Machine Learning in Healthcare - 336546

Homework 4

Dmitry Rudman - 333814283

Arseny Belousov - 961152006

Part 1

Task 1

Fully connected NN with ReLU activation results:

```
[test loss, test accuracy]: [0.3796639549732208, 0.8314286]
```

Task 2

The initial activation function that we used is ReLU, it is the most popular activation function. This function is less computationally expansive than tanh and sigmoid but still very efficient. The main drawback of ReLU is that in the region where x<0 gradient will be zero and the weights will not get adjusted during descent.

Zero gradient problem of ReLU can be prevented by Leaky ReLU activation function, it has a small positive slope in the negative area, which enables back-propagation for negative input values, but still the prediction it provides is not consistent in for negative values

We changed the activation function to Tanh. This model is more computationally expansive, but it solves the problem of negative values. Tanh unlike the Sigmoid is also zero-centered, which can be an advantage. Both models have a problem at large and small values of x, gradient change there is very low.

This change didn't result in better classification, the thing can be different while working with another dataset.

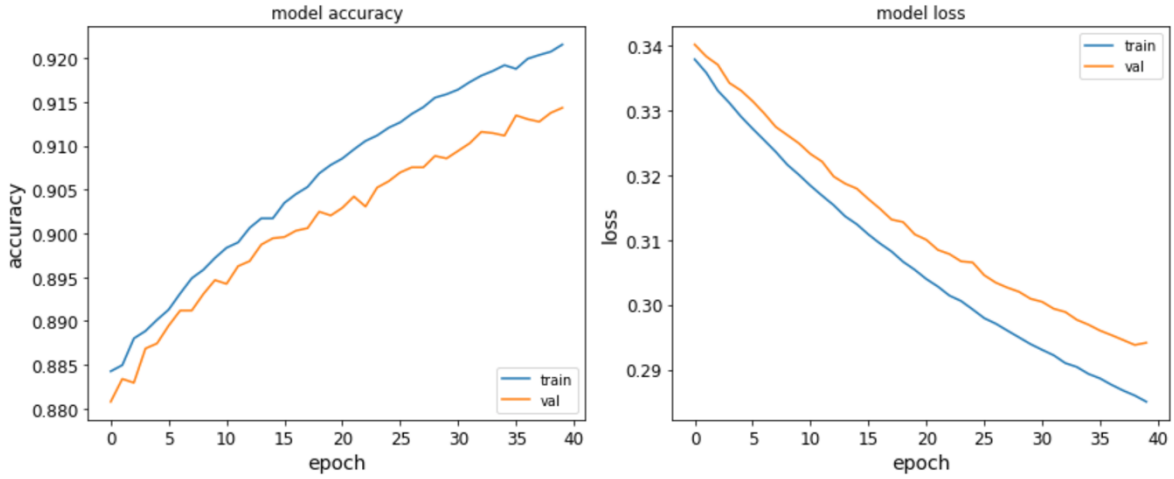
Task 3

The number of epochs defines the number of times that the learning algorithm will run through the entire training dataset. Large number of epochs allowing the learning algorithm to run until the error from the model has been sufficiently minimized. On the other hand, when the number of epochs is more than necessary, model can become overfitted, it learns patterns that are specific to sample data. This makes the model incapable to perform well on a new dataset. This model gives high accuracy on the training set but fails to achieve good accuracy on the test set.

Our results:

```
25 epochs: [test loss, test accuracy]: [0.3970744442939758, 0.85571426]
40 epochs: [test loss, test accuracy]: [0.3990015653201512, 0.8542857]
```

Performance of the model after each epoch:



Even though there is still an improvement of the model's performance between 25 and 40 epochs, it doesn't affect a test loss and a test accuracy significantly.

