Neural Network for Detecting Cardiovascular Disease in patients by looking at some related results.

Furkan ÇOLAK

a. Search Internet and find a structured dataset for binary classification. Clearly identify input(s) and output(s).

The dataset for binary classification [Cardiovascular disease]

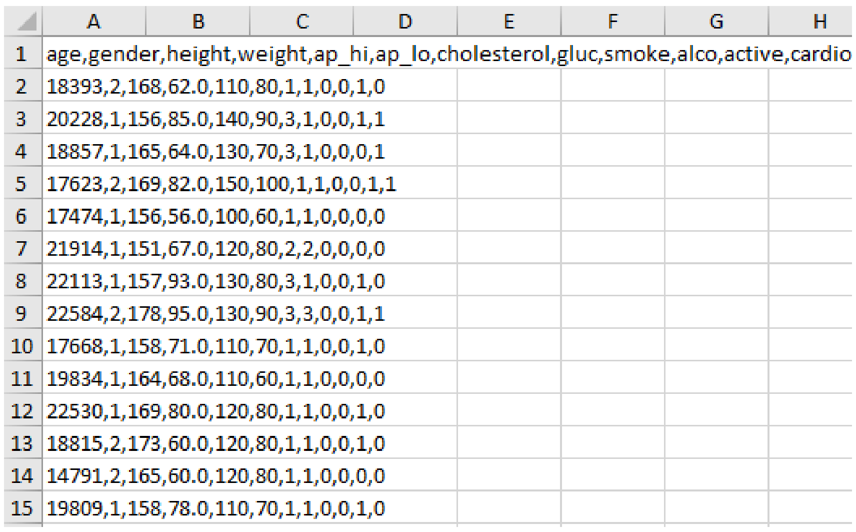
A-) There are 11 İnputs as a integer in dataset. Those are:

1. Age | Objective Feature | age | int (days)
2. Height | Objective Feature | height | int (cm) |
3. Weight | Objective Feature | weight | float (kg) |
4. Gender | Objective Feature | gender | categorical code |
5. Systolic blood pressure | Examination Feature | ap_hi | int |
6. Diastolic blood pressure | Examination Feature | ap_lo | int |
7. Cholesterol | Examination Feature | cholesterol | 1: normal, 2: above normal, 3: well above normal |
8. Glucose | Examination Feature | gluc | 1: normal, 2: above normal, 3: well above normal |
9. Smoking | Subjective Feature | smoke | binary |
10. Alcohol intake | Subjective Feature | alco | binary |
11. Physical activity | Subjective Feature | active | binary |

B_) Output is last column

12. Presence or absence of cardiovascular disease | Target Variable | cardio | binary |

Part of dataset is shown at the below.



	A	B	C	D	E	F	G	H
1	age,gender,height,weight,ap_hi,ap_lo,cholesterol,gluc,smoke,alco,active,cardio							
2	18393,2,168,62.0,110,80,1,1,0,0,1,0							
3	20228,1,156,85.0,140,90,3,1,0,0,1,1							
4	18857,1,165,64.0,130,70,3,1,0,0,0,1							
5	17623,2,169,82.0,150,100,1,1,0,0,1,1							
6	17474,1,156,56.0,100,60,1,1,0,0,0,0							
7	21914,1,151,67.0,120,80,2,2,0,0,0,0							
8	22113,1,157,93.0,130,80,3,1,0,0,1,0							
9	22584,2,178,95.0,130,90,3,3,0,0,1,1							
10	17668,1,158,71.0,110,70,1,1,0,0,1,0							
11	19834,1,164,68.0,110,60,1,1,0,0,0,0							
12	22530,1,169,80.0,120,80,1,1,0,0,1,0							
13	18815,2,173,60.0,120,80,1,1,0,0,1,0							
14	14791,2,165,60.0,120,80,1,1,0,0,0,0							
15	19809,1,158,78.0,110,70,1,1,0,0,1,0							

b. Use a 1-hidden layer NN and implement your own code in Python to train and test it (with adequate epoch number). Use Adam optimizer (beta_1=0.9, beta_2=0.999) and a fixed learning rate. Use at least three different number of neurons in the hidden layer and at least three batch sizes and compare the results by preparing confusion matrix and related metrics.

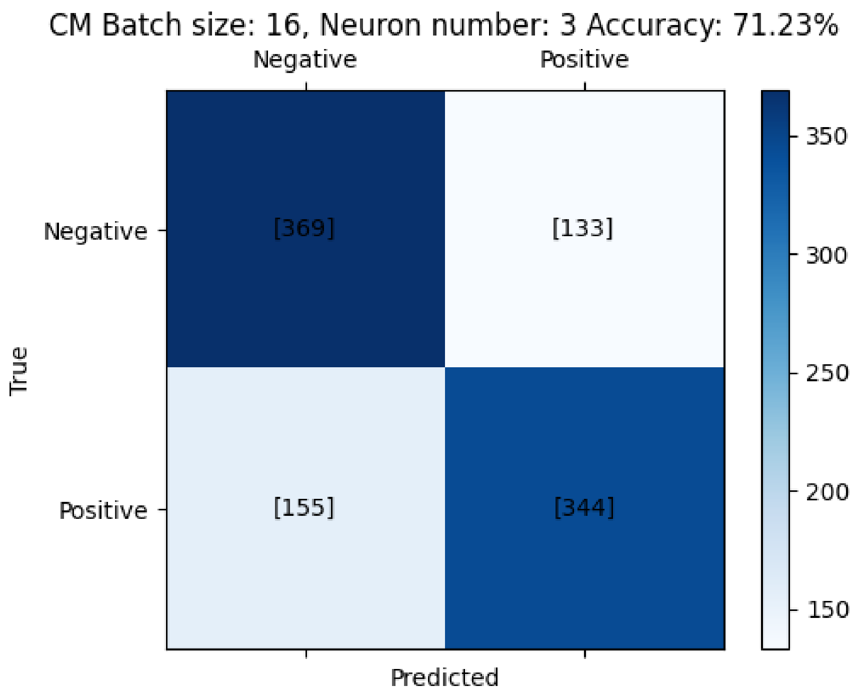
codes

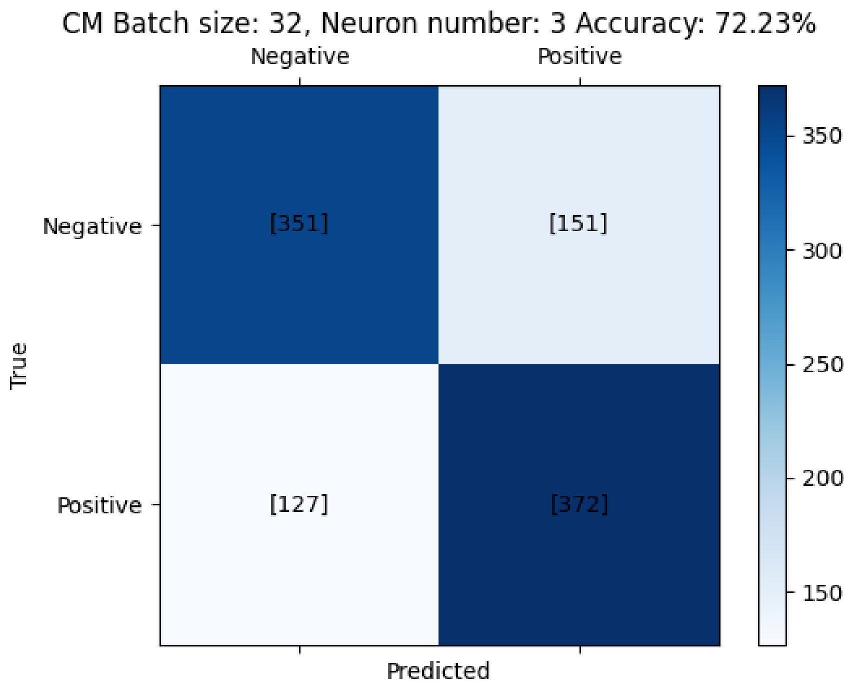
The batch sizes that I've chosen are [16,32,64,128]

The hidden neuron numbers that I've chosen are [3,5,7]

