Tuning the Airplane Balance, Nordic Hamstring Curl, and Squat Jump Machine Learning Model Hyperparameters with Validation and Testing data

Airplane Balance Model Test 1

Layers:

1. ReLU (80)
2. ReLU (1)

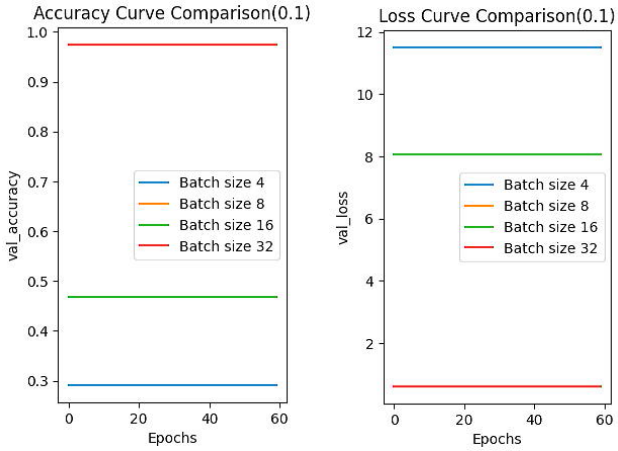
Epochs: 60

Learning rate: 0.1

Observations:

Unfortunetly the validation loss did not change throughout the training of this model unlike the accuracy. This is problematic due to the loss being simply too high.

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a lay
er. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 3s 27ms/step - accuracy: 0.2577 - loss: 10.9795 - val_accuracy: 0.2911 - val_loss: 11.5072
Epoch 60/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.2388 - loss: 10.9721 - val_accuracy: 0.2911 - val_loss: 11.5072
Training with batch size: 8
Epoch 1/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 41ms/step - accuracy: 0.8055 - loss: 3.5566 - val_accuracy: 0.9747 - val_loss: 0.6121
Epoch 60/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9684 - loss: 1.8490 - val_accuracy: 0.9747 - val_loss: 0.6121
Training with batch size: 16
Epoch 1/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 3s 83ms/step - accuracy: 0.3786 - loss: 9.3444 - val_accuracy: 0.4684 - val_loss: 8.0721
Epoch 60/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 33ms/step - accuracy: 0.3822 - loss: 7.5737 - val_accuracy: 0.4684 - val_loss: 8.0721
Training with batch size: 32
Epoch 1/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 3s 167ms/step - accuracy: 0.8548 - loss: 3.2418 - val_accuracy: 0.9747 - val_loss: 0.6121
Epoch 60/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 47ms/step - accuracy: 0.9691 - loss: 1.4801 - val_accuracy: 0.9747 - val_loss: 0.6121
```



Airplane Balance Model Test 2

Layers:

1. ReLU (80)
2. Sigmoid (1)

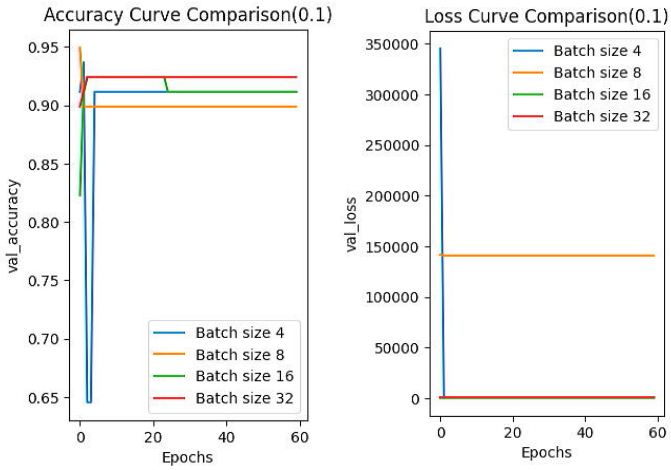
Epochs: 60

Learning rate: 0.1

Observations:

Sometimes the accuracy of the predictions did not increase and sometimes it barely increased or decreased. The loss also was just too high which rendered this model useless.

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a lay
er. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 4s 28ms/step - accuracy: 0.8117 - loss: 11410531328.0000 - val_accuracy: 0.9114 - val_loss: 345184.0625
Epoch 60/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9712 - loss: 0.1894 - val_accuracy: 0.9114 - val_loss: 941.5818
Training with batch size: 8
Epoch 1/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 3s 42ms/step - accuracy: 0.6796 - loss: 2713758203904.0000 - val_accuracy: 0.9494 - val_loss: 141656.8750
Epoch 60/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9687 - loss: 0.1976 - val_accuracy: 0.8987 - val_loss: 140778.5000
Training with batch size: 16
Epoch 1/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 3s 82ms/step - accuracy: 0.8843 - loss: 5627.3735 - val_accuracy: 0.8228 - val_loss: 172.0406
Epoch 60/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step - accuracy: 0.9823 - loss: 0.0996 - val_accuracy: 0.9114 - val_loss: 150.9310
Training with batch size: 32
Epoch 1/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 2s 164ms/step - accuracy: 0.7775 - loss: 3087.6846 - val_accuracy: 0.8987 - val_loss: 968.8321
Epoch 60/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 49ms/step - accuracy: 0.9673 - loss: 0.1575 - val_accuracy: 0.9241 - val_loss: 932.4163
```



Airplane Balance Model Test 3

Layers:

1. Sigmoid
2. ReLU

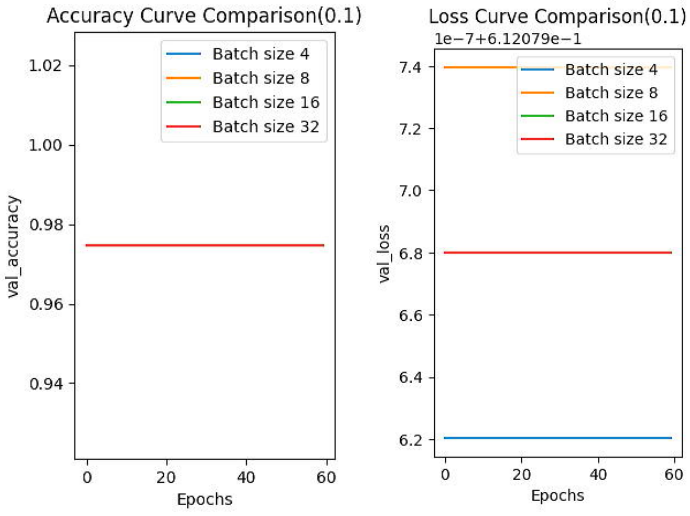
Epochs: 60

Learning rate: 0.1

Observations:

The loss also didn't change and stayed too high to continue using these settings.

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a lay
er. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 3s 24ms/step - accuracy: 0.9633 - loss: 1.7167 - val accuracy: 0.9747 - val loss: 0.6121
Epoch 60/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9770 - loss: 0.8260 - val_accuracy: 0.9747 - val_loss: 0.6121
Training with batch size: 8
Epoch 1/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 44ms/step - accuracy: 0.8537 - loss: 2.3214 - val_accuracy: 0.9747 - val_loss: 0.6121
Epoch 60/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - accuracy: 0.9659 - loss: 1.8153 - val_accuracy: 0.9747 - val_loss: 0.6121
Training with batch size: 16
Epoch 1/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 84ms/step - accuracy: 0.9625 - loss: 1.5610 - val accuracy: 0.9747 - val loss: 0.6121
Epoch 60/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step - accuracy: 0.9654 - loss: 1.4247 - val_accuracy: 0.9747 - val_loss: 0.6121
Training with batch size: 32
Epoch 1/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 2s 162ms/step - accuracy: 0.6321 - loss: 4.4960 - val accuracy: 0.9747 - val loss: 0.6121
Epoch 60/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 49ms/step - accuracy: 0.9752 - loss: 1.2402 - val_accuracy: 0.9747 - val_loss: 0.6121
```



Airplane Balance Model Test 4

Layers:

1. Sigmoid (80)
2. Sigmoid (1)

Epochs: 60 (just to determine where the accuracy and loss plateaus)

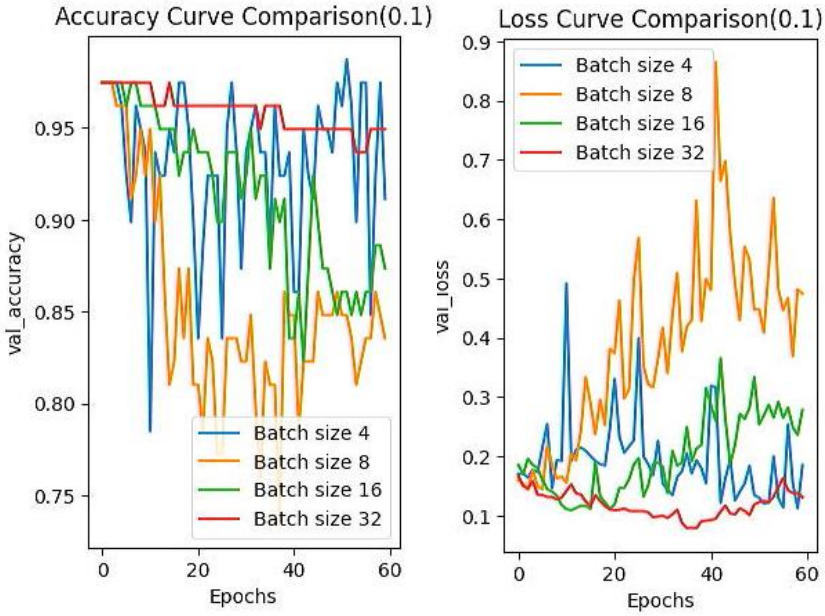
Learning rate: 0.1

Observations:

The loss value got higher than the starting loss which is not ideal. However, I saw that the loss decreased in the 5-10 epoch range before rising again which gave me hope that this model could still work if it was trained for less time.

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a lay
er. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 4s 25ms/step - accuracy: 0.8666 - loss: 0.4560 - val_accuracy: 0.9747 - val_loss: 0.1705
Epoch 60/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - accuracy: 0.9752 - loss: -0.0995 - val_accuracy: 0.9114 - val_loss: 0.1855
Training with batch size: 8
Epoch 1/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 46ms/step - accuracy: 0.9469 - loss: 0.3921 - val_accuracy: 0.9747 - val_loss: 0.1604
Epoch 60/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - accuracy: 0.9160 - loss: -0.2028 - val_accuracy: 0.8354 - val_loss: 0.4743
Training with batch size: 16
Epoch 1/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 82ms/step - accuracy: 0.9311 - loss: 0.3926 - val_accuracy: 0.9747 - val_loss: 0.1859
Epoch 60/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step - accuracy: 0.9645 - loss: -0.1116 - val_accuracy: 0.8734 - val_loss: 0.2786
Training with batch size: 32
Epoch 1/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 3s 164ms/step - accuracy: 0.8968 - loss: 0.4282 - val_accuracy: 0.9747 - val_loss: 0.1677
```

```
Epoch 60/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 1s 48ms/step - accuracy: 0.9265 - loss: -0.5330 - val_accuracy: 0.9494 - val_loss: 0.1313
```