Task 4

Batch gradient descent computes one step using the entire dataset at once, while SGD uses only 1 element. A mixture of Batch Gradient Descent and SGD, called Mini Batch Gradient Descent, it processes a fixed number of training examples at a time, and not all or one. By doing so it can overcome SGD's main disadvantages as computational time, noise vulnerability and high fluctuations of a cost function.

Our results:

```
Batch size of 64: [test loss, test accuracy]: [0.39694390620504105, 0.8485714]
```

```
Batch size of 32: [test loss, test accuracy]: [0.3600200772285461, 0.8371429]
```

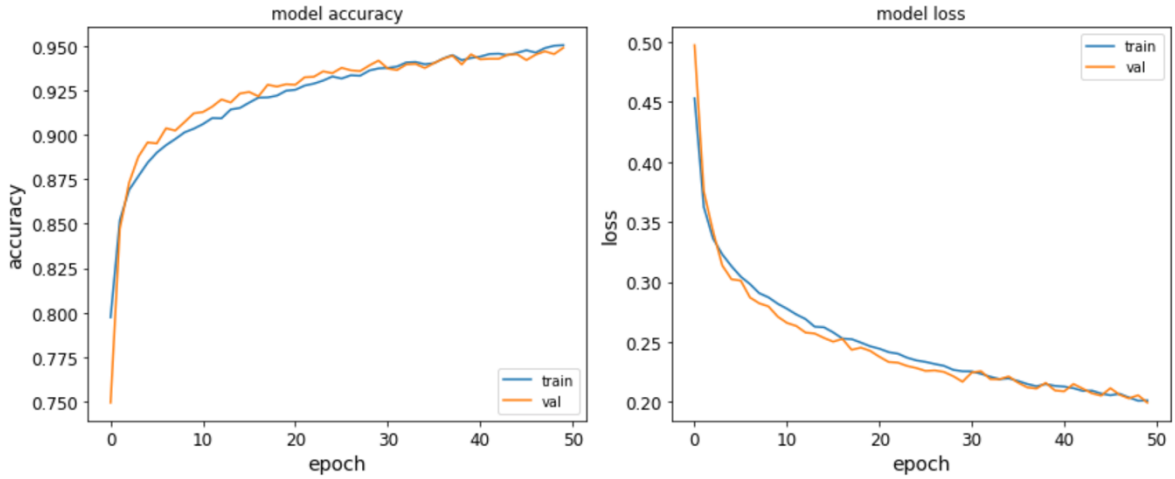
We can see that the test accuracy is slightly better when the batch size is 64, but also a test loss is a little bit higher.

Task 5

Batch-normalization is a method that improves performance and stabilizes the operation of artificial neural networks. It is implemented by calculating the mean and standard deviation of each input variable to a layer per mini-batch and using these statistics to perform the standardization. This change in the distribution of the input data can stabilize the learning process and reduce the number of epochs needed.

Our result after adding batch normalization layers:

```
[test loss, test accuracy]: [0.3933827211175646, 0.8257143]
```



As we can see, in our case adding batch normalization layers in our case didn't result in better test loss and test accuracy, but improved learning rates.

Part 2

Task 1. 2D CNN

- A model has 8 layers: 5 convolutional and 3 Dense layers
- There are (64, 128, 128, 256, 256) filters in each layer.
- Fully connected NN will have much higher number of parameters, because every neuron in a current layer is connected to every neuron of the previous layer
- Yes, this specific NN performs L2 regularization and dropout. L2 regularization penalizes parameters, reduces overfitting. Dropout drops some random nodes, as a result the parameter values will be dispersed, and the output of the layer will not depend on a single node.

Task 2. Reducing number of filters

Initial number of filters: test loss, test acc: [7.945970437186105, 0.2742857]

Number of filters reduced by half: test loss, test acc: [4.785741686139788, 0.33142856]

Since the number of filters results in number of features that network can learn, we can conclude that initial number of filters probably was redundant. After reducing the number of filters by half we've got better accuracy, meaning that initial model was overfitted.