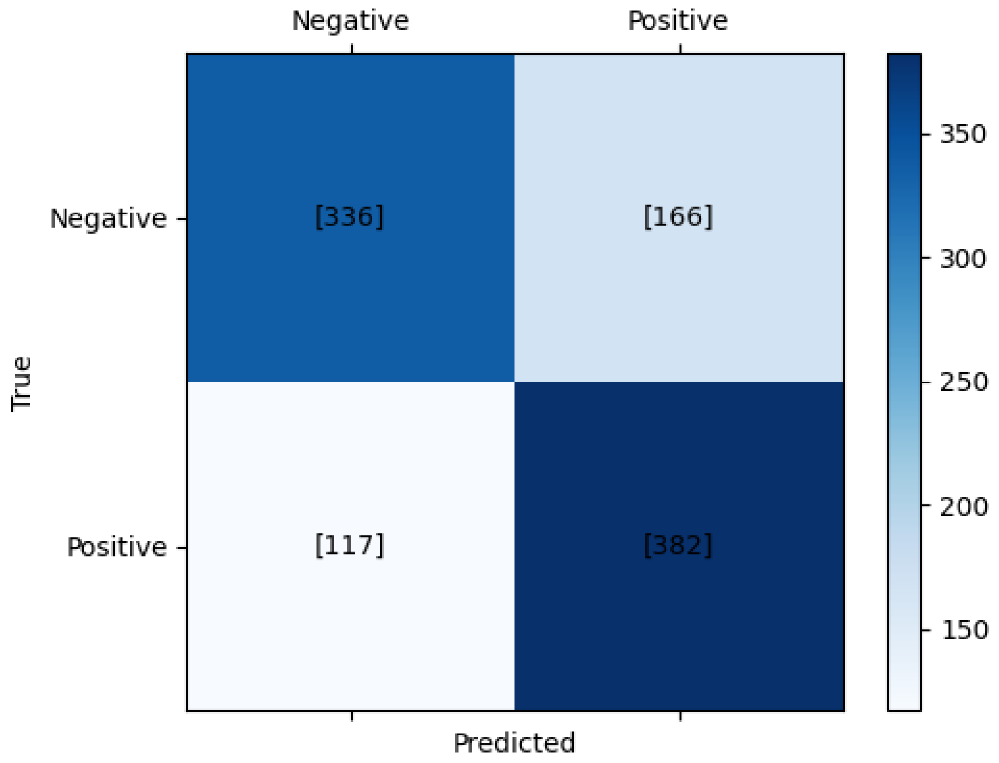
The fixed learning rate = 0.01

When hidden neuron number is 3

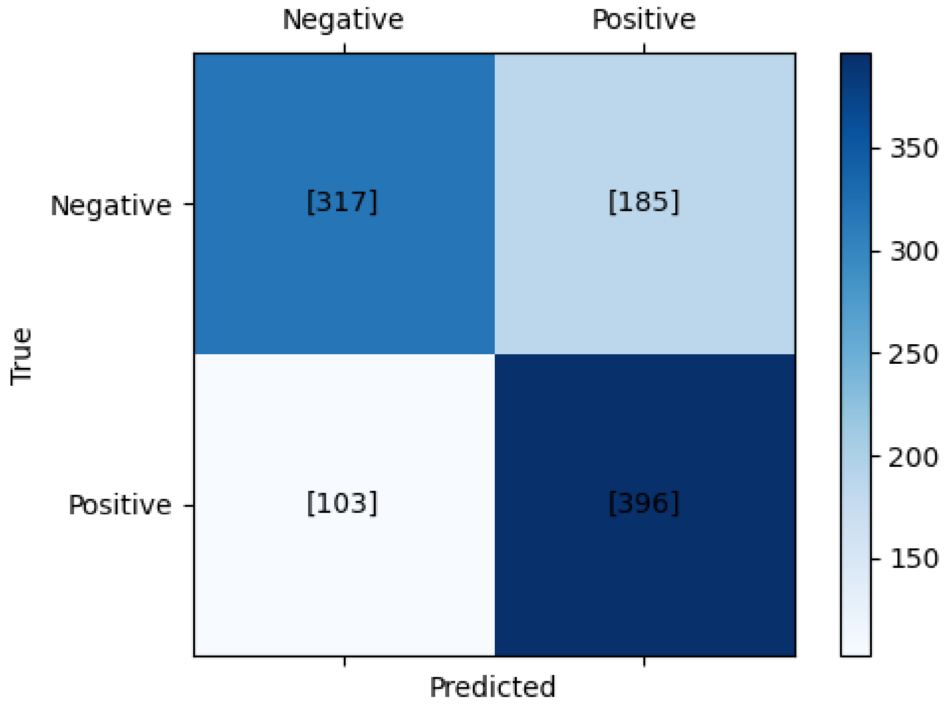




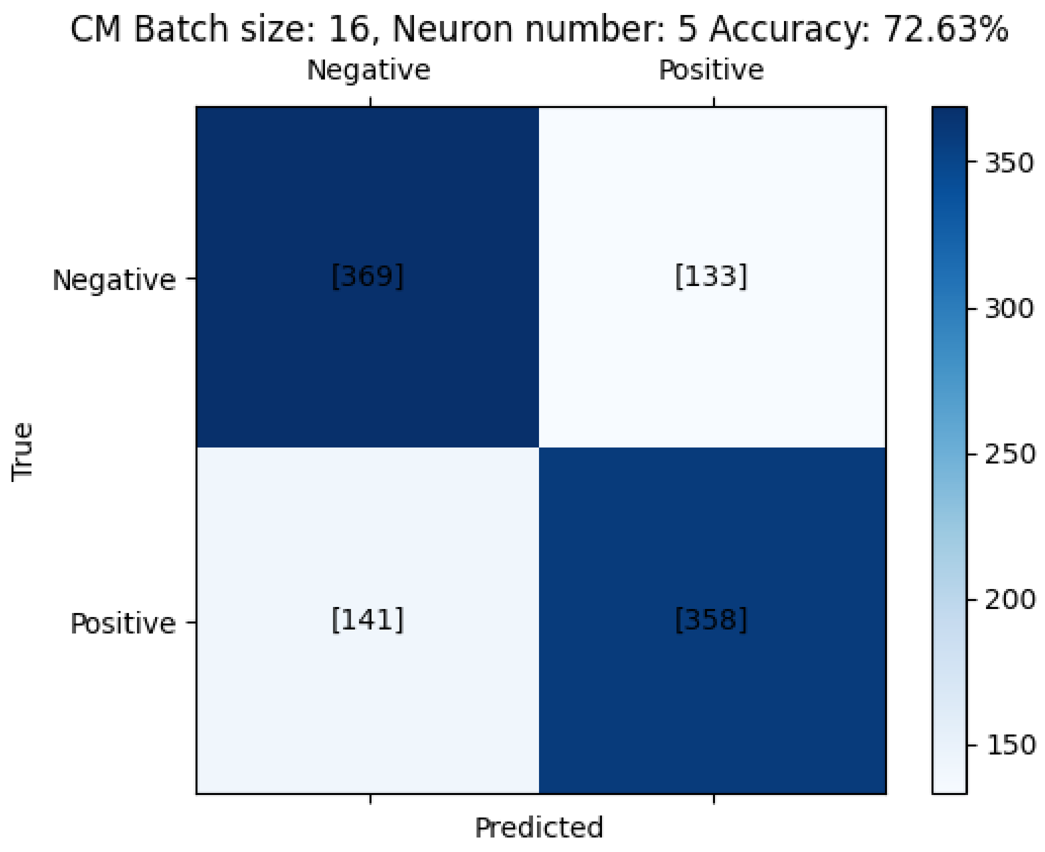
CM Batch size: 64, Neuron number: 3 Accuracy: 71.73%

