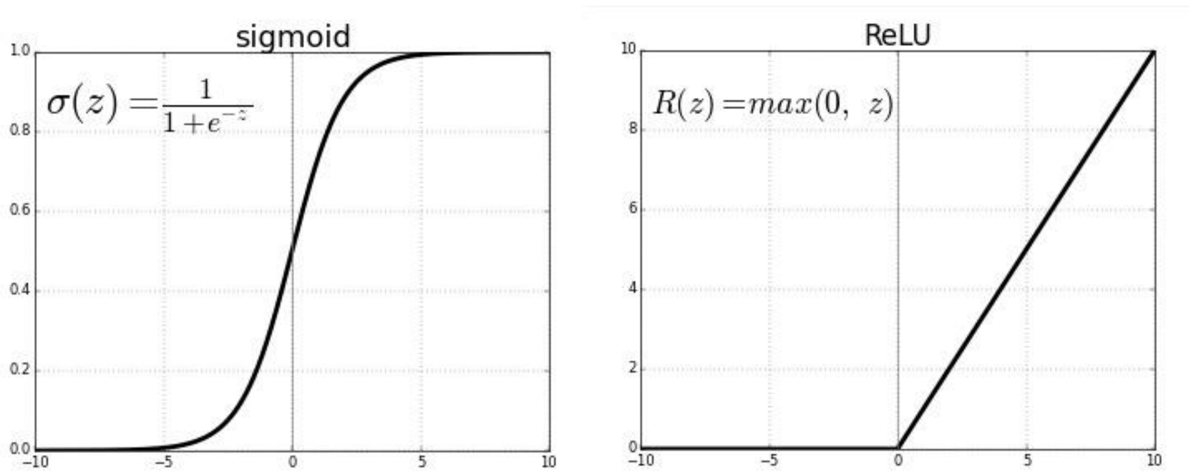
Machine Learning in Healthcare – HW4

Tania Assaf 208655951

Wajdi Nicola 315479352

<u>Part 1:</u>

- <u>Task 2:</u>
  Since the activation function plays a main rule in determining the output, and since sigmoid activation has a different behavior than the rectified linear unit (ReLU) activation function, we anticipate different outputs in each layer's output of the model. This might lead to different outcomes in the model overall and might affect the NN learning process. For example, the sigmoid activation will give outputs between 0 and 1 with smooth gradient but will have a problem reaching good prediction (the learning will be slow) when there is very high or very low values (vanishing gradient situation). In the other hand, the ReLU activation function behaves differently with a quick converge and will give zero as output for negative inputs. Figures below shows the difference between the two activation functions.



- <u>Task 3:</u>
  <u>In 25 epochs</u>: Accuracy over the testing set is 0.5314286, Loss over the testing set is 1.0985826689856393
  <u>In 40 epochs</u>: Accuracy over the testing set is 0.57714283, Loss over the testing set is 0.9959137323924473

  As we can see from the results, the accuracy became higher when we increase the epochs number, and the loss becomes smaller. Although we did not saw a great difference, there seems to be change in the model performance. We may explain that improvement in the performance when increasing the epochs since more epochs means more repeats and better converge but more complicated processes with grater computational powers are needed in the other hand (we are more precise, but the learning may take more time and resources).

- Task 4:
  Mini batches allow more precise and accurate convergence compared to big batches. While big batches may be quicker and smoother, it might not be accurate in arriving to the minimal loss.

- Task 5:
  For unnormalized batches: Accuracy over the testing set is 0.6, Loss over the testing set is 1.08478463104793
  For normalized batches: Accuracy over the testing set is 0.62857145, Loss over the testing set is 0.8818267829077584
  The normalization has improved the performance. The accuracy improved a little and the loss became smaller. This means that adding normalization before every batch might contribute to the model and improve its performance.

Part 2:

- Task 1:
  - How many layers does it have?

    The model has 8 layers as following:

    - 5 convolution layers with dropout normalization and max pooling.
    - 2 hidden layers.
    - 1 classification layer.

  - How many filter in each layer?

    For the convolution layers (according to their names in the code):

    Conv2D_1 ➔ 64

    Conv2D_2 ➔ 128

    Conv2D_3 ➔ 128

    Conv2D_4 ➔ 256

    Conv2D_5 ➔ 256

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

  We would expect the number of parameters in this model to be larger than fully connected NN. The addition of the convolution layers will increase the number of parameters that we need to estimate. Thus, we do not anticipate to have a similar number of parameters in the two cases.

- Is this specific NN performing regularization?

  As we can see from the code, the kernel regularizer parameter was determined to be l2. We expect this NN to perform regularization since we sat the regularizer accordingly.

- <u>Task 2:</u>

  For the filter input [64, 128, 128, 256, 256] → test loss, test acc: [8.112859486, 0.26857144]
  For the filter input [32, 64, 64, 128, 128] → test loss, test acc: [5.13428671700, 0.24571429]

  Reducing the number of filters leaded to smaller loss and smaller accuracy as well (the accuracy was close in the two cases). It is worth to mention that we got different behavior and results with different runs and the thing seems to be not stable. This might be because the data is not good enough (the data have low quality) to indicate in a good way the effect of the filter's number in the performance. There might be need for better data to get better explanation about the relationship between the performance and the number of filters.