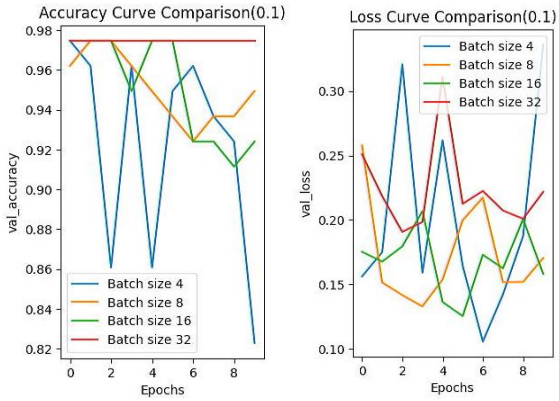
Airplane Balance Model Test 4-B

Same settings as test 4 with the epochs set to 10 instead of 60

Obervations:

The results improved but this model did not provide the same accuracy compared with those that were trained with slower learning rates.

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a lay
er. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 3s 24ms/step - accuracy: 0.9188 - loss: 0.4905 - val_accuracy: 0.9747 - val_loss: 0.1562
Epoch 10/10
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - accuracy: 0.9419 - loss: 0.1304 - val_accuracy: 0.8228 - val_loss: 0.3362
Training with batch size: 8
Epoch 1/10
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 38ms/step - accuracy: 0.9864 - loss: 0.2552 - val_accuracy: 0.9620 - val_loss: 0.2579
Epoch 10/10
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - accuracy: 0.9344 - loss: -0.1248 - val_accuracy: 0.9494 - val_loss: 0.1704
Training with batch size: 16
Epoch 1/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 75ms/step - accuracy: 0.7355 - loss: 0.5050 - val_accuracy: 0.9747 - val_loss: 0.1753
Epoch 10/10
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step - accuracy: 0.9389 - loss: 0.0219 - val_accuracy: 0.9241 - val_loss: 0.1581
Training with batch size: 32
Epoch 1/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 2s 142ms/step - accuracy: 0.8675 - loss: 0.4734 - val_accuracy: 0.9747 - val_loss: 0.2509
Epoch 10/10
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 48ms/step - accuracy: 0.9715 - loss: 0.0754 - val_accuracy: 0.9747 - val_loss: 0.2217
```



Airplane Balance Model Test 5

Layers:

1. Sigmoid (80)
2. Sigmoid (1)

Epochs: 60 (Just to see where the accuracy and loss plateaus)

Learning rate: 0.01

Observations:

This model was a good compromise between true positives and true negatives which resulted in the overall best combined accuracy. This specific model achieved my goal of limiting the amount of false positives.

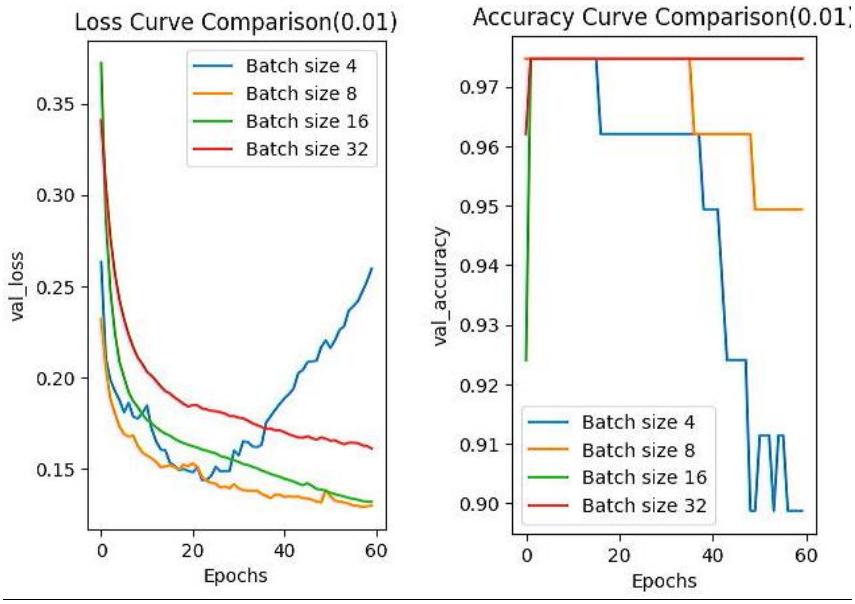
```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a lay
er. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 3s 30ms/step - accuracy: 0.5434 - loss: 0.7303 - val_accuracy: 0.9747 - val_loss: 0.2632

Epoch 60/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - accuracy: 0.9437 - loss: -0.0545 - val_accuracy: 0.8987 - val_loss: 0.2595
Training with batch size: 8
Epoch 1/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 42ms/step - accuracy: 0.9094 - loss: 0.5168 - val_accuracy: 0.9747 - val_loss: 0.2322

Epoch 60/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step - accuracy: 0.9680 - loss: -0.1442 - val_accuracy: 0.9494 - val_loss: 0.1298
Training with batch size: 16
Epoch 1/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 83ms/step - accuracy: 0.6812 - loss: 0.6086 - val_accuracy: 0.9241 - val_loss: 0.3722

Epoch 60/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 41ms/step - accuracy: 0.9886 - loss: 0.0389 - val_accuracy: 0.9747 - val_loss: 0.1319
Training with batch size: 32
Epoch 1/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 3s 175ms/step - accuracy: 0.9312 - loss: 0.4218 - val_accuracy: 0.9620 - val_loss: 0.3408
```

```
Epoch 60/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 48ms/step - accuracy: 0.9706 - loss: 0.1033 - val_accuracy: 0.9747 - val_loss: 0.1611
```



Validation dataset

```
classification_report:
              precision    recall  f1-score   support

           0       0.78      0.85      0.81        33
           1       0.88      0.83      0.85        46

    accuracy                           0.84        79
   macro avg       0.83      0.84      0.83        79
weighted avg       0.84      0.84      0.84        79
```

```
confusion matrix:
[[28  5]
 [ 8 38]]
```

Confusion Matrix Legend

Actual sample value \ Predicted Class	Negative	Positive
Negative	TN	FP
Positive	FN	TP

Training dataset

```
classification_report:
              precision    recall  f1-score   support

           0       0.74      0.82      0.78        34
           1       0.82      0.74      0.78        38

    accuracy                           0.78        72
   macro avg       0.78      0.78      0.78        72
weighted avg       0.78      0.78      0.78        72
```

```
confusion matrix:
[[28  6]
 [10 28]]
```

Airplane Balance Model Test 6

Layers:

1. Sigmoid (80)
2. Sigmoid (1)

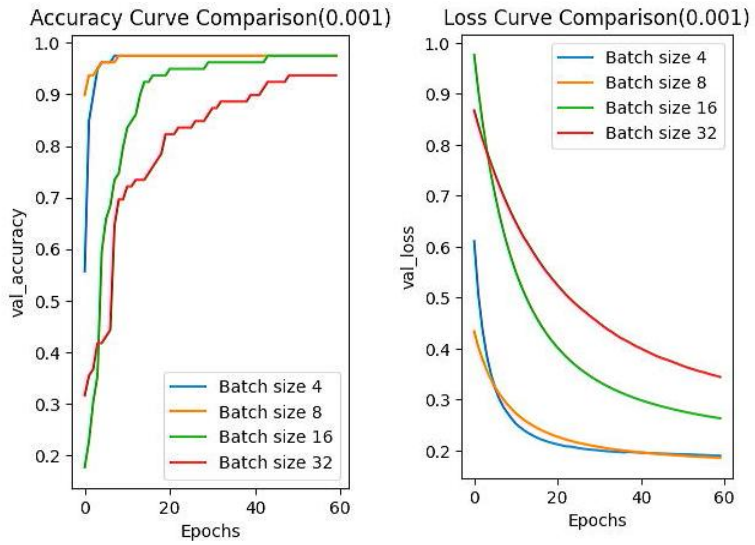
Epochs: 60 (Just to see where the accuracy and loss plateaus)

Learning rate: 0.001

Observations:

The amount of loss was accpetable, however the amount of false positives were higher than the model with the learning rate of 0.01 (Test 5).

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a lay
er. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 3s 27ms/step - accuracy: 0.4795 - loss: 0.7072 - val_accuracy: 0.5570 - val_loss: 0.6109
Epoch 60/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 48ms/step - accuracy: 0.9566 - loss: 0.3231 - val_accuracy: 0.9367 - val_loss: 0.3444
Epoch 60/60
44/44 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9735 - loss: 0.1578 - val_accuracy: 0.9747 - val_loss: 0.1898
Training with batch size: 8
Epoch 1/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 2s 41ms/step - accuracy: 0.8694 - loss: 0.5482 - val_accuracy: 0.8987 - val_loss: 0.4343
Epoch 60/60
22/22 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.9504 - loss: 0.3176 - val_accuracy: 0.9747 - val_loss: 0.1859
Training with batch size: 16
Epoch 1/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 2s 83ms/step - accuracy: 0.1653 - loss: 0.9844 - val_accuracy: 0.1772 - val_loss: 0.9765
Epoch 60/60
11/11 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step - accuracy: 0.9761 - loss: 0.2492 - val_accuracy: 0.9747 - val_loss: 0.2633
Training with batch size: 32
Epoch 1/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 2s 171ms/step - accuracy: 0.4341 - loss: 0.7365 - val_accuracy: 0.3165 - val_loss: 0.8675
```



