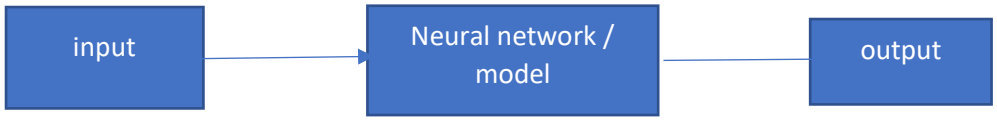
Assignment – 2 (AML)

Student Name: Aksa Taniya . Instructor: CJ wu

I categorized the assignment into 8 cases, and each case has different parameters. Before that I would like to talk about the structure and important parameters in the neural network



Input: As an input we are providing the IMDB dataset

Model: It has the hidden layers implementation and in this we can fine tune the parameters for the model performance.

Output: gives output depends on the classifier or regressor.

The important parameters refer from the keras documentation here are:

1. Number of layers: The number of layers in a deep neural network is a crucial design parameter. Too few layers can result in poor performance, while too many layers can lead to overfitting.
2. Number of neurons per layer: The number of neurons per layer determines the network's capacity to learn complex relationships in the data. A larger number of neurons can increase the network's ability to learn complex features, but also increase the risk of overfitting.
3. Activation function: The activation function is a non-linear function that introduces non-linearity into the network. Common activation functions include ReLU, sigmoid, and tanh.
4. Loss function: The loss function measures the difference between the predicted output of the network and the true output. The choice of loss function depends on the type of problem being solved. For example, for a binary classification problem, the binary cross-entropy loss function can be used.
5. Optimizer: The optimizer is an algorithm used to update the network's parameters during training. Popular optimizers include Stochastic Gradient Descent (SGD), Adam, and RMSprop.
6. Learning rate: The learning rate determines the step size at each iteration during optimization. A smaller learning rate can result in slower convergence but a more accurate solution, while a larger learning rate can result in faster convergence but may miss the optimal solution.
7. Regularization: Regularization techniques, such as dropout and weight decay, are used to prevent overfitting by introducing a penalty term on the network's parameters.

Case 1:

Parameters	Train Accuracy	Validation Accuracy	Test Accuracy
2 hidden layers with 16 hidden units Optimizer – rmsprop Loss – mse Metrics - accuracy	99.77	87.13	88.60

In the above scenario, I used 2 hidden layers with 16 hidden units. Using relu function, with rmsprop optimizer with a mean square loss I got training accuracy of 99.77, with validation and test accuracy of 87.13, 88.60 respectively.

Case 2:

Parameters	Train Accuracy	Validation Accuracy	Test Accuracy
3 layers with 16 hidden units Optimizer – rmsprop Loss – mse Metrics - accuracy	99.17	87.43	87.02

In the above scenario, I used 3 hidden layers with 16 hidden units. Using relu function, with rmsprop optimizer with a mean square loss I got training accuracy of 98.66, with validation and test accuracy of 87.54, 87.56 respectively.

Case 3:

Parameters	Train Accuracy	Validation Accuracy	Test Accuracy
1 layer with 32 hidden units Optimizer – rmsprop Loss – mse Metrics - accuracy	**98.74**	**87.65**	**88.87**

In the above scenario, I used 1 hidden layers with 32 hidden units. Using relu function, with rmsprop optimizer with a mean square loss I got training accuracy of 98.74, with validation and test accuracy of 87.65, 88.87 respectively.

Case 4:

Parameters	Train Accuracy	Validation Accuracy	Test Accuracy
3 layers with 32 hidden units Optimizer – rmsprop Loss – mse Metrics - accuracy	98.85	87.42	88.00

In the above scenario, I used 3 hidden layers with 32 hidden units. Using relu function, with rmsprop optimizer with a mean square loss I got training accuracy of 98.85, with validation and test accuracy of 87.42, 88.00 respectively.

Case 5:

Parameters	Train Accuracy	Validation Accuracy	Test Accuracy
3 layers with 64 hidden units using tanh activation. Optimizer – rmsprop	98.66	87.54	87.56

Loss – mse Metrics – accuracy			

In the above scenario, I used 3 hidden layers with 64 hidden units. Using tanh function, with rmsprop optimizer with a mean square loss I got training accuracy of 98.66, with validation and test accuracy of 87.54, 87.56 respectively.

Case 6:

Parameters	Train Accuracy	Validation Accuracy	Test Accuracy
1 layer with 64 hidden units using tanh activation. Optimizer – rmsprop Loss – mse Metrics - accuracy	98.01	87.54	88.32

In the above scenario, I used 3 hidden layers with 64 hidden units. Using tanh function, with rmsprop optimizer with a mean square loss I got training accuracy of 98.01, with validation and test accuracy of 87.54, 88.32 respectively.

Case 7:

Parameters	Train Accuracy	Validation Accuracy	Test Accuracy
1 layer with 32 hidden units Optimizer – adam Loss – mse Metrics - accuracy	89.44	88.01	86.79

In the above scenario, I used 1 hidden layers with 32 hidden units. Using tanh function, with adam optimizer with a mean square loss I got training accuracy of 89.44, with validation and test accuracy of 88.01, 86.79 respectively.

Case 8:

Parameters	Train Accuracy	Validation Accuracy	Test Accuracy
1 layer with 32 hidden units Optimizer – adam Loss – mse Metrics – accuracy L2 regularizer With dropouts	90.54	87.96	87.03

In the above scenario, I used 1 hidden layers with 32 hidden units. Using tanh function, with adam optimizer with a mean square loss I got training accuracy of 89.44, with validation and test accuracy of 88.01, 86.79 respectively.

Conclusion:

This task helps us to learn the concepts involved in the Deep Learning framework in keras. So the best accuracy I achieved here for **the validation and test accuracy is 87.65, 88.87 respectively in the case 3.**

I used the concepts of regularizers and optimizers but it seems to reduce the overall performance for this dataset. Most likely it would fit for the huge volume of the data.

So, the final combination is the case 3,

Case 3:

Parameters	Train Accuracy	Validation Accuracy	Test Accuracy
1 layer with 32 hidden units Optimizer – rmsprop Loss – mse Metrics - accuracy	**98.74**	**87.65**	**88.87**

And the following graphs are below,