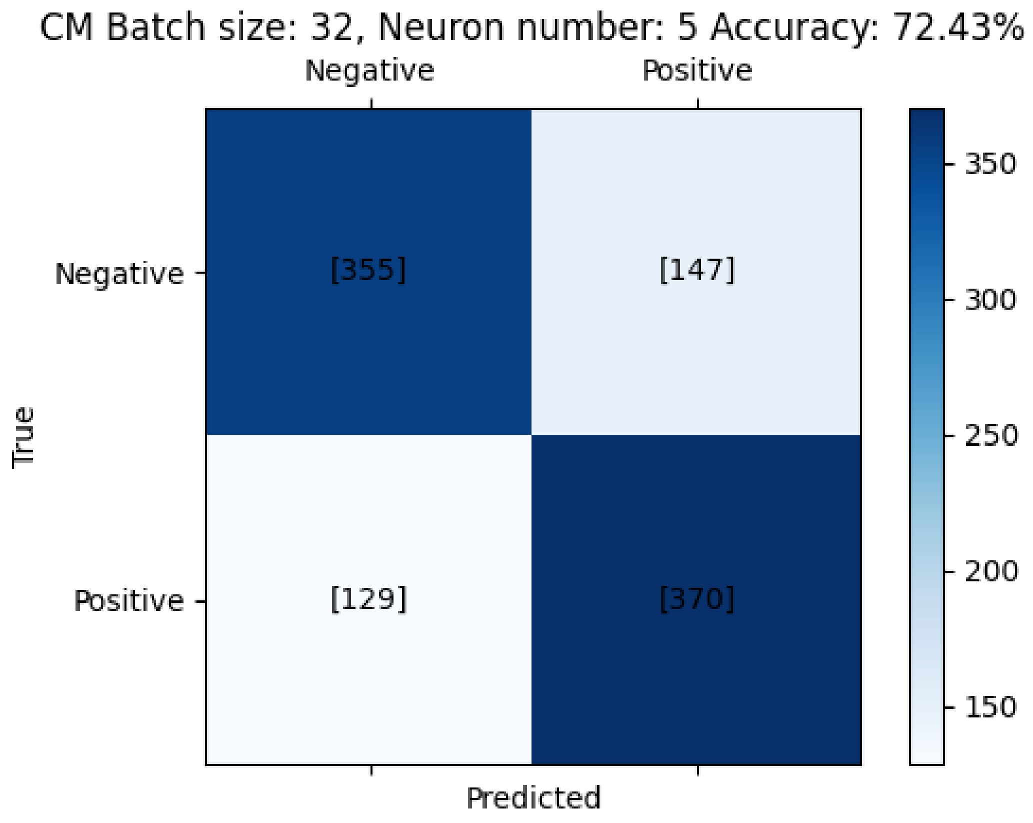


CM Batch size: 128, Neuron number: 3 Accuracy: 71.23%

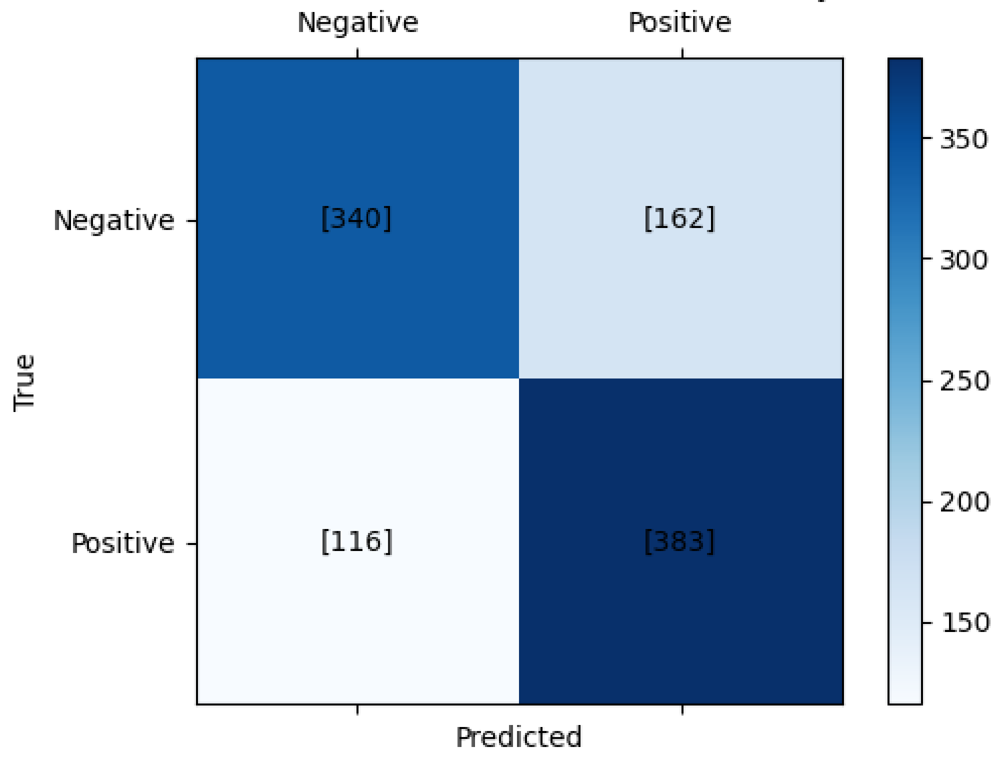


When hidden neuron number is 5

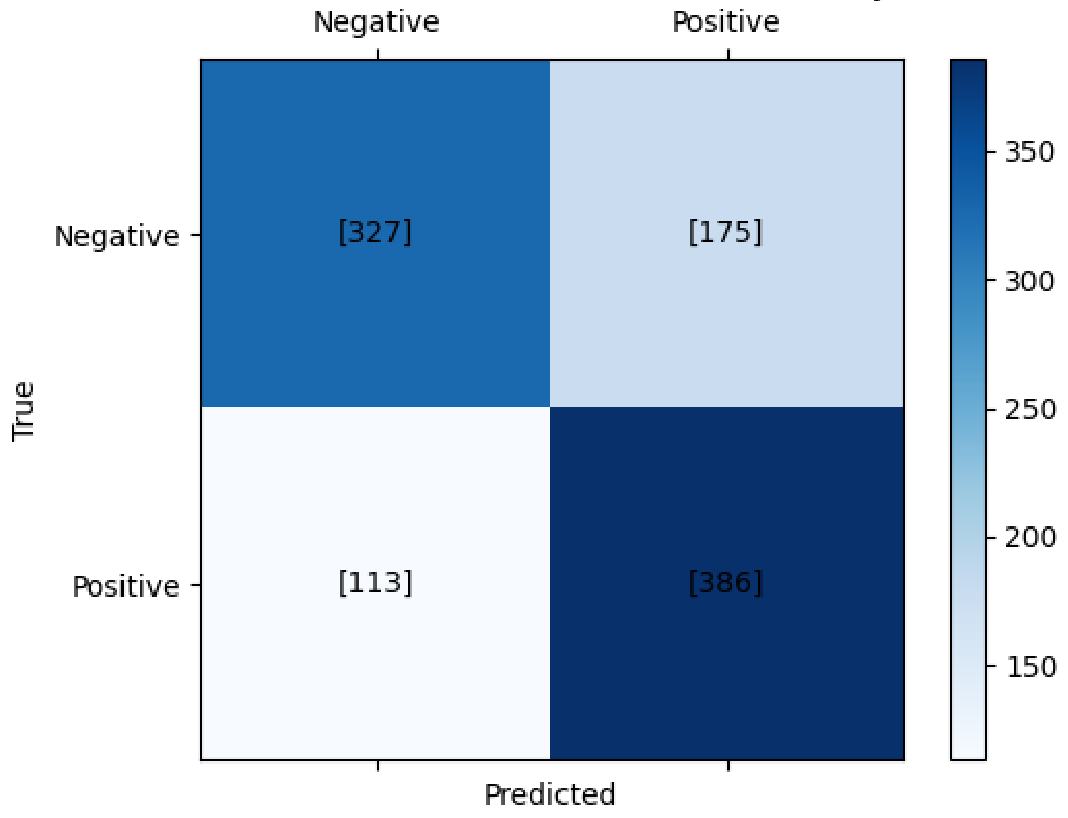




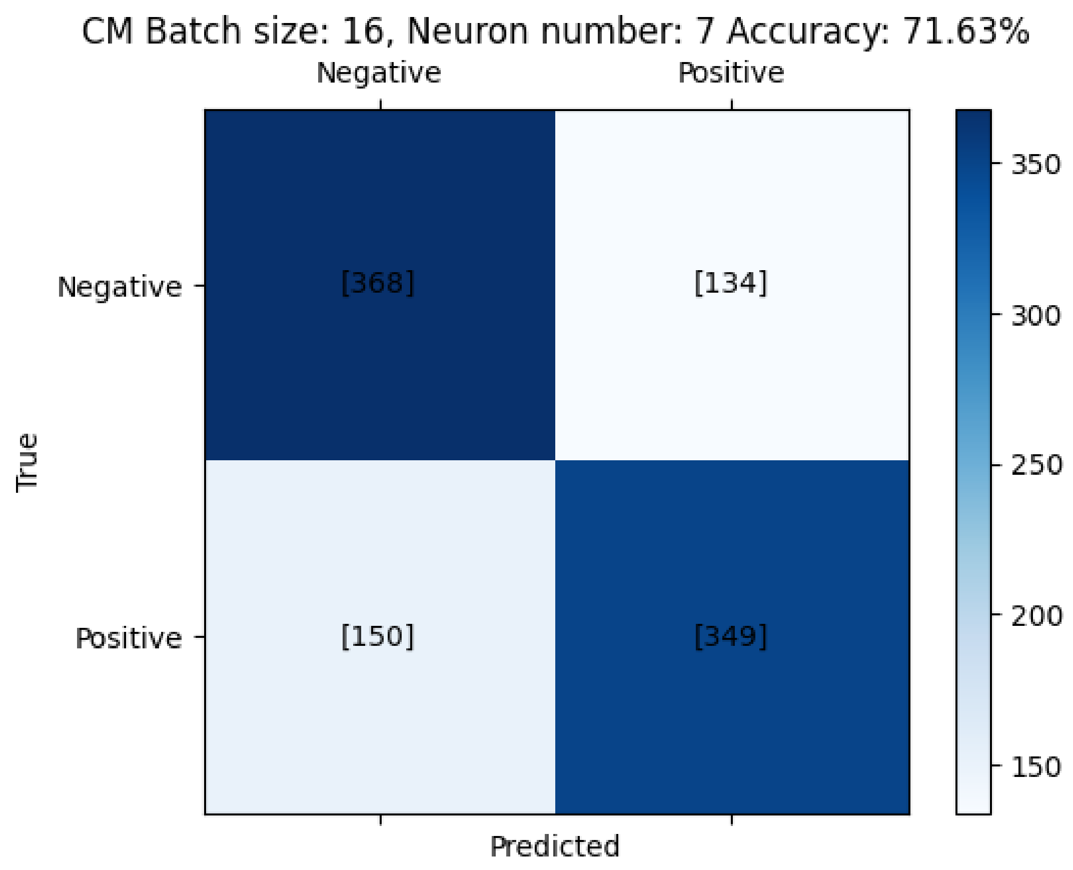
CM Batch size: 64, Neuron number: 5 Accuracy: 72.23%