6A-batch size 4:

Validation dataset

```
confusion matrix:
[[26  7]
 [ 8 38]]
```

```
classification_report:
              precision    recall  f1-score   support

           0       0.76      0.79      0.78        33
           1       0.84      0.83      0.84        46

    accuracy                           0.81        79
   macro avg       0.80      0.81      0.81        79
weighted avg       0.81      0.81      0.81        79
```

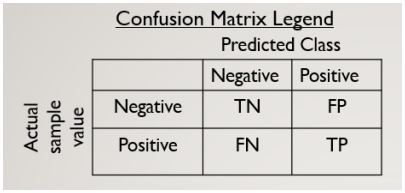
Test dataset

```
confusion matrix:
[[23 11]
 [ 6 32]]
```

```
classification_report:
              precision    recall  f1-score   support

           0       0.79      0.68      0.73        34
           1       0.74      0.84      0.79        38

    accuracy                           0.76        72
   macro avg       0.77      0.76      0.76        72
weighted avg       0.77      0.76      0.76        72
```



6B-batch size 8:

Validation dataset

```
confusion matrix:
[[23 10]
 [ 6 40]]
```

```
classification_report:
              precision    recall  f1-score   support

           0       0.79      0.70      0.74        33
           1       0.80      0.87      0.83        46

    accuracy                           0.80        79
   macro avg       0.80      0.78      0.79        79
weighted avg       0.80      0.80      0.80        79
```

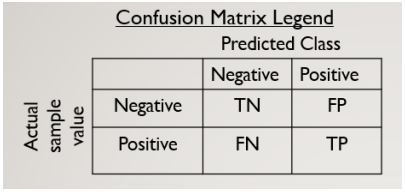
Test dataset

```
confusion matrix:
[[21 13]
 [ 5 33]]
```

```
classification_report:
              precision    recall  f1-score   support

           0       0.81      0.62      0.70        34
           1       0.72      0.87      0.79        38

    accuracy                           0.75        72
   macro avg       0.76      0.74      0.74        72
weighted avg       0.76      0.75      0.75        72
```



Final Airplane Balance (APB) Model Settings:

Layer 1 = Sigmoid (80)

Layer 2 = Sigmoid (1)

Learning Rate = 0.01

Batch Size = 8

Epochs = 40

Observations:

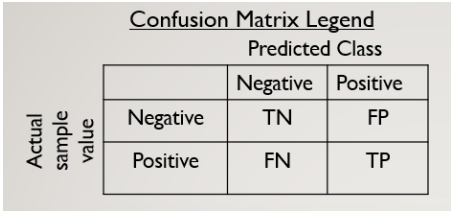
Validation Dataset Results:

```
classification_report:
              precision    recall  f1-score   support

           0       0.78      0.85      0.81        33
           1       0.88      0.83      0.85        46

    accuracy                           0.84        79
   macro avg       0.83      0.84      0.83        79
weighted avg       0.84      0.84      0.84        79
```

```
confusion matrix:
[[28  5]
 [ 8 38]]
```



Test Dataset Results:

```
classification_report:
              precision    recall  f1-score   support

           0       0.74      0.82      0.78        34
           1       0.82      0.74      0.78        38

    accuracy                           0.78        72
   macro avg       0.78      0.78      0.78        72
weighted avg       0.78      0.78      0.78        72
```

```
confusion matrix:
[[28  6]
 [10 28]]
```

Nordic Hamstring Curl Model Test 1

Layers:

1. Sigmoid (64)
2. ReLU (1)

Epochs: 40

Learning rate: 0.01

Observations:

The loss did not improve at all during the training of the model resulting in a high amount of loss on the validation data set.

```
Training with batch size: 4
Epoch 1/40
46/46 ━━━━━━━━━━━━━━━━━━━━ 2s 19ms/step - accuracy: 0.9718 - loss: 0.1461 - val_accuracy: 0.9844 - val_loss: 0.7555
Epoch 2/40
46/46 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9963 - loss: 0.0853 - val_accuracy: 0.9844 - val_loss: 0.7555
Epoch 3/40
Training with batch size: 8
Epoch 1/40
23/23 ━━━━━━━━━━━━━━━━━━━━ 2s 36ms/step - accuracy: 0.9592 - loss: 0.4241 - val_accuracy: 0.9844 - val_loss: 0.7555
Epoch 2/40
23/23 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - accuracy: 0.9910 - loss: 0.2346 - val_accuracy: 0.9844 - val_loss: 0.7555
Training with batch size: 16
Epoch 1/40
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 68ms/step - accuracy: 0.7433 - loss: 2.7887 - val_accuracy: 0.9844 - val_loss: 0.7555
Epoch 2/40
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 0.9947 - loss: 0.1142 - val_accuracy: 0.9844 - val_loss: 0.7555
Training with batch size: 32
Epoch 1/40
6/6 ━━━━━━━━━━━━━━━━━━━━ 3s 215ms/step - accuracy: 0.9692 - loss: 0.0532 - val_accuracy: 0.9844 - val_loss: 0.7620
Epoch 2/40
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 47ms/step - accuracy: 0.9929 - loss: 0.1722 - val_accuracy: 0.9844 - val_loss: 0.7620
Epoch 3/40
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 65ms/step - accuracy: 0.9867 - loss: 0.2394 - val_accuracy: 0.9844 - val_loss: 0.7620
Epoch 4/40
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 52ms/step - accuracy: 0.9867 - loss: 0.4037 - val_accuracy: 0.9844 - val_loss: 0.7620
Epoch 5/40
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 49ms/step - accuracy: 0.9832 - loss: 0.4600 - val_accuracy: 0.9844 - val_loss: 0.7620
```

Nordic Hamstring Curl Model Test 2

Layers:

1. ReLU (64)
2. Sigmoid (1)

Epochs: 40

Learning rate: 0.01

Observations:

Resulted in high amounts of loss for all except the batch of size 8 where the loss peaked after the 9th epoch before continually becoming worse and worse until the end of the training

Batch Size 8:

```
Epoch 9/40
23/23 ━━━━━━━━━━━━━━━━━━━━ 1s 21ms/step - accuracy: 0.9945 - loss: 2.4511 - val_accuracy: 0.9844 - val_loss: 0.0502
```

Nordic Hamstring Curl Model Test 3

Layers:

1. ReLU (64)
2. ReLU (1)

Epochs: 40

Learning rate: 0.01

Observations:

Validation loss did not change during training. The whole model was poorly preforming, and the amount of loss was more prevalent with the larger batch sizes.

```
Epoch 38/40
46/46 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9827 - loss: 0.3578 - val_accuracy: 0.9844 - val_loss: 0.7555
Epoch 39/40
46/46 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9870 - loss: 0.2898 - val_accuracy: 0.9844 - val_loss: 0.7555
Epoch 40/40
46/46 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9897 - loss: 0.3040 - val_accuracy: 0.9844 - val_loss: 0.7555
```

Nordic Hamstring Curl Model Test 4

Layers:

1. Sigmoid (64)
2. Sigmoid (1)

Epochs: 60 (just to determine where the accuracy and loss plateaus)

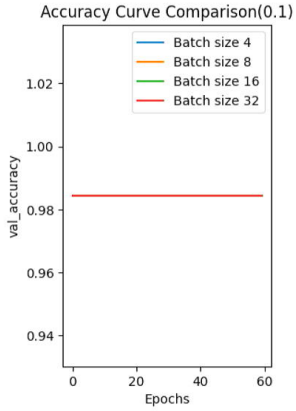
Learning rate: 0.1

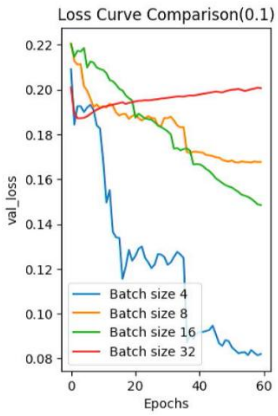
Observations:

Produced the best results for accuracy with the smallest amount of loss. Preformed better on the validation data then the next best model meaning that this was my top choice.

```
Training with batch size: 4
Epoch 1/60
46/46 ━━━━━━━━━━━━━━━━━━━━ 2s 22ms/step - accuracy: 0.8943 - loss: 0.2889 - val_accuracy: 0.9844 - val_loss: 0.2091
Epoch 60/60
46/46 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9933 - loss: -0.0611 - val_accuracy: 0.9844 - val_loss: 0.0820
Training with batch size: 8
Epoch 1/60
23/23 ━━━━━━━━━━━━━━━━━━━━ 3s 51ms/step - accuracy: 0.8753 - loss: 0.2729 - val_accuracy: 0.9844 - val_loss: 0.2206
Epoch 60/60
23/23 ━━━━━━━━━━━━━━━━━━━━ 1s 24ms/step - accuracy: 0.9969 - loss: -0.0227 - val_accuracy: 0.9844 - val_loss: 0.1677
Training with batch size: 16
Epoch 1/60
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 78ms/step - accuracy: 0.6879 - loss: 0.6252 - val_accuracy: 0.9844 - val_loss: 0.2203
Epoch 60/60
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - accuracy: 0.9881 - loss: -0.0058 - val_accuracy: 0.9844 - val_loss: 0.1485
Training with batch size: 32
Epoch 1/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 2s 150ms/step - accuracy: 0.9875 - loss: 0.2371 - val_accuracy: 0.9844 - val_loss: 0.2010
```

```
Epoch 60/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step - accuracy: 0.9949 - loss: 0.0180 - val_accuracy: 0.9844 - val_loss: 0.2007
```






```
classification_report:
              precision    recall  f1-score   support

           0       0.96      0.86      0.91        29
           1       0.89      0.97      0.93        35

    accuracy                           0.92        64
   macro avg       0.93      0.92      0.92        64
weighted avg       0.93      0.92      0.92        64
```

```
confusion matrix:
[[25  4]
 [ 1 34]]
```

