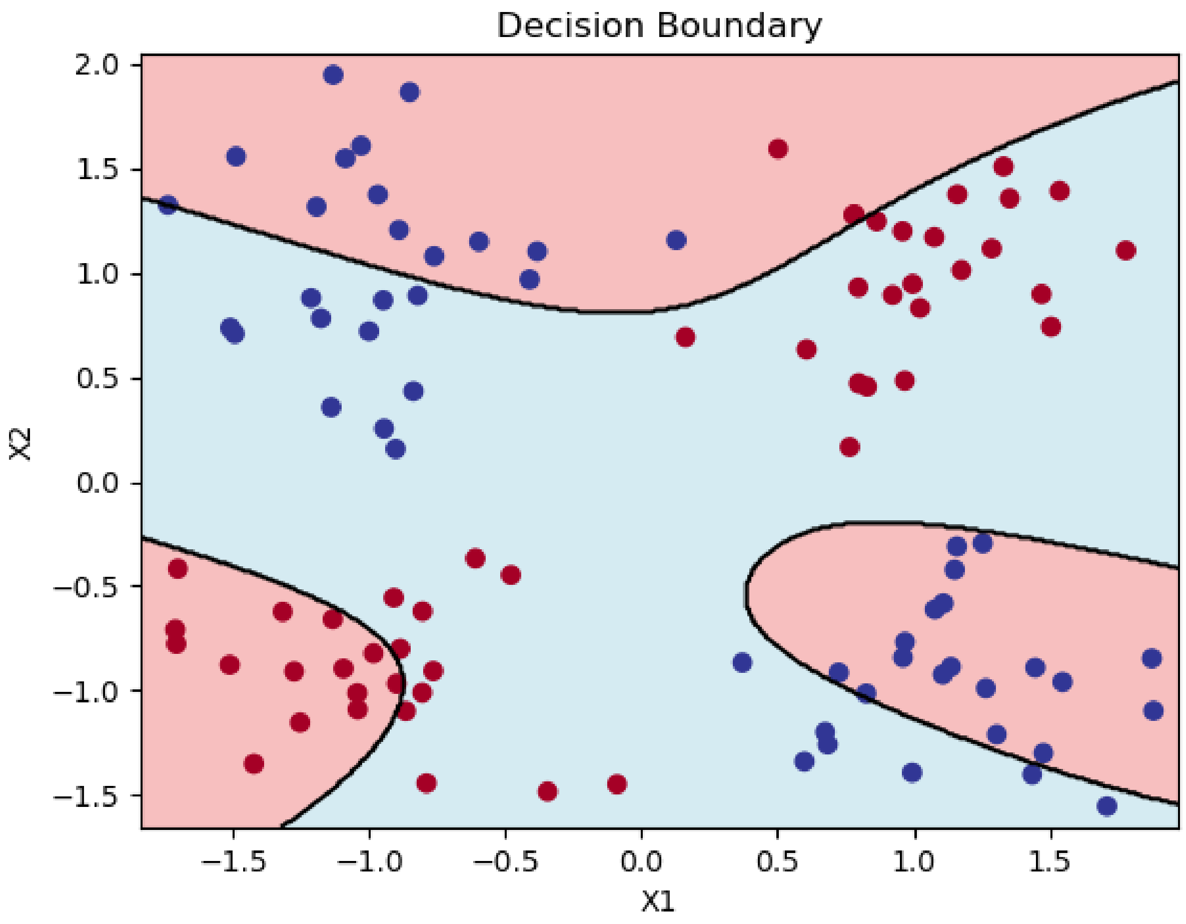


Figure 4: Sigmoid activated XOR problem output.

decision boundary for the ReLU activated network.

1.3 Question 1.3

1. All of those activation functions provide a smooth transition from one state to another. The advantage of Tanh and Sigmoid is, they are limited in the range of -1 to 1 and 0 to 1, respectively. This property can be beneficial in some cases. However, the ReLU function is not limited in the range. It is also computationally cheaper than the other two. The disadvantage of ReLU is that it is not smooth at the origin. This can cause some problems in the optimization process. Another advantage of ReLU is negative gradients are zero. This may be helpful in some cases. **2.**

XOR problem is two input decision problem where inputs have to be different than each other to obtain 1. Since the output is always with respect to the state of two variables, decision boundary is not linear. Therefore, a single layer perceptron can not solve since in one step two case evaluations can not be done. Yet, by adding a hidden layer, the problem is solved. The activation functions are used to introduce non-linearity to the network. The decision boundary of the XOR problem is shown in Figures 4, 5 and 6. As can be seen from the figures, the MLP's with activation functions can solve the XOR problem at some extent.