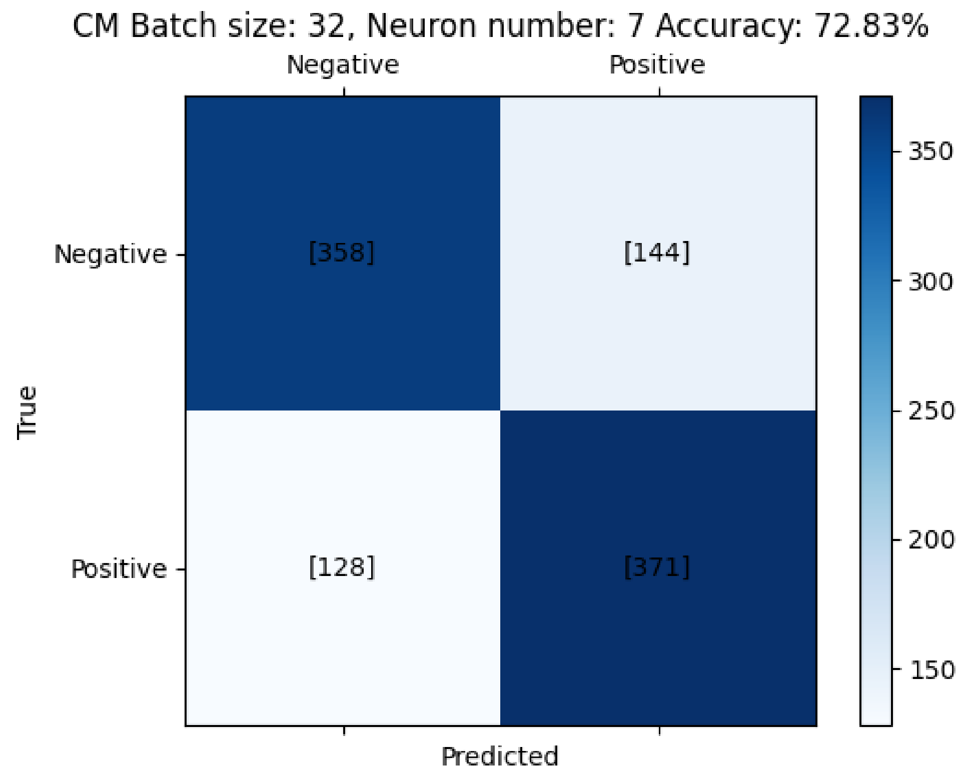


CM Batch size: 128, Neuron number: 5 Accuracy: 71.23%

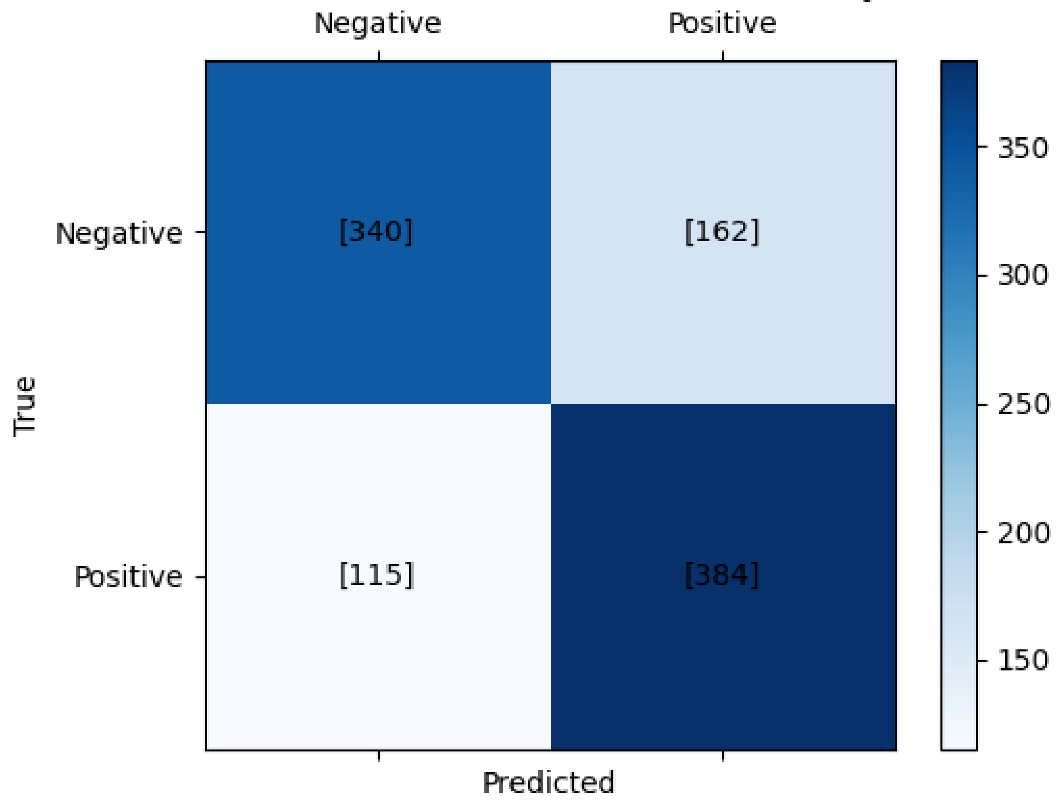


When hidden neuron number is 7

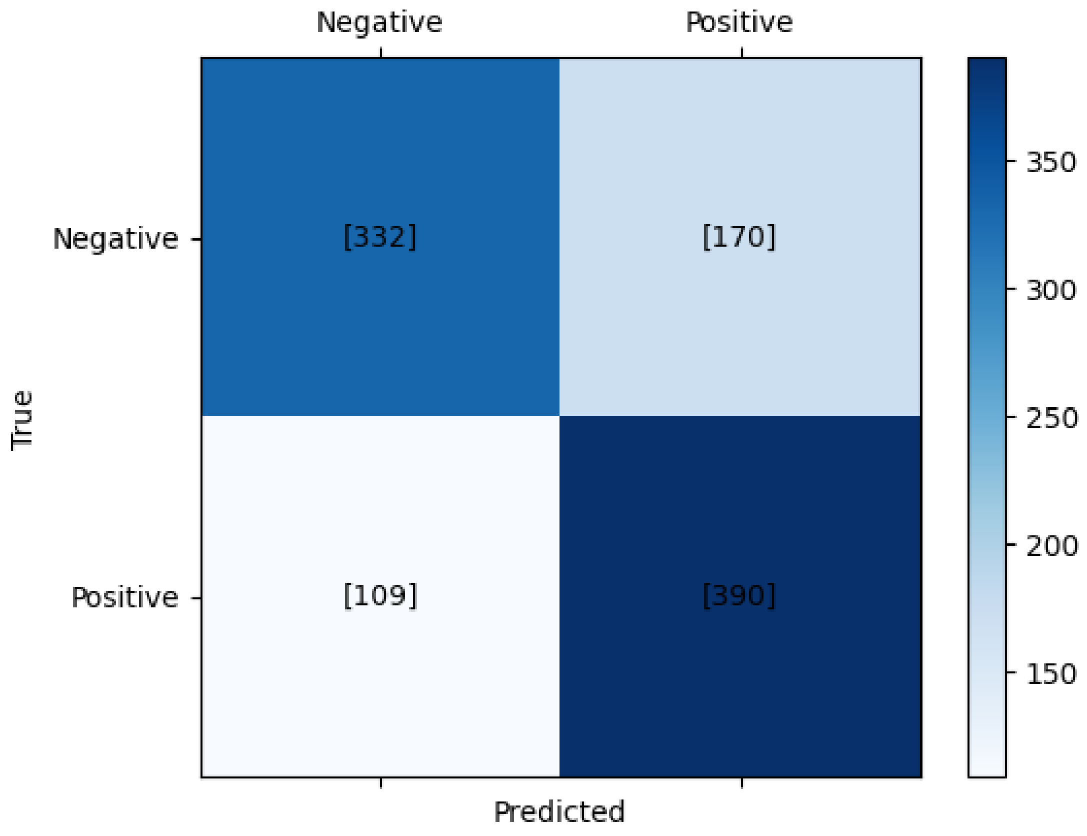




CM Batch size: 64, Neuron number: 7 Accuracy: 72.33%

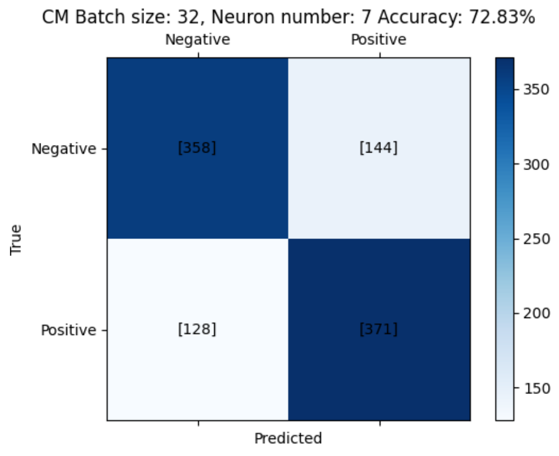


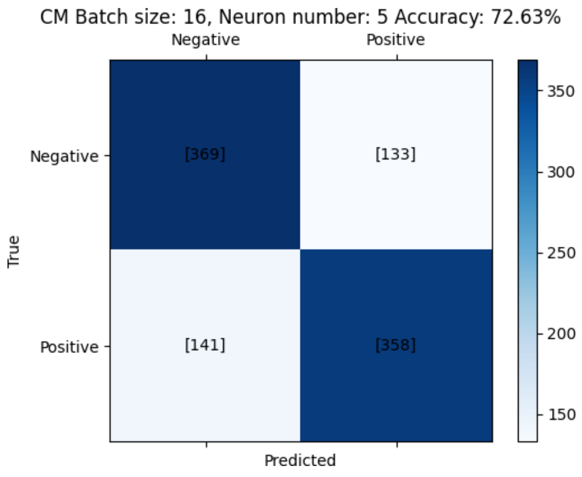
CM Batch size: 128, Neuron number: 7 Accuracy: 72.13%



In summary,

the findings indicate that optimal accuracy is achieved with a batch size of 32 and a hidden neuron count of 7. An increase in batch size can expedite the training process, albeit at the expense of reduced training depth. Conversely, during testing, it is observed that a higher number of hidden neurons correlates with increased accuracy. This suggests that the model encapsulates complex relationships, which preclude straightforward assumptions about these relationships. The complexity of these relationships underscores the necessity for more extensive data and a greater number of neurons to enhance the model's performance.





Best Result Neuron number:7 Batch size:32

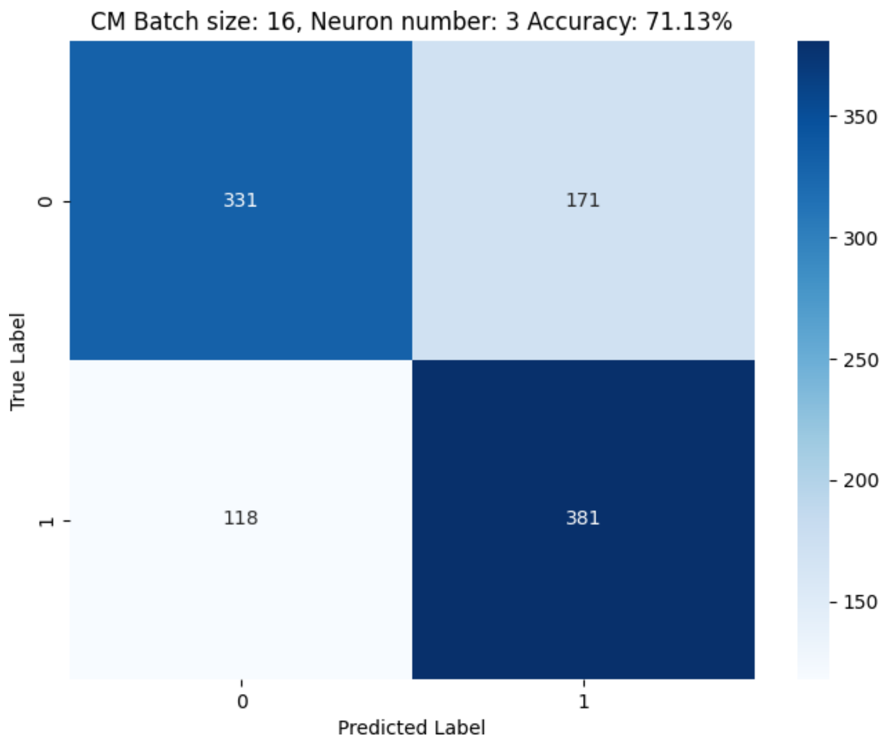
Second Best Result Neuron number:5 Batch size:16