Confusion Matrix Legend

Actual sample value	Predicted Class: Negative	Predicted Class: Positive
Negative	TN	FP
Positive	FN	TP

Nordic Hamstring Curl Model Test 5

Layers:

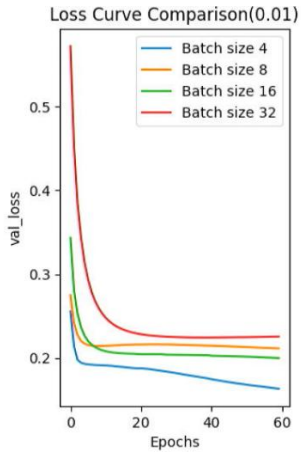
1. Sigmoid (64)
2. Sigmoid (1)

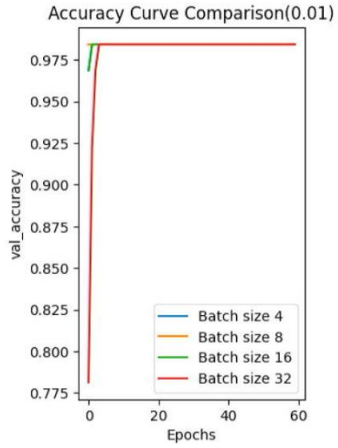
Epochs: 60 (Just to see where the accuracy and loss plateau)

Learning rate: 0.01

Observations:

It was overall the second best model with low amounts of loss and high accuracy. Unfortunetly, it struggled on the validation data creating 6 false positives and 4 false negatives resulting in a 84% accuracy.






```
classification_report:
              precision    recall  f1-score   support

           0       0.85      0.79      0.82        29
           1       0.84      0.89      0.86        35

    accuracy                           0.84        64
   macro avg       0.84      0.84      0.84        64
weighted avg       0.84      0.84      0.84        64
```

```
confusion matrix:
[[23  6]
 [ 4 31]]
```

Confusion Matrix Legend

Actual sample value	Predicted Class: Negative	Predicted Class: Positive
Negative	TN	FP
Positive	FN	TP

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
46/46 ━━━━━━━━━━━━━━━━━━━━ 2s 21ms/step - accuracy: 0.7308 - loss: 0.5235 - val_accuracy: 0.9688 - val_loss: 0.2556
Epoch 60/60
46/46 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - accuracy: 0.9917 - loss: 0.0309 - val_accuracy: 0.9844 - val_loss: 0.1630
Training with batch size: 8
Epoch 1/60
23/23 ━━━━━━━━━━━━━━━━━━━━ 2s 36ms/step - accuracy: 0.9833 - loss: 0.2823 - val_accuracy: 0.9844 - val_loss: 0.2749
Epoch 60/60
23/23 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step - accuracy: 0.9953 - loss: 0.0306 - val_accuracy: 0.9844 - val_loss: 0.2112
Training with batch size: 16
Epoch 1/60
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 71ms/step - accuracy: 0.8920 - loss: 0.4099 - val_accuracy: 0.9688 - val_loss: 0.3435
Epoch 60/60
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9841 - loss: 0.1015 - val_accuracy: 0.9844 - val_loss: 0.1996
Training with batch size: 32
Epoch 1/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 3s 173ms/step - accuracy: 0.5943 - loss: 0.7088 - val_accuracy: 0.7812 - val_loss: 0.5725
Epoch 60/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 45ms/step - accuracy: 0.9903 - loss: 0.0516 - val_accuracy: 0.9844 - val_loss: 0.2253
```

Nordic Hamstring Curl Model Test 6

Layers:

1. Sigmoid (64)
2. Sigmoid (1)

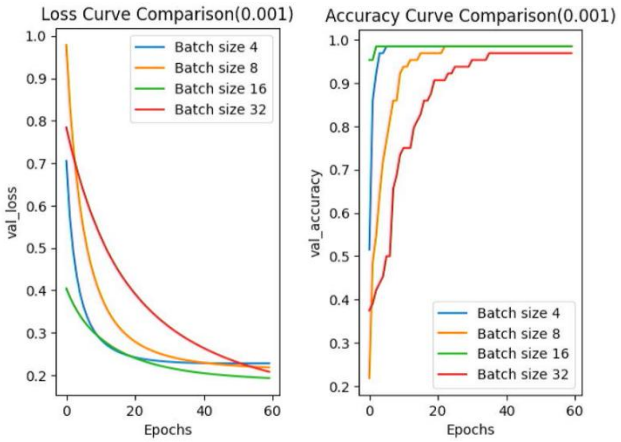
Epochs: 60 (Just to see where the accuracy and loss plateaus)

Learning rate: 0.001

Observations:

The validation loss even after training was simply too high for me to continue with the setting of this model. However, it was interesting to see that the batch size of 4 didn't have the least amount of loss.

```
Training with batch size: 4
Epoch 1/60
46/46 ━━━━━━━━━━━━━━━━━━━━ 2s 20ms/step - accuracy: 0.4422 - loss: 0.7312 - val_accuracy: 0.5156 - val_loss: 0.7050
Epoch 60/60
46/46 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9822 - loss: 0.1056 - val_accuracy: 0.9844 - val_loss: 0.2278
Training with batch size: 8
Epoch 1/60
23/23 ━━━━━━━━━━━━━━━━━━━━ 2s 37ms/step - accuracy: 0.1750 - loss: 1.0729 - val_accuracy: 0.2188 - val_loss: 0.9785
Epoch 60/60
23/23 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.9911 - loss: 0.0831 - val_accuracy: 0.9844 - val_loss: 0.2182
Training with batch size: 16
Epoch 1/60
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 72ms/step - accuracy: 0.9002 - loss: 0.4305 - val_accuracy: 0.9531 - val_loss: 0.4042
Epoch 60/60
12/12 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - accuracy: 0.9866 - loss: 0.1153 - val_accuracy: 0.9844 - val_loss: 0.1930
Training with batch size: 32
Epoch 1/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 2s 151ms/step - accuracy: 0.3382 - loss: 0.8589 - val_accuracy: 0.3750 - val_loss: 0.7839
Epoch 60/60
6/6 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step - accuracy: 0.9863 - loss: 0.1983 - val_accuracy: 0.9688 - val_loss: 0.2080
```



Final Settings for the Nordic Hamstring Curl Model:

Layer 1 = Sigmoid (64)

Layer 2 = Sigmoid (1)

Learning Rate = 0.1

Batch Size = 4

Epochs = 40

Observations:

The model performed well on the validation data resulting in a 92-93% accuracy and struggled less with false positives and negative than the next most accurate competitor.

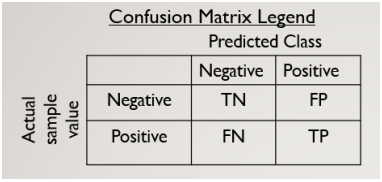
Validation Dataset Results

```
classification_report:
              precision    recall  f1-score   support

           0       0.96      0.86      0.91        29
           1       0.89      0.97      0.93        35

    accuracy                           0.92        64
   macro avg       0.93      0.92      0.92        64
weighted avg       0.93      0.92      0.92        64
```

```
confusion matrix:
[[25  4]
 [ 1 34]]
```



Test Dataset Results

```
classification_report:
              precision    recall  f1-score   support

           0       0.90      0.95      0.92        19
           1       0.97      0.94      0.95        33

    accuracy                           0.94        52
   macro avg       0.93      0.94      0.94        52
weighted avg       0.94      0.94      0.94        52
```

```
confusion matrix:
[[18  1]
 [ 2 31]]
```

Squat Jump Model Test 1

Layers:

1. ReLU (40)
2. ReLU (1)

Epochs: 60

Learning rate: 0.1

Observations:

All through the 60 epochs, each batch size's validation loss never changed indicating the model was barely improving.

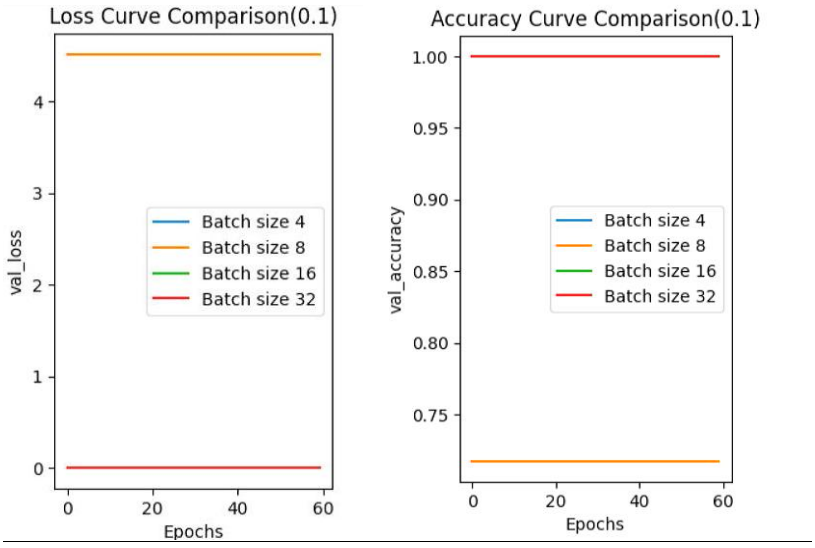
```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 3s 22ms/step - accuracy: 0.9634 - loss: 1.1420 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
```

```
Epoch 60/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9604 - loss: 1.9641 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
Training with batch size: 8
Epoch 1/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 2s 37ms/step - accuracy: 0.6219 - loss: 5.6155 - val_accuracy: 0.7172 - val_loss: 4.5148
```