3. The boundaries change in each run, since the initial weights and the data points are randomly generated. Therefore, the decision boundaries are dependent on the training process where initial randomness

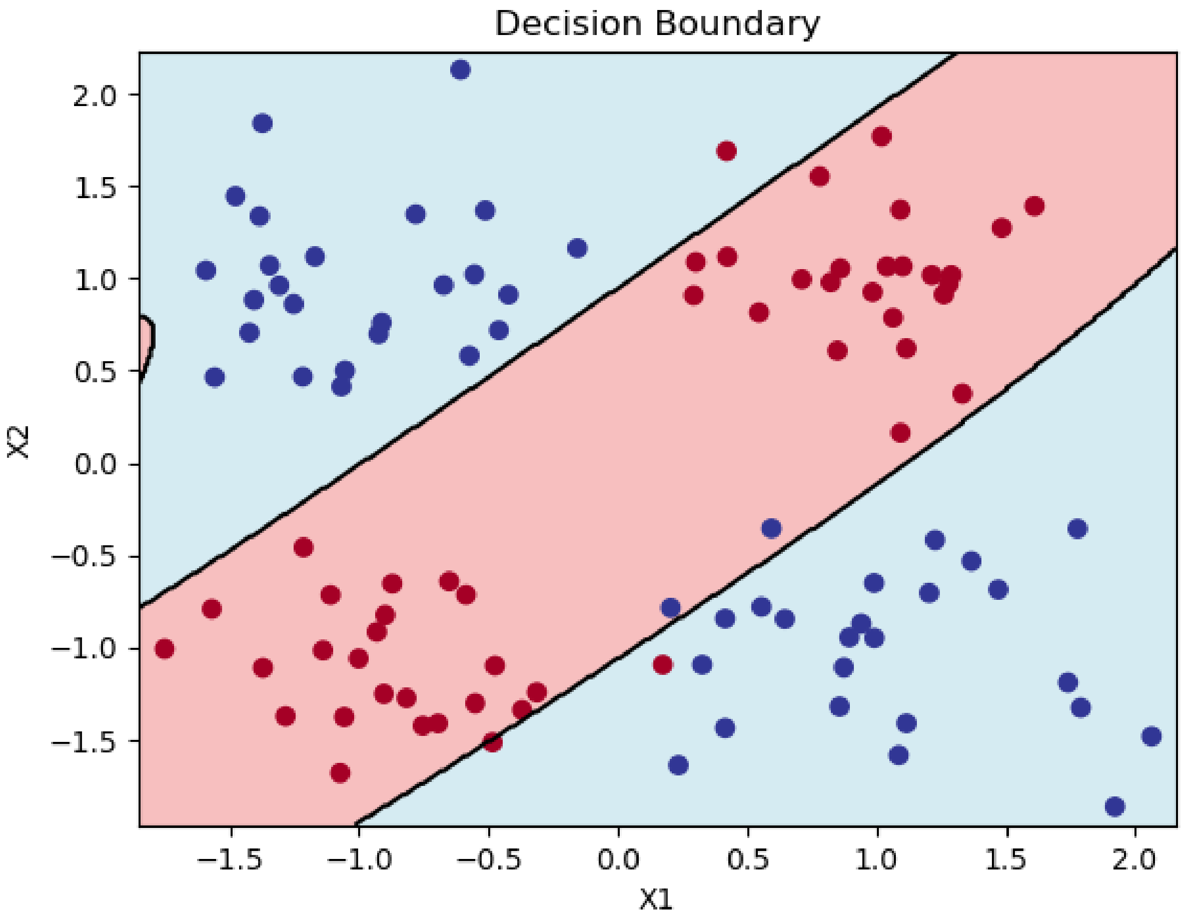


Figure 5: Tanh activated XOR problem output.