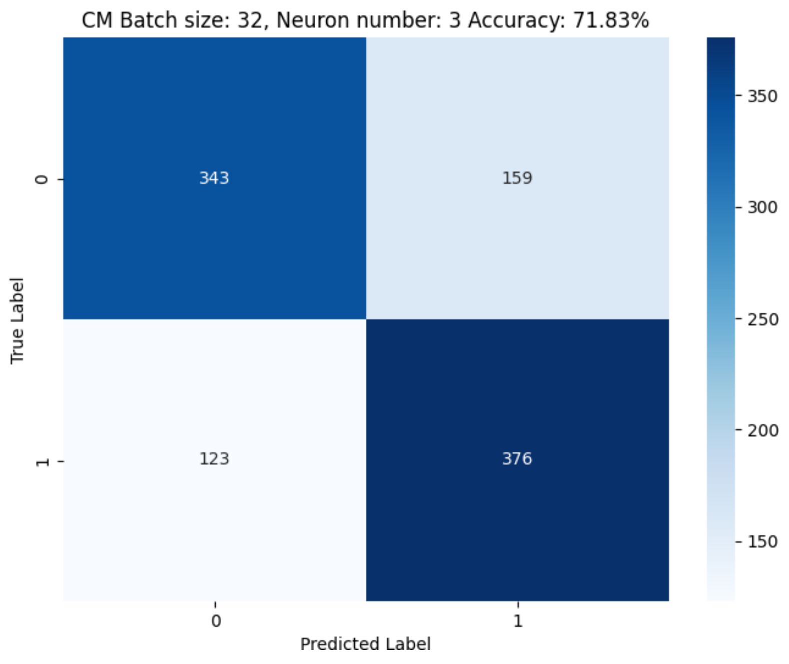
In that model generally low batch sizes; high neuron numbers should preferred.

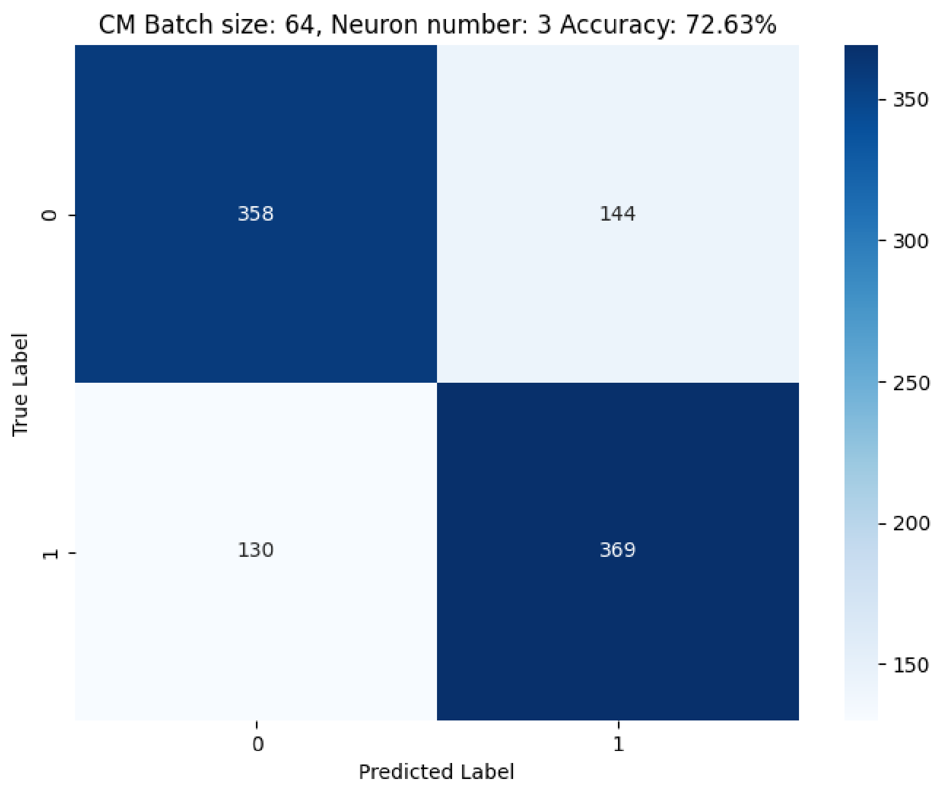
c. **Repeart part (b) with TensorFlow & Keras libraries.**

