Theoretical Questions

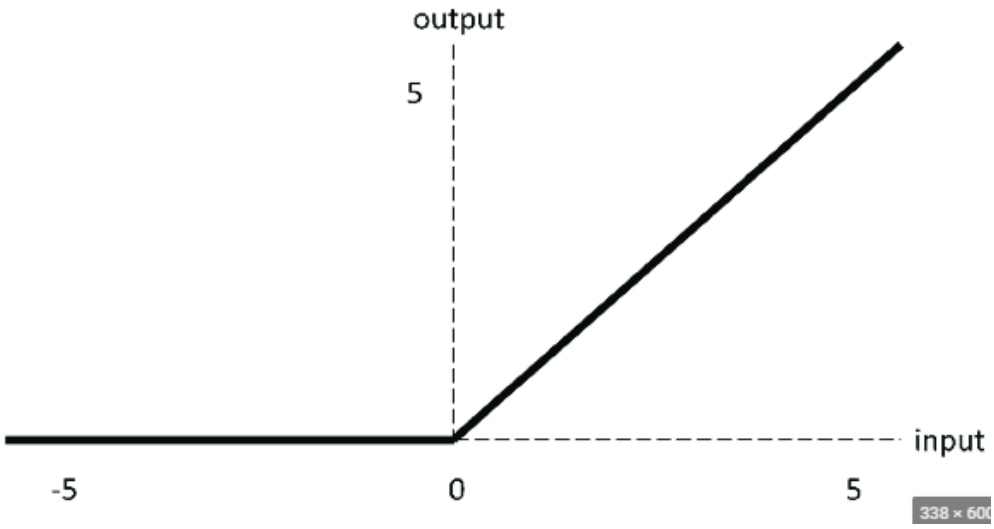
Part 1

- Task 2: Explain how it can affect the model.

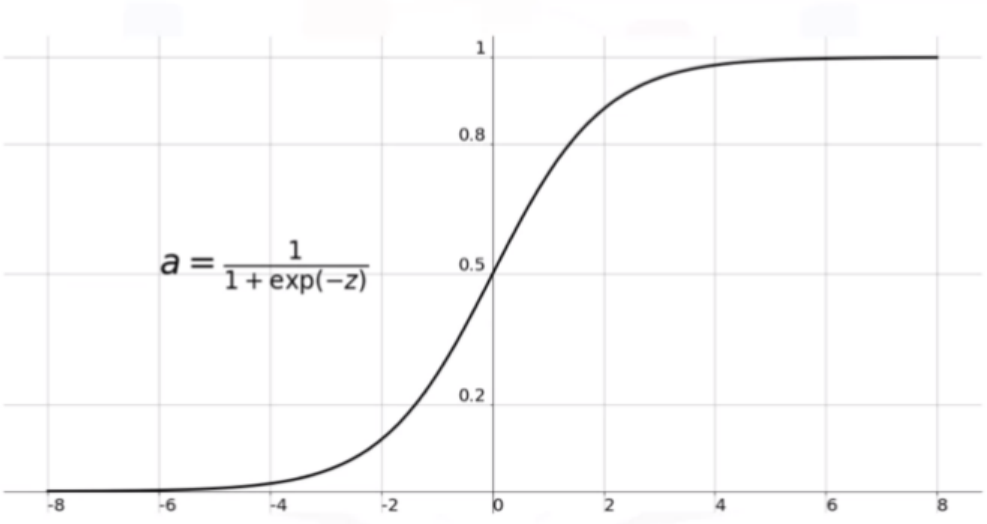
Changing the *activation function* from "relu" to "sigmoid" can affect the model since it has an influence on the final classification of each sample. The activation function's role, in each layer, is to determine that layer's output, so different activation functions will result in different outputs for the same given layer, thus perhaps changing each sample's final classification, and influencing the model's total performance.

Specifically:

Relu function:



Sigmoid function:



- Task 3: What difference does it makes in term of performance?

When increasing the number of epochs, it improves the model's performance (on the expense of longer time and higher computational power required for the model). It can be seen from the results of this section; for 25 epochs, we received a loss value of 1.056 and an accuracy value of 0.56, while using 40 epochs, on the other hand, yielded a loss value of 0.897 and an accuracy value of 0.64. Hence, both performance measures have improved with the larger number of epochs.

- Task 4: What are the advantages of the mini-batch vs. SGD?

The advantages of mini batch vs. SGD are that it is faster and contains less noise. In addition, its convergence is more accurate.

- Task 5: How does it impact your results?

The batch normalization affects the results so the new loss value received is 0.863 (vs. the previous value of 0.801) and a new accuracy value of 0.657 (vs. the previous value of 0.68). It seems that the batch normalization did not improve the model's performance in this case. The given data was not very good, so the batch normalization didn't improve the model's performance.

Part 2

- Task 1: 2D CNN

❖ How many layers does it have?

The given model has 8 layers:

5 convolutional layers (each layer includes dropout, if required, batch normalization and max pooling), 2 hidden layers (the first one includes dropout if required) and one final (classification) layer.

❖ How many filters in each layer?

"Conv2D_1" → 64 filters

"Conv2D_2", "Conv2D_3" → 128 filters

"Conv2D_4", "Conv2D_5" → 256 filters

(These are the relevant layers which include filters)

❖ Would the number of parameters be similar to a fully connected NN?

The number of parameters will not be similar to a fully connected NN since there are much more parameters to estimate in this model, as a consequence of the addition of the convolutional layers. Hence, the number of parameters in this model will be much larger.

❖ Is this specific NN performing regularization?

This specific NN is performing regularization, as can be seen in the convolutional layers when looking at the "kernel_regularizer" input argument. This means that the NN performs regularization of type "L2".

- Task 2: Number of filters

The results of the model with the original number of filters ([64, 128, 128, 256, 256]) were: Accuracy=0.25, Loss=8.31. The results of the model with the updated number of filters ([32, 64, 64, 128, 128]) were: Accuracy=0.4, Loss=4.46. As can be seen, the results have improved with the reduced number of filters.