```
Epoch 60/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.7111 - loss: 4.8570 - val_accuracy: 0.7172 - val_loss: 4.5148
Training with batch size: 16
Epoch 1/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 3s 70ms/step - accuracy: 0.5248 - loss: 6.7628 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
```

```
Epoch 60/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 25ms/step - accuracy: 0.9763 - loss: 0.9194 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
Training with batch size: 32
Epoch 1/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 3s 167ms/step - accuracy: 0.7849 - loss: 4.0372 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
```

```
Epoch 60/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 46ms/step - accuracy: 0.9852 - loss: 0.6671 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
```



Squat Jump Model Test 2

Layers:

1. ReLU (40)
2. Sigmoid (1)

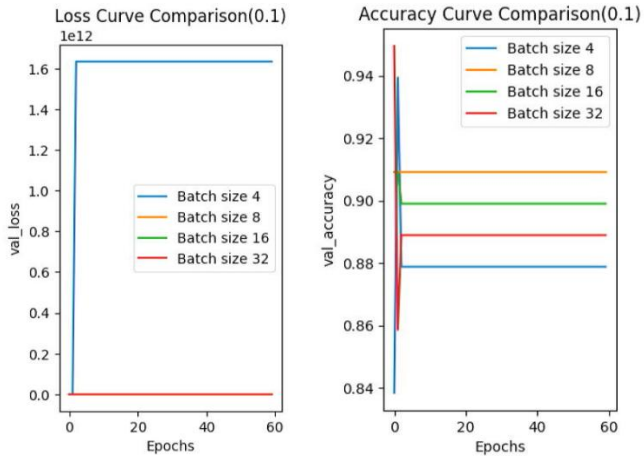
Epochs: 60

Learning rate: 0.1

Observations:

With this model, the loss and accuray was inconsistent between batch sizes. Additionally, the validation dataset proved to be problematic for the model.

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 3s 20ms/step - accuracy: 0.8075 - loss: 74433.1953 - val_accuracy: 0.8384 - val_loss: 390.9991
Epoch 60/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 16ms/step - accuracy: 0.9800 - loss: 0.2651 - val_accuracy: 0.8788 - val_loss: 1633046626304.0000
Training with batch size: 8
Epoch 1/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 2s 36ms/step - accuracy: 0.7871 - loss: 396269.7188 - val_accuracy: 0.9091 - val_loss: 21700.2559
Epoch 60/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9298 - loss: 0.0312 - val_accuracy: 0.9091 - val_loss: 17509.2070
Training with batch size: 16
Epoch 1/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 2s 70ms/step - accuracy: 0.7719 - loss: 588.0164 - val_accuracy: 0.9091 - val_loss: 4788.0308
Epoch 60/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 42ms/step - accuracy: 0.9818 - loss: 0.1135 - val_accuracy: 0.8990 - val_loss: 10026.0312
Training with batch size: 32
Epoch 1/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 2s 141ms/step - accuracy: 0.5457 - loss: 742536.1875 - val_accuracy: 0.9495 - val_loss: 1492.1595
Epoch 2/60
Epoch 60/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - accuracy: 0.9855 - loss: 0.1515 - val_accuracy: 0.8889 - val_loss: 8476091.0000
```



Squat Jump Model Test 3

Layers:

1. Sigmoid (40)
2. ReLU (1)

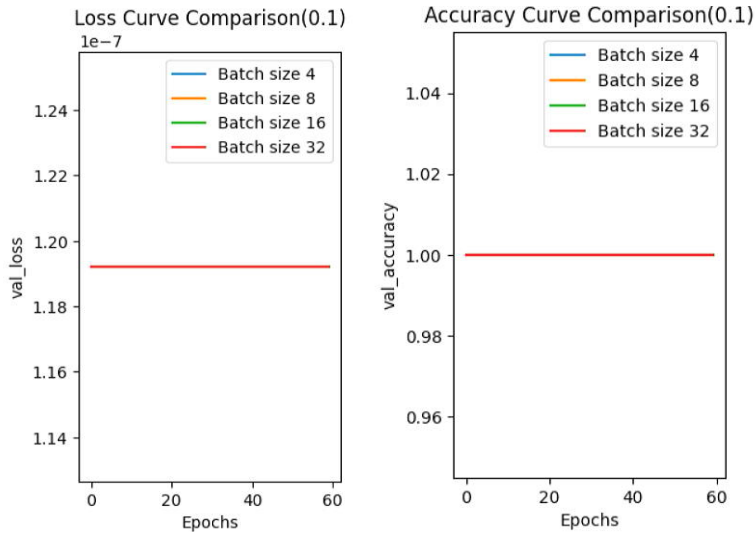
Epochs: 60

Learning rate: 0.1

Observations:

Throughout every batch size, the validation loss and the accuracy did not change indicating that there is an issue with this model.

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `in
put_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in
the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 4s 31ms/step - accuracy: 0.9083 - loss: 2.0287 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
Epoch 2/60
Epoch 60/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9848 - loss: 0.6742 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
Training with batch size: 8
Epoch 1/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 2s 36ms/step - accuracy: 0.9374 - loss: 1.3000 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
Epoch 60/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.9777 - loss: 1.1815 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
Training with batch size: 16
Epoch 1/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 3s 69ms/step - accuracy: 0.9794 - loss: 1.1633 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
Epoch 60/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 49ms/step - accuracy: 0.9774 - loss: 1.1393 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
Training with batch size: 32
Epoch 1/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 3s 144ms/step - accuracy: 0.8858 - loss: 1.4582 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
Epoch 60/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 47ms/step - accuracy: 0.9791 - loss: 1.2432 - val_accuracy: 1.0000 - val_loss: 1.1921e-07
```



Squat Jump Model Test 4

Layers:

1. Sigmoid (40)
2. Sigmoid (1)

Epochs: 60 (just to determine where the accuracy and loss plateau)

Learning rate: 0.1

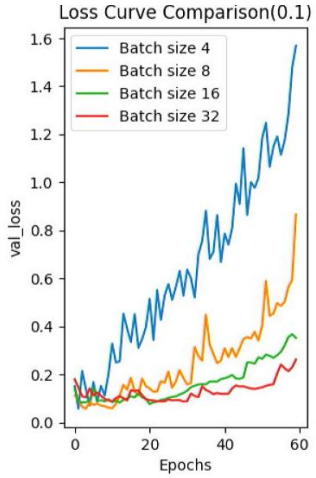
Observations:

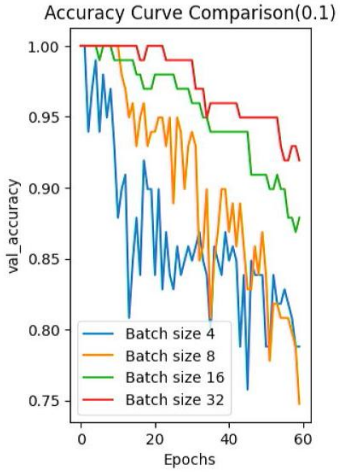
This model clearly suffered from overfitting as the models trained on each batch size had their best validation accuracy and loss result way before the final epoch, indicating that the model was tailoring its training to improve the results on the training dataset. On my second attempt with these settings, I shortened the epochs to 10, but unfortunetly, the result was entirely different than the previous attempt, and now changing the best result to around the 6-7th epoch.

```
Epoch 2/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9680 - loss: 0.1935 - val_accuracy: 1.0000 - val_loss: 0.0572
Epoch 3/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - accuracy: 0.9773 - loss: 0.0840 - val_accuracy: 0.9394 - val_loss: 0.2138
Epoch 4/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9849 - loss: 0.1107 - val_accuracy: 0.9697 - val_loss: 0.1384
Epoch 5/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9699 - loss: 0.0018 - val_accuracy: 0.9899 - val_loss: 0.0702
Epoch 6/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9772 - loss: 0.1197 - val_accuracy: 0.9394 - val_loss: 0.1682
Epoch 7/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9675 - loss: -0.0218 - val_accuracy: 0.9798 - val_loss: 0.0870
Epoch 8/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9852 - loss: -0.0212 - val_accuracy: 0.9495 - val_loss: 0.1519
Epoch 60/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - accuracy: 0.9642 - loss: -0.4465 - val_accuracy: 0.7879 - val_loss: 1.5680
Training with batch size: 8
Epoch 1/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 2s 36ms/step - accuracy: 0.8375 - loss: 0.4079 - val_accuracy: 1.0000 - val_loss: 0.1120
Epoch 2/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9890 - loss: 0.1582 - val_accuracy: 1.0000 - val_loss: 0.0941
Epoch 3/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9693 - loss: 0.1768 - val_accuracy: 1.0000 - val_loss: 0.0651
Epoch 4/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9731 - loss: 0.1458 - val_accuracy: 1.0000 - val_loss: 0.0569
Epoch 5/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9863 - loss: 0.0813 - val_accuracy: 1.0000 - val_loss: 0.0841
Epoch 6/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9691 - loss: 0.1512 - val_accuracy: 1.0000 - val_loss: 0.0728
Epoch 7/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9775 - loss: 0.0731 - val_accuracy: 1.0000 - val_loss: 0.0778
Epoch 8/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9806 - loss: 0.0766 - val_accuracy: 1.0000 - val_loss: 0.0705
Epoch 9/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9670 - loss: 0.0223 - val_accuracy: 1.0000 - val_loss: 0.0673
Epoch 10/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 0.9696 - loss: -0.0072 - val_accuracy: 1.0000 - val_loss: 0.0616
Epoch 11/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 20ms/step - accuracy: 0.9719 - loss: -0.0256 - val_accuracy: 1.0000 - val_loss: 0.0582
Epoch 12/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9696 - loss: 0.0312 - val_accuracy: 0.9798 - val_loss: 0.0765
```