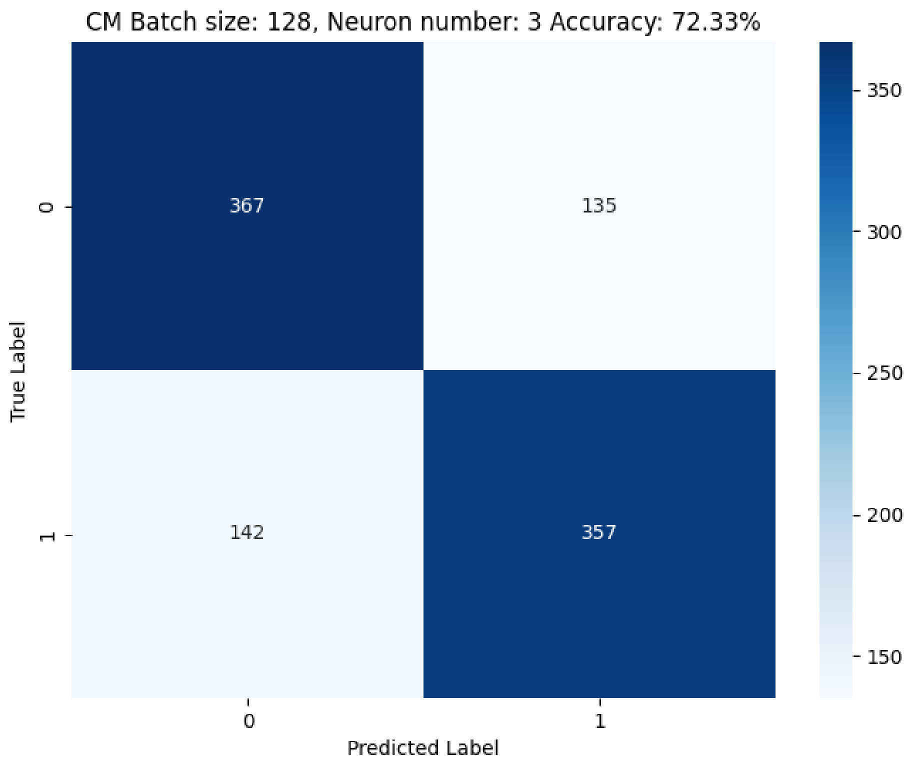
codes

When hidden neuron number is 3

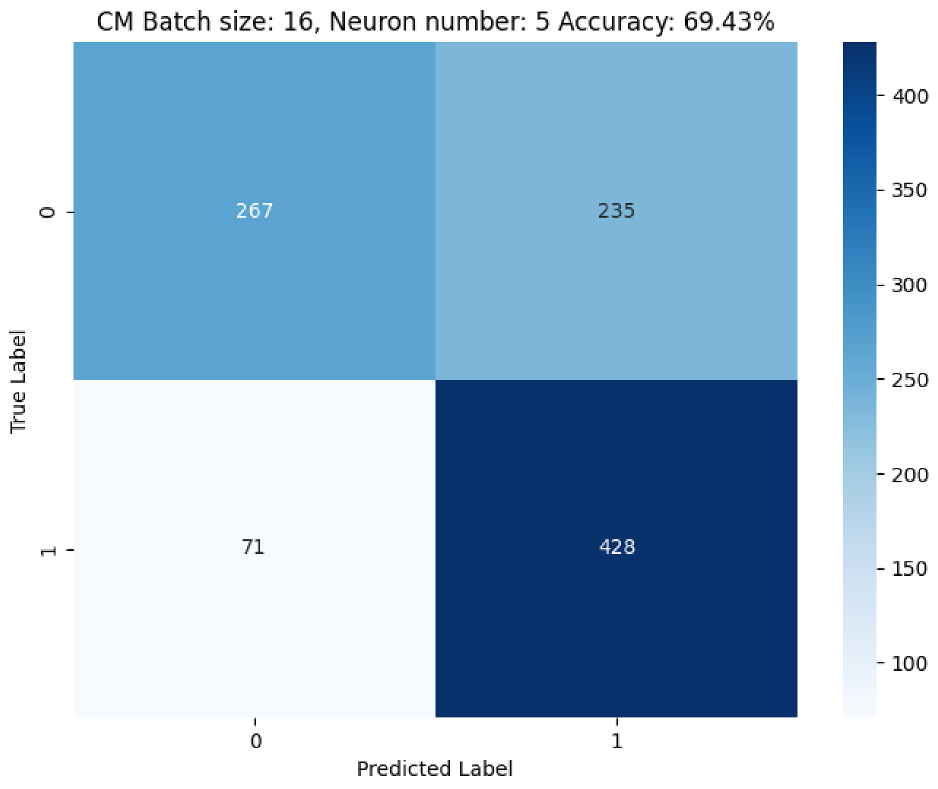


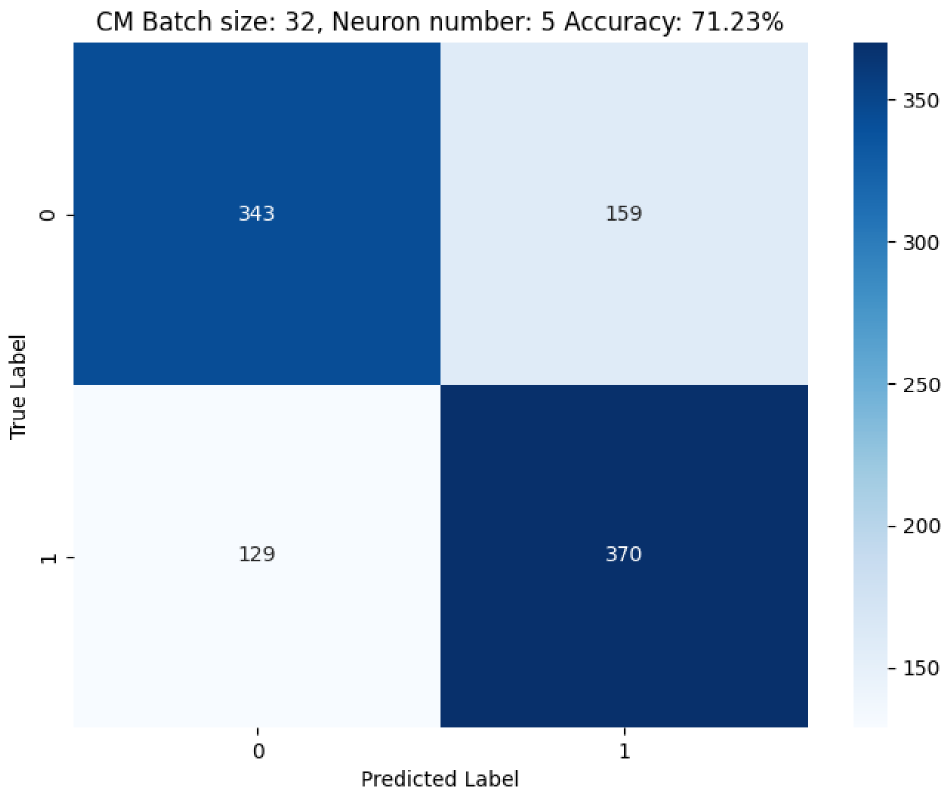


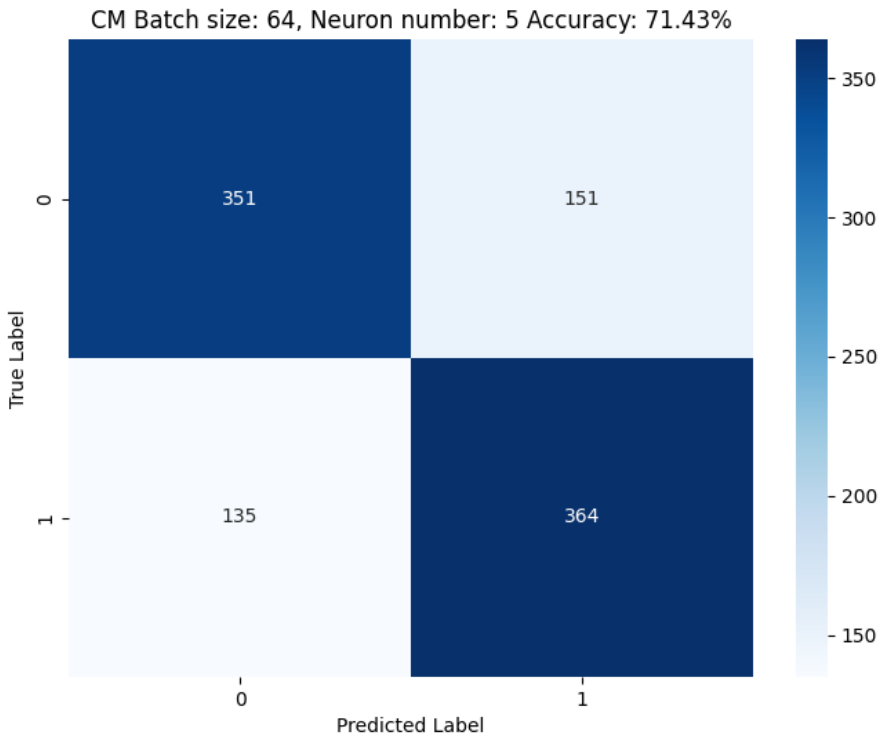

CM Batch size: 64, Neuron number: 3 Accuracy: 72.63%
358
144
130
369
True Label
Predicted Label
0
1
350
300
250
200
150

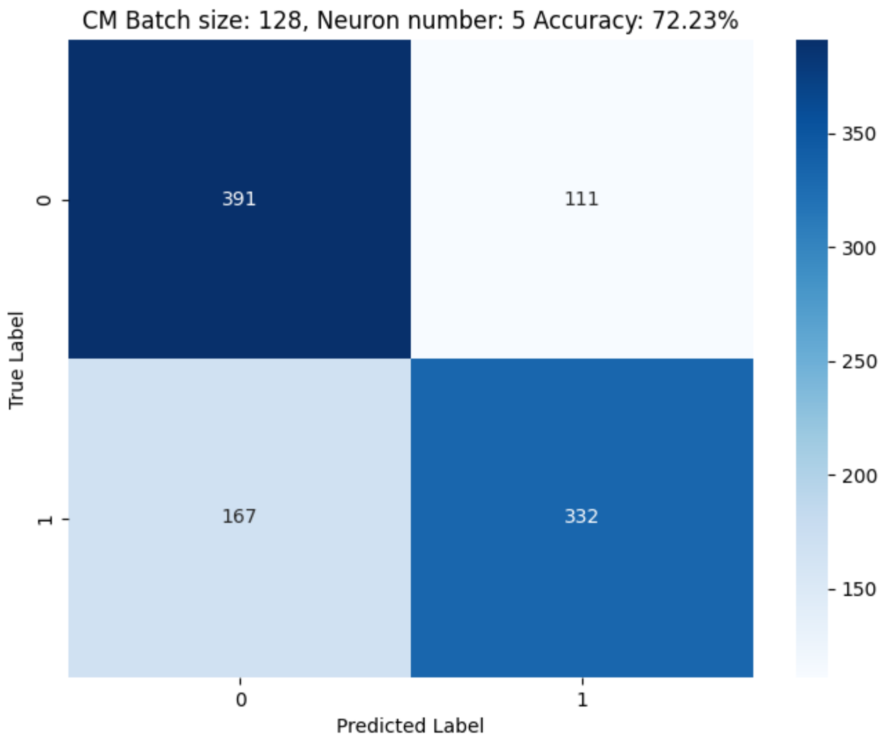

CM Batch size: 128, Neuron number: 3 Accuracy: 72.33%
367
135
142
357
True Label
Predicted Label
0
1
350
300
250
200
150

