Machine learning in Healthcare - HW4

Ofri Vizenblit 205894348
Rotem Shapira 206236366

Theoretical Questions

The numbers may change between runs.

Part 1- Fully connected layers

Task 2- Activation function

Change the activation functions to LeakyRelu or tanh or sigmoid. Name the new model new_a_model. Explain how it can affect the model.

Selecting different activation functions can affect the learning process of the network. NN are trained using gradient descent that consists of the backward propagation step which is basically the chain rule. The activation functions are directly reflected in the derivation according to the chain rule and therefore the characteristics of the function will affect the updating of the weights and the learning of the network.

In addition, the range of the outputs of the activation functions may shift the gradients to a particular direction ([0, 1] or [-1, 1]). For example, for dealing with probabilities we will prefer a range of [0, 1], like the sigmoid function.

The change of the activation functions to sigmoid, tanh and softmax causes a minor decrease in accuracy, from 0.6685 to 0.5828, and to different values of loss, 0.8149 for Relu and 0.9775 for the new_a_model.

Task 3- Number of epochs

Train the new model using 25 and 40 epochs. What difference does it makes in term of performance?

One epoch is when an entire dataset is passed through the neural network only once. As the number of epochs increases, the greater number of times the weights are changed in the neural network and the model goes from underfitting to optimal to overfitting. In our model, increasing the number of epochs to 40 causes a minor increase in the accuracy of the model, from 0.5828 to 0.6171, and decrease in the loss, from 0.9775 to 0.9253. Hence the additional epochs contributed to the convergence of the model to a minimum and to a certain improvement exactly. Yet, for both models these results are far from ideal, it may be because the model does not fit the type of data, or the data is not good enough.

Task 4- Mini batches

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

Mini-batch is the tradeoff between SGD and Batch GD. Mini-batch gradient descent attempts to find a balance between the robustness of stochastic gradient descent and the efficiency of batch gradient descent. One advantage is that the batched updates provide a more efficient process, computation-wise, than stochastic gradient descent. In addition, batching is more economical in the memory aspect since there is no need to have all the training data saved to the disk at the same time.

Running new_a_model with batch size of 32 and 50 epochs causes increase in accuracy to 0.6514.

Task 4- Batch normalization

Build the new_a_model again and add batch normalization layers. How does it impact your results?
The batch normalization in needed because the scaling of the different parameters of the network that linked to the gradient updates might lead to exploding and vanishing gradients. In addition, some activations might be pushed to their saturation areas. In our model, adding batch normalization layers causes a decrease of accuracy to the value of 0.5142 and to a loss value of 1.025. We got a decrease in the quality of the model; this may be due to the quality of the data.

Part 2- CNN

Task 1- 2D CNN

How many layers does it have?
There are 8 layers overall, 5 CNN and 3 fully connected layers.

How many filters in each layer?
There are 64, 128, 128, 256, 256 filters in layers 1-5 accordingly.

Would the number of parameters be similar to a fully connected NN?
By using CNN and filters, the number of parameters is reduced, so there are less parameters to learn vs. fully connected network of the same size.

Is this specific NN performing regularization?
Yes, this NN performing L2 regularization with value of 1e-2, and there is a dropout stage to avoid overfitting.

Task 2- Number of filters

Rebuild the function get_net to have as an input argument a list of number of filters in each layer, i.e., for the CNN defined above the input should have been [64, 128, 128, 256, 256]. Now train the model with the number of filters reduced by half. What were the results?
The smaller number of filters, the less parameters the NN have. The number of trained parameters in the original network is 3,271,332, and the number of trained parameters in the reduced network is 1,392,612. The accuracy of the original model is 0.3428 and the loss is 7.841, the accuracy of the reduced model is 0.2285, and the loss is 4.7785.
We got a decrease in the accuracy of the model for filter reduction. Note that the low accuracy values for all tasks in this section could be the cause of the mismatch between the model structure and the data, or that the data is not of good enough quality.

In both parts of this task, low accuracy data were obtained for the different models. This may be due to a mismatch of the neural network to the data or data that was not of good quality in advance.