**Neural Network Architecture and Hyperparameter Experiments**

1. Experiments on Validation Set

Experiment 1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lr = 0.001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 32, epoch = 100 | | | | |
| No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| 1 (50) | 0.53 | 0.53 | 0.58 | 1099.45 sec |

Experiment 2: Increase the batch size to 128

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lr = 0.001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 128, epoch = 100 | | | | |
| No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| 1 (50) | 0.54 | 0.55 | 0.6 | 434.28 sec |

Experiment 3: With batch size of 128, decrease the learning rate from Lr=0.001 to Lr=0.0001

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lr = 0.0001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 128, epoch = 100 | | | | |
| No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| 1 (50) | 0.58 | 0.59 | 0.62 | 200.68 sec |

At epoch 100, the Ave. Loss was still 0.5393, but it still produced better results. Next, we’ll increase the epoch size from 100 to 200.

Experiment 4: Training with 200 epochs: Batch size = 128, learning rate = 0.0001

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lr = 0.0001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 128, epoch = 200 | | | | |
| No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| 1 (50) | 0.57 | 0.58 | 0.61 | 388.17 sec |

At epoch 200, the Ave. Loss was at 0.48, which means that the loss only reduced by 0.06 after another 100 epochs, which took an additional 188.17 sec. The results did not improve too.

Experiment 6:

Using 100 epochs, batch size = 128, Lr = 0.0001, we will try to add more layers to our network.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Lr = 0.0001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 128, epoch = 100 | | | | | |
| Model | No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| A | 1 (50) | 0.58 | 0.59 | 0.62 | 171.04 sec |
| B | 2 (500, 50) | 0.55 | 0.56 | 0.60 | 992.17 sec |
| C | 2 (50, 5) | 0.58 | 0.59 | 0.62 | 213.36 sec |
| D | 3 (1000, 500, 50) | 0.52 | 0.52 | 0.57 | 1851.47 sec |
| E | 3 (50, 500, 50) | 0.53 | 0.54 | 0.58 | 226.24 sec |

It seems like our Model A (1 layer with 50 neurons) is still the best. Model C (2 Layers (50,5)) achieved the same results as Model A even with an additional layer and training duration. Adding additional layers to the left or to the right of our initial hidden layer (1 layer with 50 neurons) does not seem to benefit the performance of our neural network.

Experiment 7:

To try regularization, we will add Dropout(p=0.75) to Model A.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lr = 0.0001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 128, epochs = 100, Dropout (p) = 0.5 | | | | |
| No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| 1 (50) | 0.58 | 0.59 | 0.62 | 465.44 sec |

At 100th epoch, the average loss is at 0.61. Adding Dropout with probability of 0.75 of neuron being zeroed did not improve the performance of our network, even the training period was doubled.

Experiment 8:

Doubling the number of neurons from 50 to 100.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lr = 0.0001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 128, epochs = 100 | | | | |
| No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| 1 (100) | 0.58 | 0.58 | 0.61 | 587.16 sec |

Increasing the number of neurons extended the training period, but the network did not benefit from these changes. A network with 1 hidden layer of 50 neurons still performed the best.

**In the end**, we will use these specifications for our model:

Number of Hidden Layers: **1**

Number of Neurons in Hidden Layer: **50**

Epochs = **100**, Batch size = **128**, Learning rate = **0.0001**, Loss\_fn = **CrossEntropyLoss()**, optimizer = **Adam**

1. Result on Test Set

**Full Training Set Model Results on Test Set**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lr = 0.0001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 128, epochs = 100 | | | | |
| No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| 1 (50) | 0.60 | 0.61 | 0.63 | 240.45 sec |

1. Results on Test Set of Models Trained with Balanced Dataset

**With AutoEncoder**

Experiment 1:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lr = 0.0001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 128, epoch = 100 | | | | |
| No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| 1 (50) | 0.77 | 0.78 | 0.78 | 1288.15 sec |

Experiment 2:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lr = 0.0001, Loss\_fn = CrossEntropyLoss(), optimizer = Adam, batch\_size = 128, epoch = 200 | | | | |
| No. of Hidden Layers | Macro-Recall | F1-Macro | Accuracy | Training Period |
| 1 (50) | 0.85 | 0.86 | 0.85 | 1451.29 sec |

Increasing the epoch size by 100 achieved outstanding results: **+0.08** on recall, **+0.08** on F1-Macro, and **+0.07** on Accuracy. It cost an additional ~150 seconds (2 ½ minutes) but the results outweighed the costs.