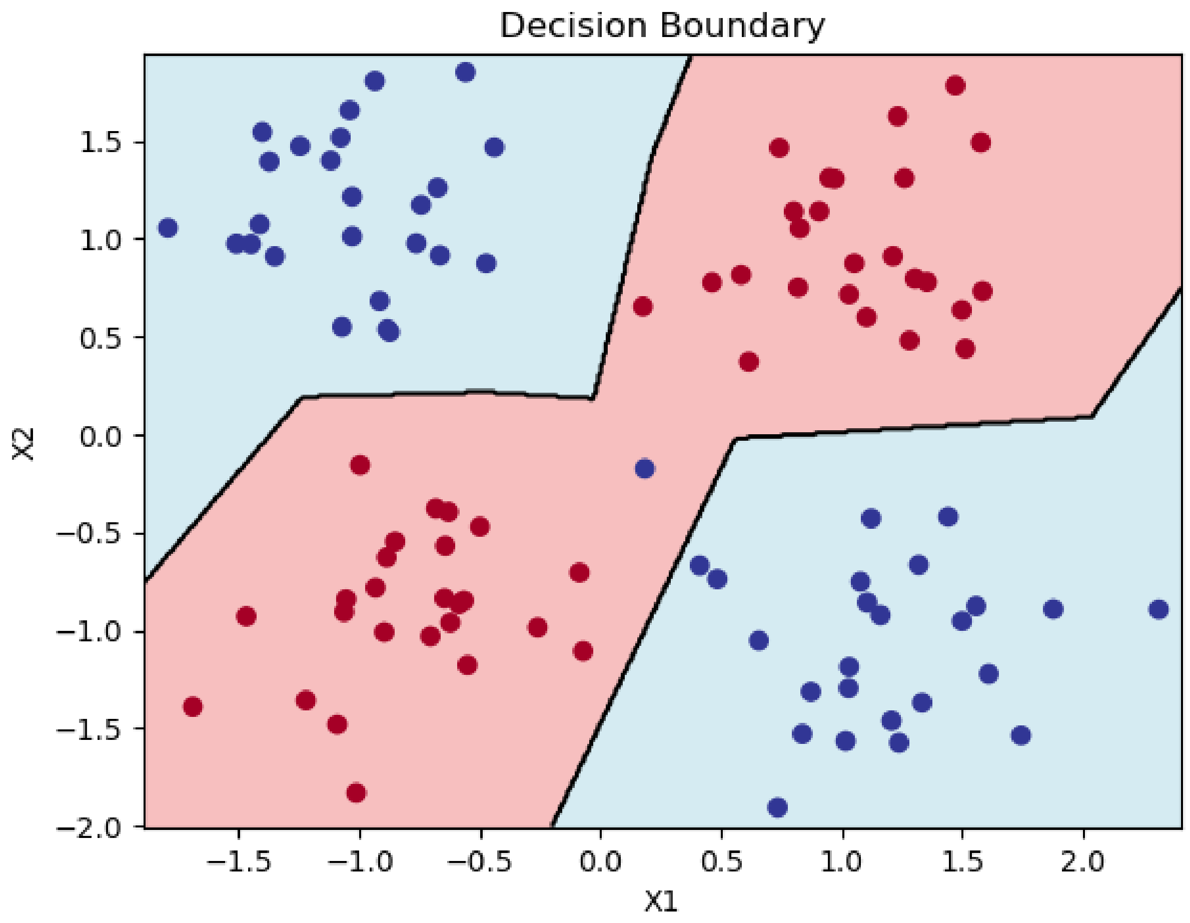


Figure 6: ReLU activated XOR problem output.

lead to different outputs.

2 Question 2

A convolution operator is implemented using a set of nested loop. The outputs are checked on a seperate code so that they are identical to the output of torch.conv2d. Using the provided inputs, outputs provided in Figure 7 are obtained.

2.1 Question 2.2 - Discussions

1. Two dimensional convolution operation provides filtering in two dimension via a kernel. The kernel is applied to the input image in a sliding window fashion. The output is obtained by element-wise multiplication of the kernel and the input image. The kernel is then shifted by a stride and the process is repeated. That is, two dimensional convolution operation with learnable kernal entries are commonly used in image processing to extract features from the image. The kernel is learned during the training process. Those feature maps encode necessary information about distinct clues in the image. For example in the case of object recognition, a specific kernel can be trained to detect bicyle rims. Kernel of a convolution layer corresponds to the filter function in one dimension. It is used to suppress or enhance certain information in the input image.

2.The sizes of the kernel corresponds to the size of the filter. Therefore, the size list can be explained as follows (batch size, input channels, output channels, filter height, filter width) . Batch size is the number of input sets in a batch if batching is used, here we have not utilized it. Input channels are the number of depth dimension in the image, for example in RGB frames it is three. This might be different for different sensor inputs and representations. The output channel number determines the depth dimension of the output tensor. The filter height and width are the dimensions of the filter.

3. To understand what exactly happened here, let us plot the kernel and input for the number zero. Figure 8 illustrates the plot.

When we have a look at what output image represents, on each row an input image is convolved with a kernel. There are different kernels applied to the input image. The output image is the result of those convolutions. Figure 8 shows the input and kernel pairs in order.