```
Epoch 24/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9579 - loss: -0.0159 - val_accuracy: 0.9293 - val_loss: 0.1715
Epoch 25/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9496 - loss: 0.0517 - val_accuracy: 0.9495 - val_loss: 0.1646
Epoch 26/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 25ms/step - accuracy: 0.9600 - loss: 0.0209 - val_accuracy: 0.8889 - val_loss: 0.2117
Epoch 27/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.9434 - loss: -0.3509 - val_accuracy: 0.9495 - val_loss: 0.1440
Epoch 28/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9793 - loss: -0.0243 - val_accuracy: 0.9394 - val_loss: 0.1700
Epoch 29/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9650 - loss: -0.1207 - val_accuracy: 0.8990 - val_loss: 0.2179
Epoch 30/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9524 - loss: -0.2466 - val_accuracy: 0.9293 - val_loss: 0.1854
Epoch 31/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.9198 - loss: -0.3673 - val_accuracy: 0.9394 - val_loss: 0.1582
Epoch 32/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9491 - loss: -0.6221 - val_accuracy: 0.9293 - val_loss: 0.1623
Epoch 33/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 20ms/step - accuracy: 0.9641 - loss: 0.0365 - val_accuracy: 0.8485 - val_loss: 0.3134
Epoch 34/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.9406 - loss: -0.1174 - val_accuracy: 0.8586 - val_loss: 0.2769
Epoch 35/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step - accuracy: 0.9220 - loss: -0.0034 - val_accuracy: 0.8990 - val_loss: 0.2568
Epoch 36/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9778 - loss: -0.0045 - val_accuracy: 0.8081 - val_loss: 0.4482
Epoch 37/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9402 - loss: -0.0297 - val_accuracy: 0.8485 - val_loss: 0.3244
Epoch 38/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.9558 - loss: -0.5803 - val_accuracy: 0.8687 - val_loss: 0.2857
Epoch 39/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.9459 - loss: -1.1225 - val_accuracy: 0.8990 - val_loss: 0.2469
Epoch 60/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9248 - loss: -0.0430 - val_accuracy: 0.7475 - val_loss: 0.8654
Training with batch size: 16
Epoch 1/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 2s 70ms/step - accuracy: 0.8442 - loss: 0.4098 - val_accuracy: 1.0000 - val_loss: 0.1434
Epoch 2/60
Epoch 10/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9909 - loss: 0.0284 - val_accuracy: 0.9899 - val_loss: 0.0959
Epoch 11/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step - accuracy: 0.9603 - loss: -0.1347 - val_accuracy: 0.9899 - val_loss: 0.0847
Epoch 12/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 0.9885 - loss: -0.0058 - val_accuracy: 0.9899 - val_loss: 0.0900
Epoch 13/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step - accuracy: 0.9636 - loss: -0.1095 - val_accuracy: 0.9899 - val_loss: 0.0830
Epoch 14/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 28ms/step - accuracy: 0.9789 - loss: -0.0414 - val_accuracy: 0.9899 - val_loss: 0.0936
Epoch 15/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 49ms/step - accuracy: 0.9800 - loss: -0.0525 - val_accuracy: 0.9899 - val_loss: 0.0994
Epoch 16/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - accuracy: 0.9761 - loss: -0.1286 - val_accuracy: 0.9798 - val_loss: 0.1091
Epoch 17/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 37ms/step - accuracy: 0.9704 - loss: -0.0919 - val_accuracy: 0.9798 - val_loss: 0.1042
Epoch 18/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step - accuracy: 0.9867 - loss: -0.0429 - val_accuracy: 0.9697 - val_loss: 0.1220
Epoch 19/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 0.9762 - loss: -0.1961 - val_accuracy: 0.9697 - val_loss: 0.1039
Epoch 20/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step - accuracy: 0.9589 - loss: -0.1289 - val_accuracy: 0.9697 - val_loss: 0.0975
Epoch 21/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 0.9642 - loss: -0.1390 - val_accuracy: 0.9798 - val_loss: 0.0761
Epoch 22/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 0.9770 - loss: -0.1809 - val_accuracy: 0.9798 - val_loss: 0.0842
Epoch 23/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9628 - loss: -0.3280 - val_accuracy: 0.9798 - val_loss: 0.0860
Epoch 24/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step - accuracy: 0.9678 - loss: -0.4018 - val_accuracy: 0.9798 - val_loss: 0.0933
Epoch 60/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9570 - loss: -0.4919 - val_accuracy: 0.8788 - val_loss: 0.3518
Training with batch size: 32
Epoch 1/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 2s 142ms/step - accuracy: 0.6511 - loss: 0.6224 - val_accuracy: 1.0000 - val_loss: 0.1798
```

```
Epoch 60/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9570 - loss: -0.4919 - val_accuracy: 0.8788 - val_loss: 0.3518
Training with batch size: 32
Epoch 1/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 2s 142ms/step - accuracy: 0.6511 - loss: 0.6224 - val_accuracy: 1.0000 - val_loss: 0.1798
Epoch 19/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 41ms/step - accuracy: 0.9778 - loss: -0.0035 - val_accuracy: 1.0000 - val_loss: 0.1131
Epoch 20/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 41ms/step - accuracy: 0.9707 - loss: 0.0046 - val_accuracy: 1.0000 - val_loss: 0.0967
Epoch 21/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step - accuracy: 0.9806 - loss: 0.0468 - val_accuracy: 1.0000 - val_loss: 0.0929
Epoch 22/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step - accuracy: 0.9836 - loss: 0.0308 - val_accuracy: 1.0000 - val_loss: 0.0907
Epoch 23/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step - accuracy: 0.9729 - loss: -0.0686 - val_accuracy: 1.0000 - val_loss: 0.0888
Epoch 24/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 45ms/step - accuracy: 0.9780 - loss: -0.0081 - val_accuracy: 0.9899 - val_loss: 0.0878
Epoch 25/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 42ms/step - accuracy: 0.9861 - loss: -0.0114 - val_accuracy: 0.9899 - val_loss: 0.0877
Epoch 26/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 42ms/step - accuracy: 0.9751 - loss: -0.0916 - val_accuracy: 0.9899 - val_loss: 0.0954
Epoch 27/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 42ms/step - accuracy: 0.9787 - loss: -0.0241 - val_accuracy: 0.9899 - val_loss: 0.0918
Epoch 28/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - accuracy: 0.9790 - loss: 0.0166 - val_accuracy: 0.9899 - val_loss: 0.0928
Epoch 29/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step - accuracy: 0.9829 - loss: -0.0414 - val_accuracy: 0.9899 - val_loss: 0.0925
Epoch 30/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 46ms/step - accuracy: 0.9725 - loss: -0.0897 - val_accuracy: 0.9899 - val_loss: 0.0869
Epoch 31/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 43ms/step - accuracy: 0.9759 - loss: 0.0106 - val_accuracy: 0.9899 - val_loss: 0.0882
Epoch 32/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - accuracy: 0.9725 - loss: -0.1162 - val_accuracy: 0.9697 - val_loss: 0.1196
Epoch 60/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 42ms/step - accuracy: 0.9667 - loss: -0.0842 - val_accuracy: 0.9192 - val_loss: 0.2615
```



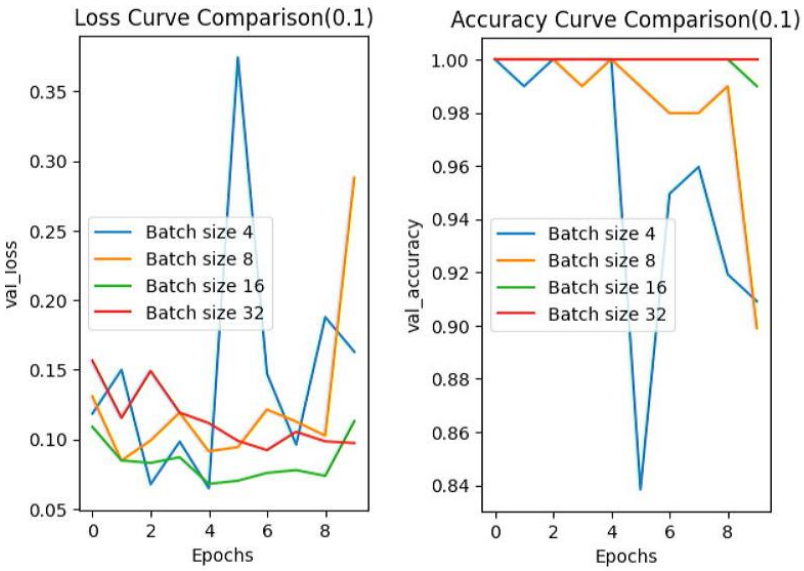


Squat Jump Model Test 4-B

Same settings as test 4 with the epochs set to 10 instead of 60

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `inp
ut_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in th
e model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 3s 24ms/step - accuracy: 0.9515 - loss: 0.2918 - val_accuracy: 1.0000 - val_loss: 0.1184
Epoch 2/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 23ms/step - accuracy: 0.9782 - loss: 0.1943 - val_accuracy: 0.9899 - val_loss: 0.1500
Epoch 3/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 22ms/step - accuracy: 0.9772 - loss: 0.1430 - val_accuracy: 1.0000 - val_loss: 0.0676
Epoch 4/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9658 - loss: 0.0805 - val_accuracy: 1.0000 - val_loss: 0.0983
Epoch 5/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9497 - loss: 0.0261 - val_accuracy: 1.0000 - val_loss: 0.0647
Epoch 6/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9783 - loss: 0.0433 - val_accuracy: 0.8384 - val_loss: 0.3745
Epoch 7/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 11ms/step - accuracy: 0.9547 - loss: 0.0426 - val_accuracy: 0.9495 - val_loss: 0.1471
Epoch 8/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9691 - loss: -0.1964 - val_accuracy: 0.9596 - val_loss: 0.0961
Epoch 9/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9803 - loss: -0.0015 - val_accuracy: 0.9192 - val_loss: 0.1878
Epoch 10/10
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9524 - loss: -0.0521 - val_accuracy: 0.9091 - val_loss: 0.1629
Training with batch size: 8
Epoch 1/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 2s 34ms/step - accuracy: 0.6471 - loss: 0.8826 - val_accuracy: 1.0000 - val_loss: 0.1308
Epoch 2/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step - accuracy: 0.9629 - loss: 0.2929 - val_accuracy: 1.0000 - val_loss: 0.0846
Epoch 3/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step - accuracy: 0.9757 - loss: 0.1614 - val_accuracy: 1.0000 - val_loss: 0.0990
Epoch 4/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.9919 - loss: 0.0937 - val_accuracy: 0.9899 - val_loss: 0.1191
Epoch 5/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 32ms/step - accuracy: 0.9707 - loss: 0.1816 - val_accuracy: 1.0000 - val_loss: 0.0913
Epoch 6/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - accuracy: 0.9850 - loss: 0.0490 - val_accuracy: 0.9899 - val_loss: 0.0943
Epoch 7/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.9864 - loss: 0.0065 - val_accuracy: 0.9798 - val_loss: 0.1214
Epoch 8/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.9816 - loss: 0.0020 - val_accuracy: 0.9798 - val_loss: 0.1127
Epoch 9/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.9808 - loss: -0.0873 - val_accuracy: 0.9899 - val_loss: 0.1026
Epoch 10/10
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9748 - loss: -6.8350e-04 - val_accuracy: 0.8990 - val_loss: 0.
Training with batch size: 16
Epoch 1/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 2s 63ms/step - accuracy: 0.5588 - loss: 0.6363 - val_accuracy: 1.0000 - val_loss: 0.1089
Epoch 2/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 0.9777 - loss: 0.2390 - val_accuracy: 1.0000 - val_loss: 0.0847
Epoch 3/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 30ms/step - accuracy: 0.9766 - loss: 0.1636 - val_accuracy: 1.0000 - val_loss: 0.0829
Epoch 4/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - accuracy: 0.9814 - loss: 0.1587 - val_accuracy: 1.0000 - val_loss: 0.0871
Epoch 5/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step - accuracy: 0.9596 - loss: 0.1694 - val_accuracy: 1.0000 - val_loss: 0.0679
Epoch 6/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9807 - loss: 0.1181 - val_accuracy: 1.0000 - val_loss: 0.0702
Epoch 7/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step - accuracy: 0.9803 - loss: 0.1167 - val_accuracy: 1.0000 - val_loss: 0.0758
Epoch 8/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - accuracy: 0.9645 - loss: 0.1338 - val_accuracy: 1.0000 - val_loss: 0.0778
Epoch 9/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step - accuracy: 0.9754 - loss: 0.0747 - val_accuracy: 1.0000 - val_loss: 0.0737
```

```
Epoch 7/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step - accuracy: 0.9803 - loss: 0.1167 - val_accuracy: 1.0000 - val_loss: 0.0758
Epoch 8/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - accuracy: 0.9645 - loss: 0.1338 - val_accuracy: 1.0000 - val_loss: 0.0778
Epoch 9/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step - accuracy: 0.9754 - loss: 0.0747 - val_accuracy: 1.0000 - val_loss: 0.0737
Epoch 10/10
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9911 - loss: 0.0745 - val_accuracy: 0.9899 - val_loss: 0.1130
Training with batch size: 32
Epoch 1/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 2s 114ms/step - accuracy: 0.9165 - loss: 0.4054 - val_accuracy: 1.0000 - val_loss: 0.1566
Epoch 2/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - accuracy: 0.9686 - loss: 0.2911 - val_accuracy: 1.0000 - val_loss: 0.1154
Epoch 3/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step - accuracy: 0.9814 - loss: 0.2108 - val_accuracy: 1.0000 - val_loss: 0.1491
Epoch 4/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 42ms/step - accuracy: 0.9810 - loss: 0.2024 - val_accuracy: 1.0000 - val_loss: 0.1191
Epoch 5/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 112ms/step - accuracy: 0.9740 - loss: 0.2258 - val_accuracy: 1.0000 - val_loss: 0.1116
Epoch 6/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 45ms/step - accuracy: 0.9709 - loss: 0.2234 - val_accuracy: 1.0000 - val_loss: 0.0988
Epoch 7/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 48ms/step - accuracy: 0.9728 - loss: 0.1918 - val_accuracy: 1.0000 - val_loss: 0.0921
Epoch 8/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 38ms/step - accuracy: 0.9808 - loss: 0.1571 - val_accuracy: 1.0000 - val_loss: 0.1053
Epoch 9/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 44ms/step - accuracy: 0.9709 - loss: 0.1857 - val_accuracy: 1.0000 - val_loss: 0.0985
Epoch 10/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 56ms/step - accuracy: 0.9832 - loss: 0.1351 - val_accuracy: 1.0000 - val_loss: 0.0972
```