When hidden neuron number is 5

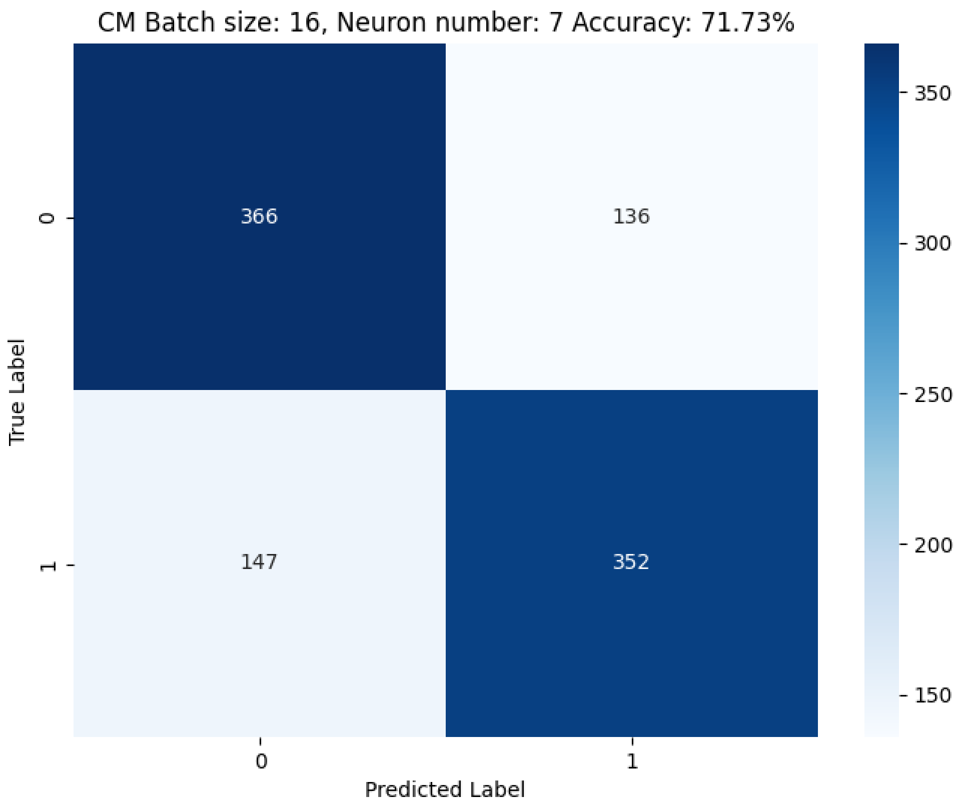


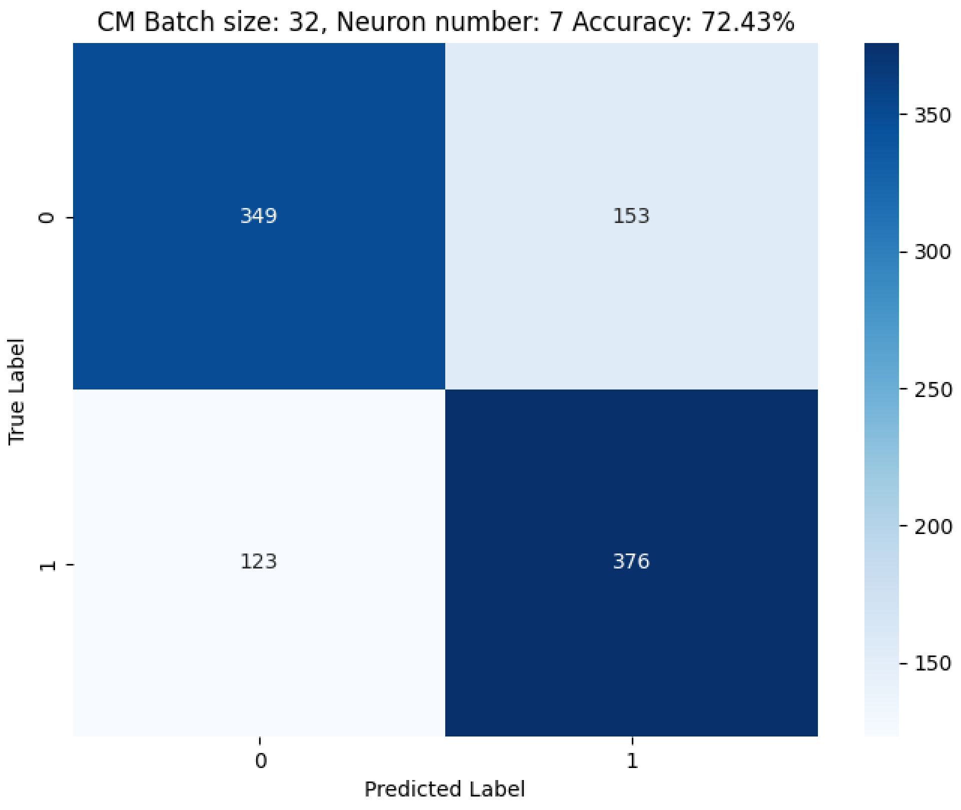


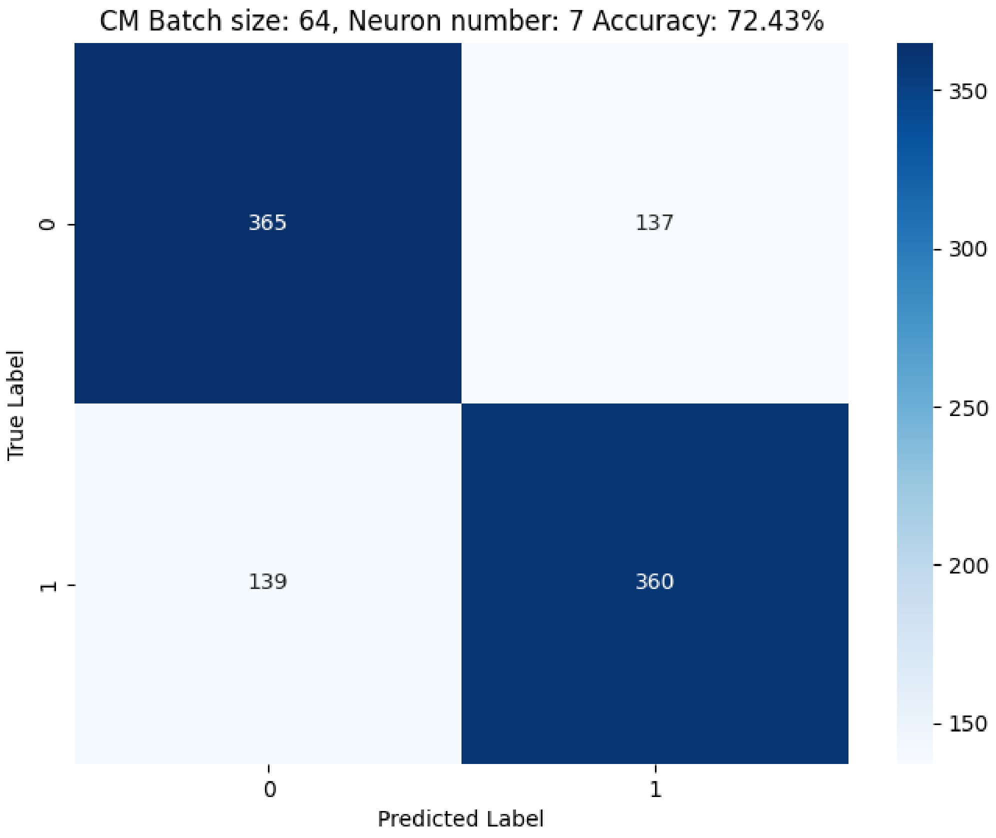

CM Batch size: 64, Neuron number: 5 Accuracy: 71.43%
True Label
Predicted Label
0
1
351
151
135
364
350
300
250
200
150

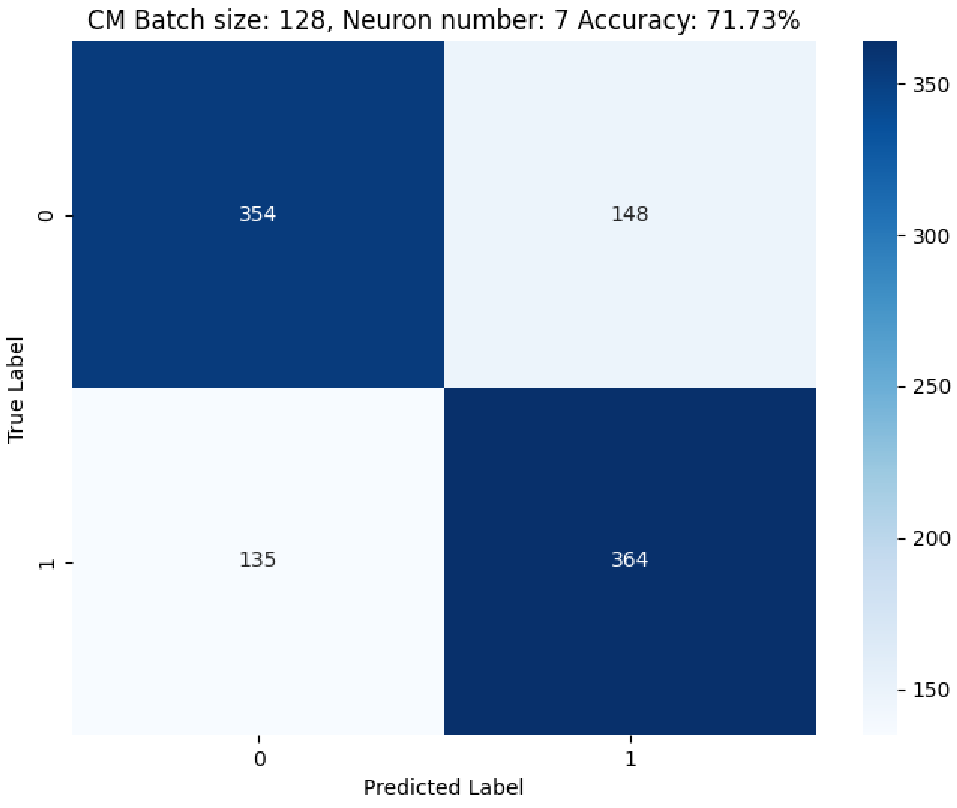

CM Batch size: 128, Neuron number: 5 Accuracy: 72.23%
True Label
Predicted Label
0
1
391
111
167
332
350
300
250
200
150

When hidden neuron number is 7



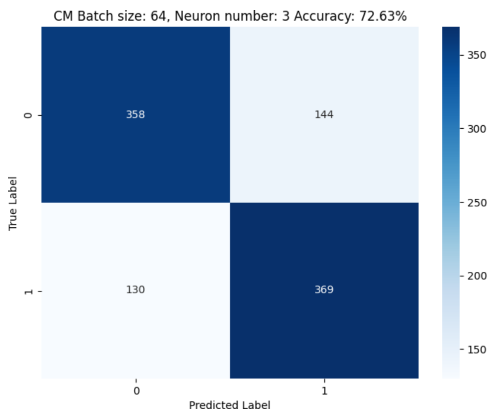


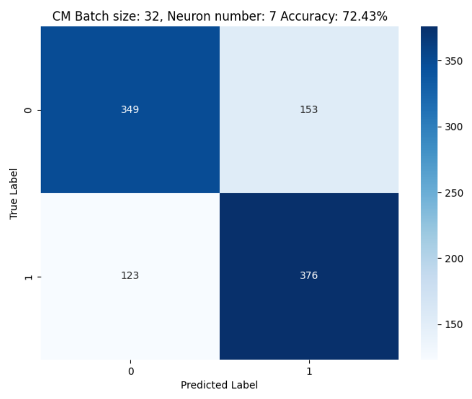

CM Batch size: 64, Neuron number: 7 Accuracy: 72.43%
True Label
Predicted Label
0
1
0
1
365
137
139
360
350
300
250
200
150

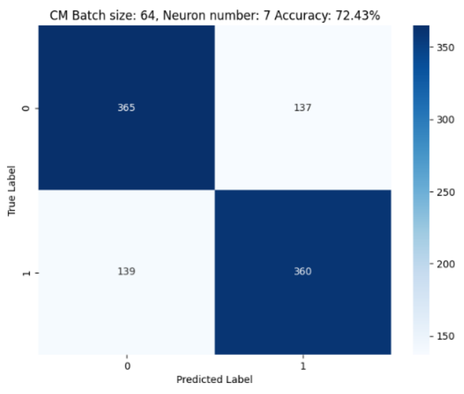

CM Batch size: 128, Neuron number: 7 Accuracy: 71.73%
True Label
Predicted Label
0
1
0
1
354
148
135
364
350
300
250
200
150

In conclusion,

the results demonstrate that the highest accuracy is achieved with a batch size of either 32 or 64, and a hidden neuron count of 7. An increase in batch size can facilitate a more rapid training process. However, in my model, a batch size of 64 did not yield favorable results, suggesting that different libraries may employ varying algorithms. Furthermore, during testing, it is not necessarily true that a higher number of hidden neurons will guarantee maximum accuracy. This indicates that while the model exhibits complex relationships, a smaller number of neurons can sometimes also produce satisfactory outcomes. Below, the top three results are presented for further examination..







