Empirical evaluation of GCML: an auxiliary attention mechanism for Convolutional Neural Networks

Machine Learning Summer Research

By Tobias Martin

1. Introduction

ES 499 [Professional Experience for Engineering Majors] is the course credit for Clarkson University's engineering curricula that every undergraduate student must fulfill. According to the university's 2020 – 2021 course catalogue, "The student must participate in a project-based professional experience such as a co-op, internship, directed research, significant responsibility in an appropriate team project, or a community project clearly related to the student's professional goals."(Clarkson).

This research was conducted under the supervision of Prof. Faraz Hussain of Clarkson University's Department of Electrical and Computer Engineering with extensive help from his advisee Edward Verenich, a Ph.D. candidate in the Department of Computer Science.

The purpose of the undergraduate research conducted over the Summer of 2021 can be summarized by Professor Hussain in his original email in the thread, "Undergraduate Research on ML in data scarce scenarios" [ML is shorthand for Machine Learning],

> Tobias, as I mentioned, Ed is my doctoral student working on solving machine learning problems that arise in specific situations such as data scarcity and class overlap. You will be helping in some of the machine learning model training, fine-tuning, and testing. The Python experience you have will come in handy here and you should be able to pick up a few new platforms/libraries, e.g. Pytorch.

2. Creating the Environment

Originally the research was to be conducted on a machine running Windows 10 with an NVIDIA 2600 Super. The choice of graphics card is important, as the code to be run was dependent on NVIDIA's Compute Unified Device Architecture (CUDA) toolkit. However, after encountering issues with installing and running Pytorch (another necessary coding library), Edward Verenich provided the following suggestion:

> I don't know how familiar you are with Ubuntu, but dual booting your machine may be the easiest approach here. Windows is known to be quirky with anaconda and pytorch. Installing Ubuntu it relatively straightforward, and it will come in handy for any future ML development that you do. You'll just have both OSs. Is this something that is feasible for you?

Taking this advice, the next step was to install Ubuntu Linux on this machine to dual boot both operating systems. The process was fairly straightforward, and began with creating a 250 gigabyte partition on the PC's hard drive to allocate space for the new OS.

After loading the Linux installation ISO onto a flash drive and using the free tool *Rufus*, all of the prerequisites to the installation were complete. After restarting the machine and booting it with the media drive, installing Ubuntu was simple. The next few days of research was spent verifying settings, growing comfortable with the new OS and installing all of the aforementioned requirements for the code to execute correctly (NVIDIA drivers, CUDA, Pytorch).

3. Research

The files train_gcml.py and test_gcml.py were provided by the research team for experimentation. The files work consecutively to produce two matrices, the first of which being the Globally Correlated Maximum Likelihood (*GCML*) and the second being the Convolutional Neural Network (*CNN*).

3.1. Parameters

The parameters of note that are used in experimentation are the cam_activation_point or "Tau Value", learning_rate and num_epochs. The Tau Value changed with every test, and the default values of the learning_rate is 0.001 and num_epochs is 100 in the train_gcml.py file and 1 in the test_gcml.py file. The following tests produced notable results from the experimentation pool. The results of each test can be found in the following section. Graduate researcher Edward Verenich provided definitions for each of the parameters and their purpose.

cam_activation_point - determines the threshold at which activation intensities are set to either 0 or 1 in the tensor that is then converted to a *GCML* datastore index.

num_epochs - refers to the number of times that the entire dataset is iterated over during training, one epoch consists of running a training session on the entire dataset

learning_rate - refers to the magnitude of change that gradient descent applies when updating weights when neural networks are trained.

3.1. Early Experimentation

3.1.1 Test 1

Tau Value (*TV*) = 0.009

The early testing stages were primarily focused on learning the structure of the code, and its functions. Therefore, the first tau value was chosen arbitrarily as a “small” integer.

3.1.2. Test 2

TV = 500.0

Continuing from the first test’s theme of experimentation, the second tau value was made a “large” integer.

3.1.3. Test 3

TV = 0.0000000000005

Evaluating the early tests and noticing promising results in the lower boundary meant further experimentation with smaller tau values.

3.1.4. Test 4

TV = 0.00000000000000000000005

This tau value was determined by doubling the amount of zeros present in the previous test’s value.