Squat Jump Model Test 5

Layers:

1. Sigmoid (40)
2. Sigmoid (1)

Epochs: 60 (Just to see where the accuracy and loss plateaus)

Learning rate: 0.01

Observations:

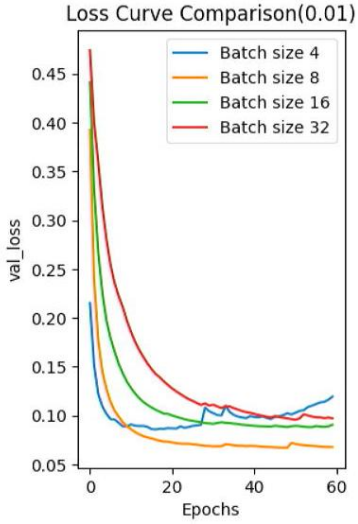
These settings provided very solid results in which the batch size of 8 came out on top in both the validation loss and accuracy. We can also see that the results have plateaued at the 40 epoch mark. My model had a final accuracy of 78%, which is not ideal, but just like the model in "Test 6", it falsely identified 19 samples as "negative" showing that a proper squat jump exercise could be more difficult to detect.

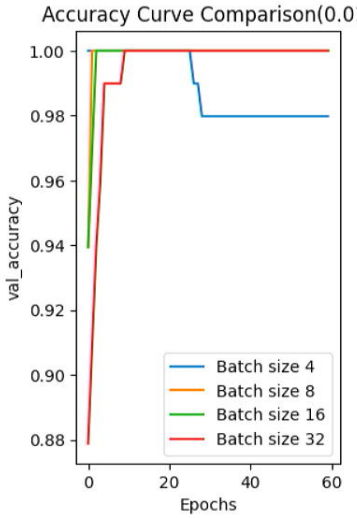
The objective is to minimize the number of false positives, allowing only those completed with proper exercise fidelity to pass through

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `in
put_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in
the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 3s 21ms/step - accuracy: 0.9032 - loss: 0.4222 - val_accuracy: 1.0000 - val_loss: 0.2151
```

```
Epoch 7/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9913 - loss: 0.1185 - val_accuracy: 1.0000 - val_loss: 0.0958
Epoch 8/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - accuracy: 0.9707 - loss: 0.1754 - val_accuracy: 1.0000 - val_loss: 0.0923
Epoch 9/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9748 - loss: 0.1984 - val_accuracy: 1.0000 - val_loss: 0.0888
Epoch 10/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9771 - loss: 0.1817 - val_accuracy: 1.0000 - val_loss: 0.0890
Epoch 11/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9885 - loss: 0.1400 - val_accuracy: 1.0000 - val_loss: 0.0908
Epoch 12/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9829 - loss: 0.1571 - val_accuracy: 1.0000 - val_loss: 0.0895
Epoch 13/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9783 - loss: 0.1487 - val_accuracy: 1.0000 - val_loss: 0.0892
Epoch 14/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9832 - loss: 0.1137 - val_accuracy: 1.0000 - val_loss: 0.0892
Epoch 15/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9757 - loss: 0.1006 - val_accuracy: 1.0000 - val_loss: 0.0882
Epoch 16/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.9585 - loss: 0.1637 - val_accuracy: 1.0000 - val_loss: 0.0860
Epoch 17/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9798 - loss: 0.1240 - val_accuracy: 1.0000 - val_loss: 0.0857
Epoch 18/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9839 - loss: 0.0756 - val_accuracy: 1.0000 - val_loss: 0.0864
Epoch 19/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9613 - loss: 0.2103 - val_accuracy: 1.0000 - val_loss: 0.0863
Epoch 20/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 15ms/step - accuracy: 0.9831 - loss: 0.1062 - val_accuracy: 1.0000 - val_loss: 0.0870
Epoch 21/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 17ms/step - accuracy: 0.9809 - loss: 0.0938 - val_accuracy: 1.0000 - val_loss: 0.0868
Epoch 22/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9771 - loss: 0.1294 - val_accuracy: 1.0000 - val_loss: 0.0866
Epoch 23/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9641 - loss: 0.1633 - val_accuracy: 1.0000 - val_loss: 0.0887
Epoch 60/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 14ms/step - accuracy: 0.9752 - loss: 0.0204 - val_accuracy: 0.9798 - val_loss: 0.1193
Training with batch size: 8
Epoch 1/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 3s 36ms/step - accuracy: 0.4566 - loss: 0.7105 - val_accuracy: 0.9394 - val_loss: 0.3925
Epoch 45/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.9784 - loss: 0.0179 - val_accuracy: 1.0000 - val_loss: 0.0677
Epoch 46/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9679 - loss: -0.0638 - val_accuracy: 1.0000 - val_loss: 0.0674
Epoch 47/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9833 - loss: -0.0485 - val_accuracy: 1.0000 - val_loss: 0.0672
Epoch 48/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9780 - loss: -0.1024 - val_accuracy: 1.0000 - val_loss: 0.0670
Epoch 49/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9823 - loss: -0.0476 - val_accuracy: 1.0000 - val_loss: 0.0669
Epoch 50/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 19ms/step - accuracy: 0.9889 - loss: -0.0052 - val_accuracy: 1.0000 - val_loss: 0.0718
Epoch 60/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.9866 - loss: -0.0463 - val_accuracy: 1.0000 - val_loss: 0.0676
Training with batch size: 16
Epoch 1/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 2s 69ms/step - accuracy: 0.8448 - loss: 0.5930 - val_accuracy: 0.9394 - val_loss: 0.4412
Epoch 55/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9913 - loss: 0.0925 - val_accuracy: 1.0000 - val_loss: 0.0881
Epoch 56/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - accuracy: 0.9909 - loss: 0.0813 - val_accuracy: 1.0000 - val_loss: 0.0891
Epoch 57/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9649 - loss: 0.1087 - val_accuracy: 1.0000 - val_loss: 0.0886
Epoch 58/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - accuracy: 0.9864 - loss: 0.0809 - val_accuracy: 1.0000 - val_loss: 0.0883
Epoch 59/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9861 - loss: 0.1042 - val_accuracy: 1.0000 - val_loss: 0.0889
Epoch 60/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step - accuracy: 0.9879 - loss: 0.0931 - val_accuracy: 1.0000 - val_loss: 0.0904
Training with batch size: 32
Epoch 1/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 2s 143ms/step - accuracy: 0.8262 - loss: 0.5289 - val_accuracy: 0.8788 - val_loss: 0.4740
```