Best Neuron:3 Batch:64   Second Neuron:7 Batch:32   Third Neuron: 7 Batch: 32

d. Using TensorFlow & Keras libraries and the same dataset, try to find a better NN topology (i.e., lower test error) by playing with number of hidden layers and neurons.

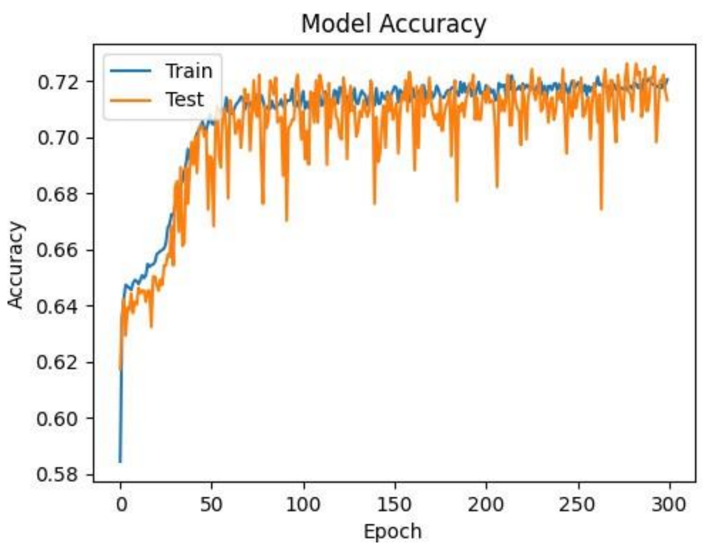
Please code on Google Colab and share your link here.

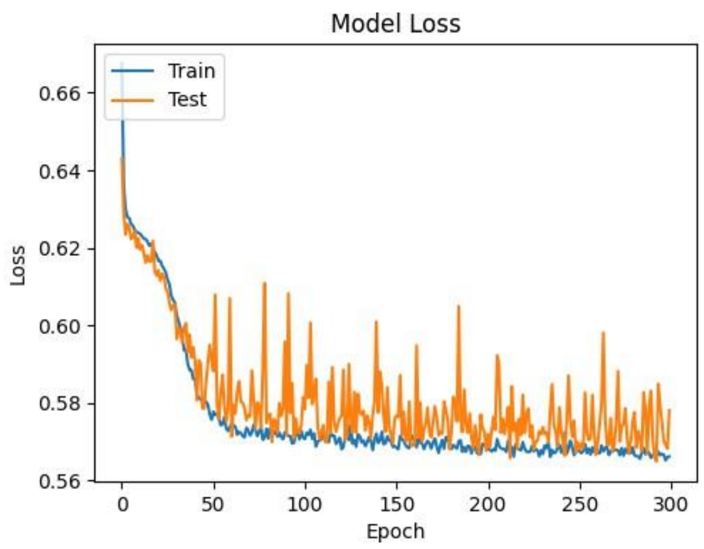
For the confusion matrix and evaluation metrics, please prepare a short report in pdf format and upload here.

code

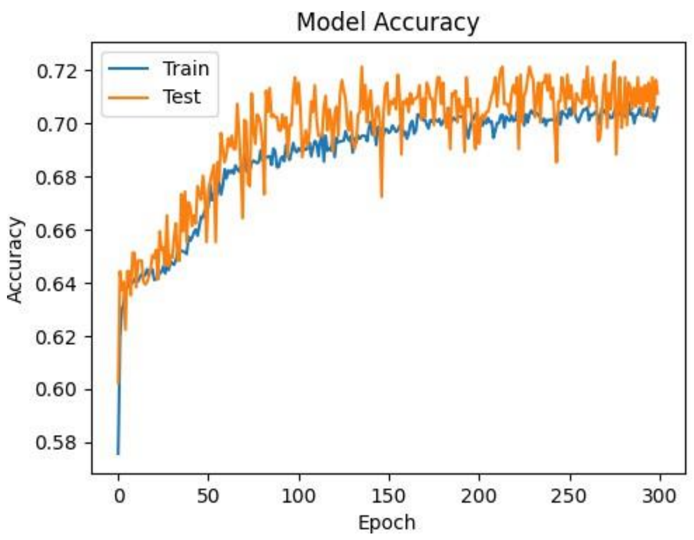
In the course of refining the neural network model, an interesting observation was made regarding the accuracy metrics. It was noted that upon reaching an accuracy threshold of 72% per epoch, there was a tendency for the accuracy to fluctuate, often surpassing the peak before declining. To address this, a strategy was implemented involving the reduction of the step size. This approach effectively prevented overshooting the peak accuracy. However, it necessitated an increase in the number of epochs to achieve comparable levels of accuracy.

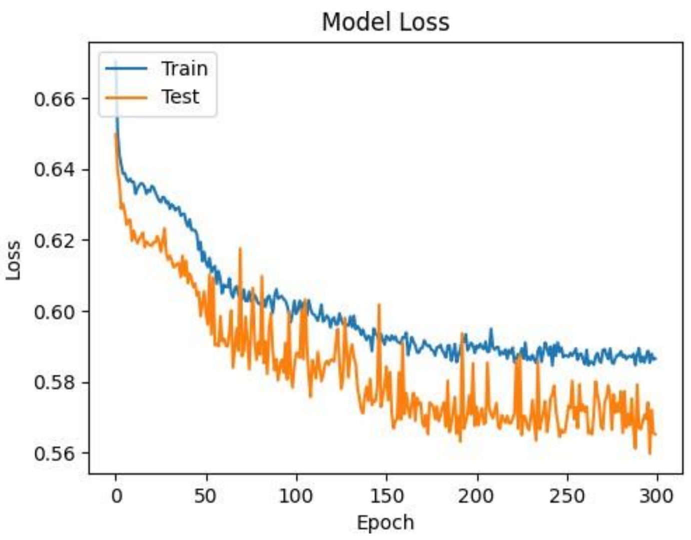
Additionally, to mitigate the risk of overfitting, the incorporation of dropout techniques was explored. Dropout, by randomly omitting units from the neural network during training, helps in preventing the model from becoming overly dependent on specific features of the training data. While this method has proven beneficial in enhancing the generalization capability of the model, it also introduces a complexity in the training process. Specifically, the training duration has notably increased, now requiring approximately one hour to complete..





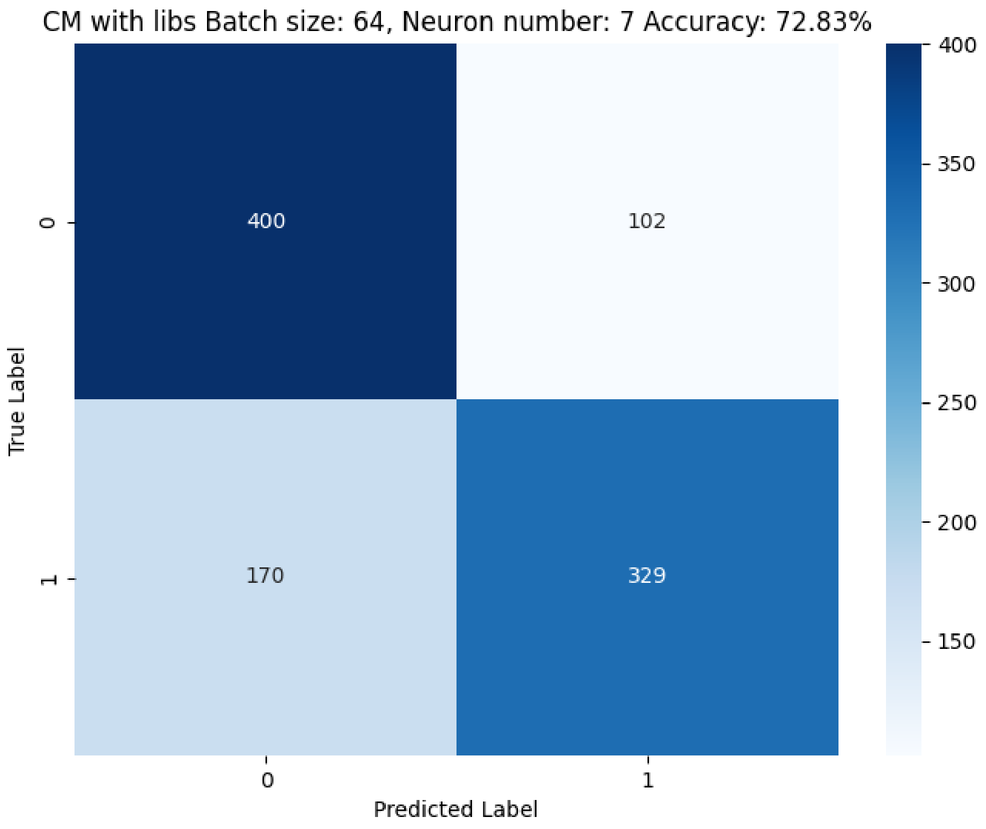
Without Dropout





With Dropout

The confusion matrix show the improved code for N.N result



The best result with dropout

- Batch size =64
- Hidden neuron=7
- Learning rate=0.005
- Epoch=300

References

The dataset that I used

https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset/data

introduction to the field of deep learning, including the basics of neural networks.

https://www.deeplearningbook.org/

to understand backpropagation and gradient descent there is a fundamental concept in that reference

https://jingyuexing.github.io/Ebook/Machine_Learning/Neural%20Networks%20and%20Deep%20Learning-eng.pdf

understanding the adam optimization algorithm

https://www.semanticscholar.org/paper/Adam%3A-A-Method-for-Stochastic-Optimization-Kingma-Ba/a6cb366736791bcccc5c8639de5a8f9636bf87e8?utm_source=direct_link