3.1.5. Test 5

TV = 0.0000000000000000000000000000000000000000000000005

Similarly to test 4, this tau value was determined by doubling the amount of zeros present in the previous test’s value.

3.1.6. Test 6

TV = 10,000.0

The next step was to further test in the opposite direction. The tau value was chosen as a large, positive integer to further experiment with larger values.

3.1.6. Test 7

TV = 100,000,000.0

Expanding on the premise from the previous test, this tau value was chosen to further explore the results from larger values.

3.2. Training other Parameters

3.2.1. Test 8

TV = 0.009

Learning Rate (*LR*) = 0.005

After a brief meeting with Edward Verenich, his suggestion was to branch out and examine the results when the other parameters are altered. The first of these tests were conducted with an altered learning_rate and a previously established tau value.

3.2.2. Test 9

TV = 0.499

Test 9 acts as a control for the following tests which focus on the number of epochs for both the train_gcml.py and the test_gcml.py files.

3.2.3. Test 10

TV = 0.499

Number of Epochs of train_gcml (*NE_train*) = 200

The next conducted test was focused on how doubling the num_epochs value of the train_gcml.py file would affect the results.

3.2.4. Test 11

TV = 0.499

Number of Epochs of test_gcml (*NE_test*) = 20

The following test was focused on how increasing the num_epochs value of the test_gcml.py file would affect the results.

3.3. Testing for Accuracy

3.3.1. Test 12

$TV = 0.999$

After seeing how the other parameters interacted with the accuracies, the testing pivoted to trying to get both GCML and CNN accuracy as close to 100% as possible. The following tests used varying tau values to find a rough estimate as to where the most accurate results could be found.

3.3.2. Test 13

$TV = 9.999$

Continuing from the previous test, this experiment was conducted to see if more accurate results could be produced from a positive integer.

3.3.3. Test 14

$TV = 0.999999999$

This test was conducted to see if a value approaching 1 but not reaching it would be sufficient in producing accurate data.

3.3.4. Test 15

$TV = 0.0009$

Recognizing that the most accurate results in the past were formed from smaller tau values, the value of this test was chosen as one that would result in an accurate outcome.

3.3.5. Test 16

$TV = 0.0009$

$NE\_train = 200$

Satisfied with the results found with this smaller tau value, the results were made even more accurate with the inclusion of more epochs.

3.3.6. Test 17

TV = 0.00011

The additional percentage increase in accuracy provided by increasing epochs was not so great as to stop testing for better results with different tau values. Still trying to find the most accurate results possible, the tau value once again was altered.

3.3.7. Test 18

TV = 0.0001

Much like the previous test, this was conducted in an attempt to find the most accurate results with only the tau value being altered.

3.3.4. Test 19

TV = 0.0001

NE_train = 500

The final significant test was taken with test 18's tau value with the addition of an increased amount of epochs in the train_gcml.py file. Counterintuitively to the theory of increasing epochs generating more accurate results, this test's *GCML* accuracy was less than that of test 18.

4. Conclusions

The results of the experiments conducted in this research were used in a paper titled "*Pulmonary Disease Classification Using Globally Correlated Maximum Likelihood: an Auxiliary Attention mechanism for Convolutional Neural Networks*", authored by Edward Verenich, Tobias Martin, Alvaro Velasquez, Nazar Khan, and Faraz Hussain. The paper has been submitted for review at an IEEE journal, and per the abstract, "present[s] a novel technique that serves as an auxiliary attention mechanism to existing CNN architectures, in order to extract global correlations between salient features."(Edward).