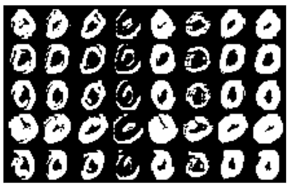
4. Each convolution kernel embeds a certain feature of the image. That is, if we have a look at Figure 8.a we can see that kernel 4 enhances the countours on the image whereas kernel 6 enhances the filled part of those contours. This is the reason why we see similarly formatted outputs on the same column. **5.** Similarly, since we "highlight" different properties on the image with out filters, we see different patterns on the output side even if the input is the same.

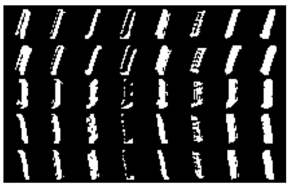
6. So, we can interpret form 4, and 5 that the output of the convolution layer is the result of the feature extraction process. The output is the result of the convolution of the input image with the learned filters. The output is the feature map that encodes the information about the input image. By post-processing those feature maps, we can obtain the necessary interpreations about the input image.

3 Question 3

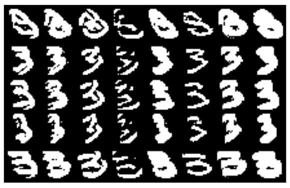
3.1 Question 3.1

The implementations related to each architectures are completed as instructed. The code is given in appendix. As the result the plot shown in Figure


(a) Output 0


(b) Output 1

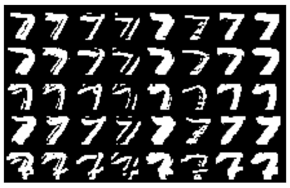
(c) Output 2

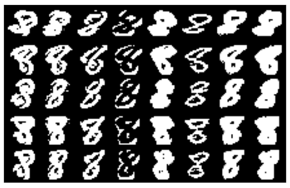

(d) Output 3

(e) Output 4

(f) Output 5

(g) Output 6


(h) Output 7


(i) Output 8

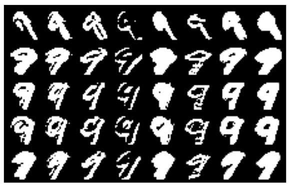
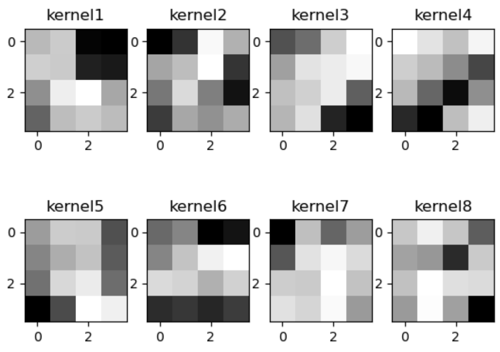
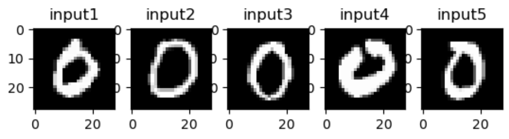

(j) Output 9

Figure 7: Convolution output



(a) Kernels



(b) Input set for number zero.

Figure 8: Kernel and input for number zero.

