**HW 4 – Theory Questions**

**Part 1**

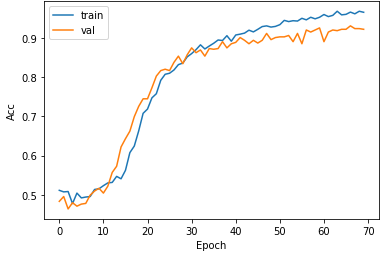
Task 2[1]:

The model can affect the efficiency of computer computation as well as the inference results. For example: RELU is very efficient and fast in calculations, but when the input values ​​are approximately zero or negative, it is impossible to calculate backpropagation, which means there is no training. At the same time, the leaky RELU allows training through backpropagation, but the prediction with negative input data is imprecise.

Other more accurate models of activation functions are sigmoid and hyperbolic tangent. They give smooth gradients and accurate predictions (the hyperbolic tangent is also centered around zero, making it easier to compute for inputs that are very negative, very positive, or neutral in value). However, these features also have their drawbacks. Such models are computationally heavy, and at their boundaries (very large or very small values ​​on the x-axis) lead to gradient vanishing, because at these ends the gradient change is almost zero. This can lead to inaccurate and slow predictions.

In our case, we expect to get slightly more accurate results for the sigmoid model, because our date is normalized from 0 to 1 and the hyperbolic tangent cannot lead us to great results.

Task 3[2]:



Figure

The number of epochs affects the performance. When there is a small number of epochs, the influence is significant and you can see a strong improvement in the performance (see Figure 1), but at some point, the system comes to a state of equilibrium and an increase in the number of epochs has almost no effect on the results. In our case, as we can see, there was no significant improvement. **Overfitting, maybe architecture not good**

Task 4[3] Mini-Batches:

To begin with, let's define the disadvantages of the SGD. Steps to a minimum are very noisy and not directional. And because of the noise, it can take longer to reach the minimum of loss function. Stochastic gradient descent is harder to compute.

At the same time, a mini-batch, it is a mixture of SGD and a Batch gradient descent. It is more directional (its vectorization is better) so it is computationally simpler, and computations require less memory and less noisy. If this algorithm gets stuck in a local minimum, then some noisy steps can cause it to break out of the local minimum.

Task 4 – Batch normalization [4]:

Batch normalization is normalizing the mean and variance of each feature at every level of representation during training. It enables to accelerate the learning process due to using higher learning rates and helps with training of very deep networks.

We expected to get better results, but the results we get are worser sometimes than without normalization.

**Part 2**

Task 1: 2D CNN

*How many layers does it have?* It has 5 layers + 3 Dence.

*How many filters in each layer?* [64, 128, 128, 256, 256]

*Would the number of parameters be similar to a fully connected NN?* The number of parameters will be bigger in fully connected NN because in fully connected NN every neuron in one layer connected to every neuron in another layer, while in CNN groups of neurons (depends on filter) have the same weight so we need to learn less. [5]

*Is this specific NN performing regularization?* There is dropout in the code. The function randomly drops out nodes during training and so provide regularization to reduce overfitting [6]. L2 – tries to

Task 2:

References:

[1] <https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/>

[2] <https://www.researchgate.net/figure/Accuracy-versus-different-epoch-plot_fig3_327261283>

[3] <https://medium.com/@divakar_239/stochastic-vs-batch-gradient-descent-8820568eada1>

[4] Lecture 16, slide 35.

[5] <https://medium.com/swlh/fully-connected-vs-convolutional-neural-networks-813ca7bc6ee5>

[6] <https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/>