```
Epoch 57/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 41ms/step - accuracy: 0.9843 - loss: 0.1166 - val_accuracy: 1.0000 - val_loss: 0.0980
Epoch 58/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 41ms/step - accuracy: 0.9730 - loss: 0.1455 - val_accuracy: 1.0000 - val_loss: 0.0972
Epoch 59/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 42ms/step - accuracy: 0.9829 - loss: 0.1071 - val_accuracy: 1.0000 - val_loss: 0.0979
Epoch 60/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 1s 41ms/step - accuracy: 0.9823 - loss: 0.1214 - val_accuracy: 1.0000 - val_loss: 0.0970
```






```
confusion matrix:
[[47  3]
 [19 30]]
```

Confusion Matrix Legend

Actual sample value	Predicted Class: Negative	Predicted Class: Positive
Negative	TN	FP
Positive	FN	TP

```
classification_report:
              precision    recall  f1-score   support

           0       0.71      0.94      0.81        50
           1       0.91      0.61      0.73        49

    accuracy                           0.78        99
   macro avg       0.81      0.78      0.77        99
weighted avg       0.81      0.78      0.77        99
```

Squat Jump Model Test 6

Layers:

1. Sigmoid (40)
2. Sigmoid (1)

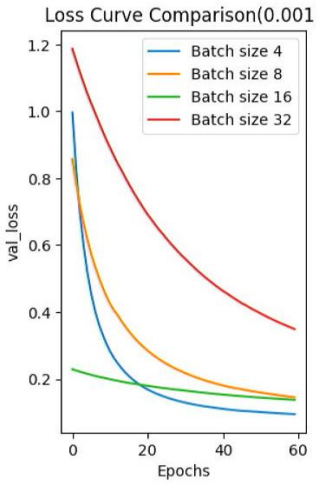
Epochs: 60 (Just to see where the accuracy and loss plateaus)

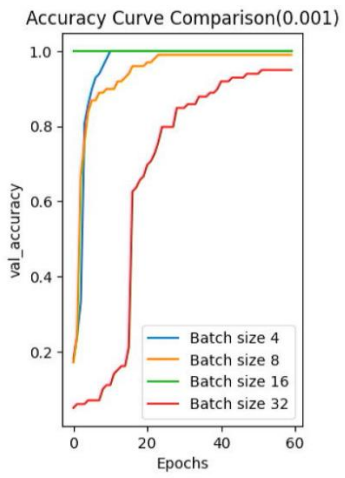
Learning rate: 0.001

Observations:

This model did not preform as well as the 0.01 learning rate model, however it shows consistent progress towards the 100% target accuracy. One weakness with this model is that it identified 29 false negatives, which was the ultimate reason for my decision to go with the "Test 5" model instead.

```
Training with batch size: 4
/home/platobearodog/CYSF_Project/myenv/lib/python3.11/site-packages/keras/src/layers/core/dense.py:87: UserWarning: Do not pass an `in
put_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in
the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
Epoch 1/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 3s 21ms/step - accuracy: 0.1192 - loss: 1.1347 - val_accuracy: 0.1818 - val_loss: 0.9963
Epoch 60/60
51/51 ━━━━━━━━━━━━━━━━━━━━ 1s 13ms/step - accuracy: 0.9582 - loss: 0.3324 - val_accuracy: 1.0000 - val_loss: 0.0950
Training with batch size: 8
Epoch 1/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 2s 35ms/step - accuracy: 0.1962 - loss: 0.8198 - val_accuracy: 0.1717 - val_loss: 0.8573
Epoch 60/60
26/26 ━━━━━━━━━━━━━━━━━━━━ 1s 18ms/step - accuracy: 0.9650 - loss: 0.2102 - val_accuracy: 0.9899 - val_loss: 0.1456
Training with batch size: 16
Epoch 1/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 3s 68ms/step - accuracy: 0.9711 - loss: 0.3487 - val_accuracy: 1.0000 - val_loss: 0.2290
Epoch 60/60
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step - accuracy: 0.9897 - loss: 0.1912 - val_accuracy: 1.0000 - val_loss: 0.1378
Training with batch size: 32
Epoch 1/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 2s 137ms/step - accuracy: 0.1132 - loss: 1.0838 - val_accuracy: 0.0505 - val_loss: 1.1866
Epoch 60/60
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 41ms/step - accuracy: 0.9545 - loss: 0.4115 - val_accuracy: 0.9495 - val_loss: 0.3493
```






```
confusion matrix:
[[48  2]
 [29 20]]
```

Confusion Matrix Legend

Actual sample value	Predicted Class: Negative	Predicted Class: Positive
Negative	TN	FP
Positive	FN	TP

```
classification_report:
              precision    recall  f1-score   support

           0       0.62      0.96      0.76        50
           1       0.91      0.41      0.56        49

    accuracy                           0.69        99
   macro avg       0.77      0.68      0.66        99
weighted avg       0.76      0.69      0.66        99
```

Final Settings of the Squat Jump Model:

Layer 1 = Sigmoid

Layer 2 = Sigmoid

Learning Rate = 0.01

Batch Size = 8

Epochs = 40

Classification Threshold = 0.35

Observations:

The model performed well at keeping out false positives but it was not as effective at keeping out false negatives. This was the reason why I adjusted the threshold in which predictions are set to true or false. In this case, the best overall threshold on both the training and validation datasets was 0.35. Overall, the model is not perfect, but it meets my objective with an approximate 82-86% accuracy on both the validation dataset, and the test dataset.

Confusion Matrix Legend

Actual sample value	Predicted Class: Negative	Predicted Class: Positive
Negative	TN	FP
Positive	FN	TP

<u>Validation Dataset Results:</u>

```
confusion matrix:
[[47  3]
 [19 30]]
```

```
classification_report:
              precision    recall  f1-score   support

           0       0.71      0.94      0.81        50
           1       0.91      0.61      0.73        49

    accuracy                           0.78        99
   macro avg       0.81      0.78      0.77        99
weighted avg       0.81      0.78      0.77        99
```

<u>When Classification Threshold =0.4</u>

```
confusion matrix:
[[45  5]
 [16 33]]
```

```
classification_report:
              precision    recall  f1-score   support

           0       0.74      0.90      0.81        50
           1       0.87      0.67      0.76        49

    accuracy                           0.79        99
   macro avg       0.80      0.79      0.78        99
weighted avg       0.80      0.79      0.78        99
```

Classification Threshold = 0.35 (Final Selected Model)

```
classification_report:
              precision    recall  f1-score   support

           0       0.78      0.90      0.83        50
           1       0.88      0.73      0.80        49

    accuracy                           0.82        99
   macro avg       0.83      0.82      0.82        99
weighted avg       0.83      0.82      0.82        99

confusion matrix:
[[45  5]
 [13 36]]
```

Classification Threshold = 0.3

```
classification_report:
              precision    recall  f1-score   support

           0       0.83      0.90      0.87        50
           1       0.89      0.82      0.85        49

    accuracy                           0.86        99
   macro avg       0.86      0.86      0.86        99
weighted avg       0.86      0.86      0.86        99

confusion matrix:
[[45  5]
 [ 9 40]]
```

Classification Threshold = 0.2

```
classification_report:
              precision    recall  f1-score   support

           0       0.85      0.80      0.82        50
           1       0.81      0.86      0.83        49

    accuracy                           0.83        99
   macro avg       0.83      0.83      0.83        99
weighted avg       0.83      0.83      0.83        99

[[40 10]
 [ 7 42]]
```

Test Dataset Results:

Classification Threshold = 0.4

```
classification_report:
              precision    recall  f1-score   support

           0       0.90      0.84      0.87        32
           1       0.85      0.91      0.88        32

    accuracy                           0.88        64
   macro avg       0.88      0.88      0.87        64
weighted avg       0.88      0.88      0.87        64
```

```
confusion matrix:
[[27  5]
 [ 3 29]]
```

Classification Threshold = 0.35 (Final selected model)

```
confusion matrix:
[[26  6]
 [ 2 30]]
```

```
classification_report:
              precision    recall  f1-score   support

           0       0.93      0.81      0.87        32
           1       0.83      0.94      0.88        32

    accuracy                           0.88        64
   macro avg       0.88      0.88      0.87        64
weighted avg       0.88      0.88      0.87        64
```

Classification Threshold = 0.3

```
confusion matrix:
[[25  7]
 [ 2 30]]
```

```
classification_report:
              precision    recall  f1-score   support

           0       0.93      0.78      0.85        32
           1       0.81      0.94      0.87        32

    accuracy                           0.86        64
   macro avg       0.87      0.86      0.86        64
weighted avg       0.87      0.86      0.86        64
```

Classification Threshold = 0.2

```
confusion matrix:
[[23  9]
 [ 0 32]]
```

```
classification_report:
              precision    recall  f1-score   support

           0       1.00      0.72      0.84        32
           1       0.78      1.00      0.88        32

    accuracy                           0.86        64
   macro avg       0.89      0.86      0.86        64
weighted avg       0.89      0.86      0.86        64
```