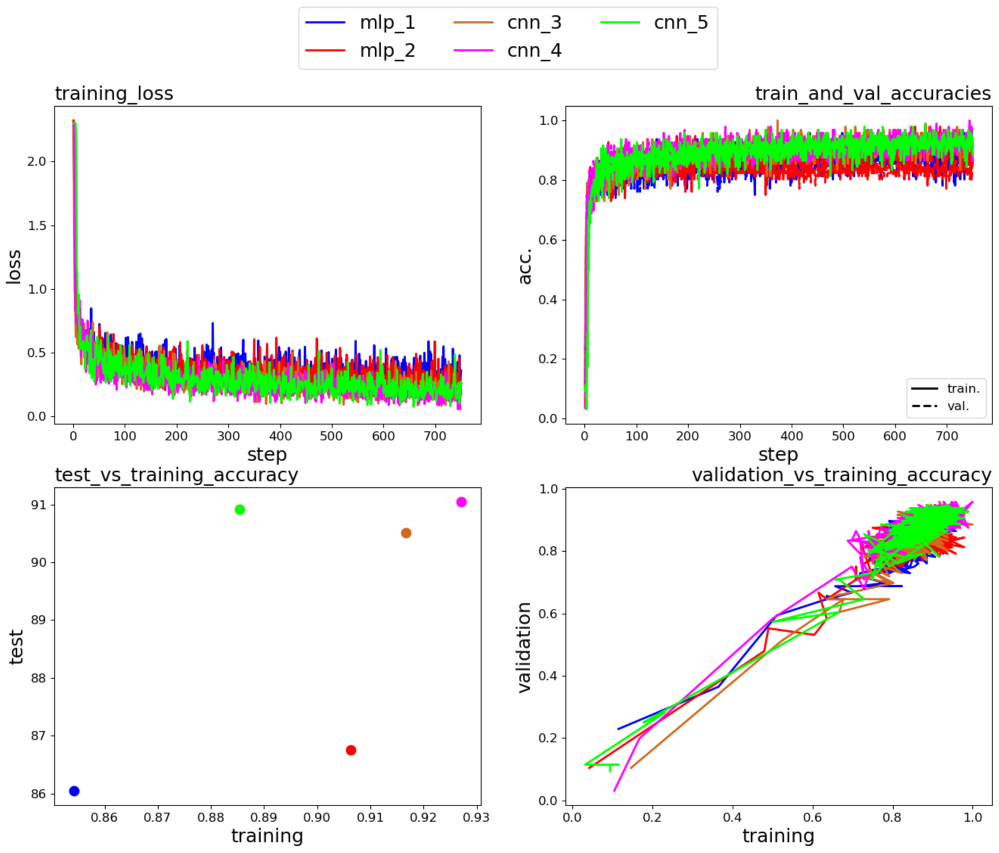
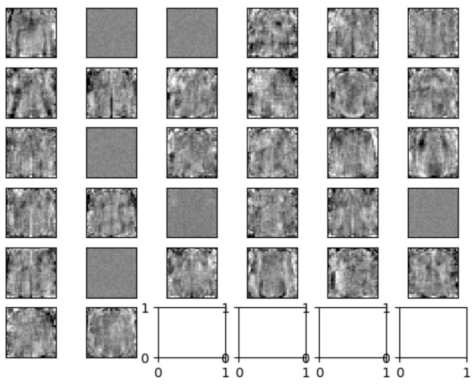
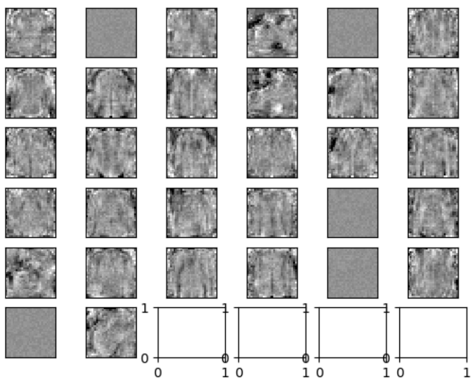
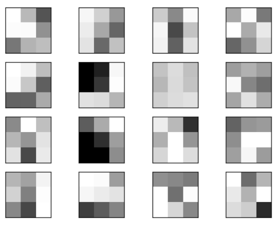
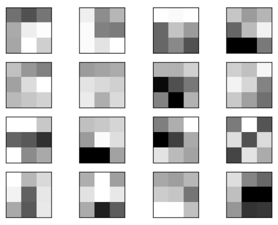


Figure 9: Benchmark of five different architectures.


(a) mlp 1


(b) mlp 2


(c) cnn 3


(d) cnn 4

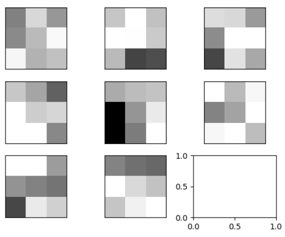

(e) cnn 5

Figure 10: Weights of the first layers.

3.2 Question 3.2

1. 2. 3. 4. 5. 6. 7. 8. 9. 10.

4 Question 4

5 Question 5

6 References

References

[1] M. Stimberg, R. Brette, and D. F. Goodman, “Brian 2, an intuitive and efficient neural simulator,” *eLife*, vol. 8, p. e47314, Aug. 2019.

[2] R. Brette, M. Rudolph-Lilith, N. T. Carnevale, M. L. Hines, D. Beeman, J. M. Bower, M. Diesmann, A. Morrison, P. H. Goodman, F. C. Harris, M. Zirpe, T. Natschläger, D. Pecevski, B. Ermentrout, M. Djurfeldt, A. Lansner, O. Rochel, T. Viéville, E. B. Müller, A. P. Davison, S. E. Boustani, and A. Destexhe, “Simulation of networks of spiking neurons: A review of tools and strategies,” *Journal of Computational Neuroscience*, vol. 23, pp. 349–398, 2006.

Submitted by Ahmet Akman 2442366 on April 4, 2024.