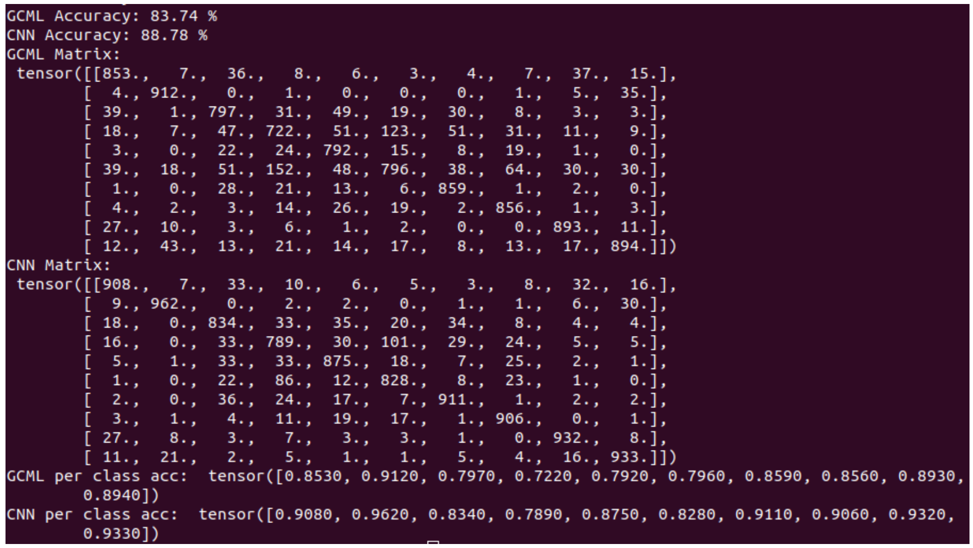
5. Results

Test 1:

TV: 0.009

GCML: 83.74%

CNN: 88.78%



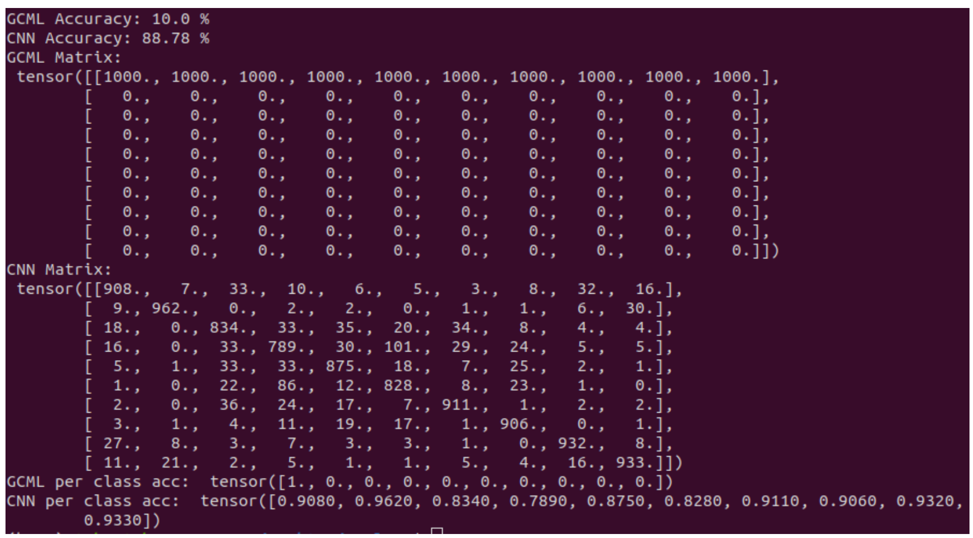
```
GCML Accuracy: 83.74 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[853.,   7.,  36.,   8.,   6.,   3.,   4.,   7.,  37.,  15.],
        [  4., 912.,   0.,   1.,   0.,   0.,   0.,   1.,   5.,  35.],
        [ 39.,   1., 797.,  31.,  49.,  19.,  30.,   8.,   3.,   3.],
        [ 18.,   7.,  47., 722.,  51., 123.,  51.,  31.,  11.,   9.],
        [  3.,   0.,  22.,  24., 792.,  15.,   8.,  19.,   1.,   0.],
        [ 39.,  18.,  51., 152.,  48., 796.,  38.,  64.,  30.,  30.],
        [  1.,   0.,  28.,  21.,  13.,   6., 859.,   1.,   2.,   0.],
        [  4.,   2.,   3.,  14.,  26.,  19.,   2., 856.,   1.,   3.],
        [ 27.,  10.,   3.,   6.,   1.,   2.,   0.,   0., 893.,  11.],
        [ 12.,  43.,  13.,  21.,  14.,  17.,   8.,  13.,  17., 894.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8530, 0.9120, 0.7970, 0.7220, 0.7920, 0.7960, 0.8590, 0.8560, 0.8930,
        0.8940])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 2:

TV: 500.0

GCML: 10.0%

CNN: 88.78%



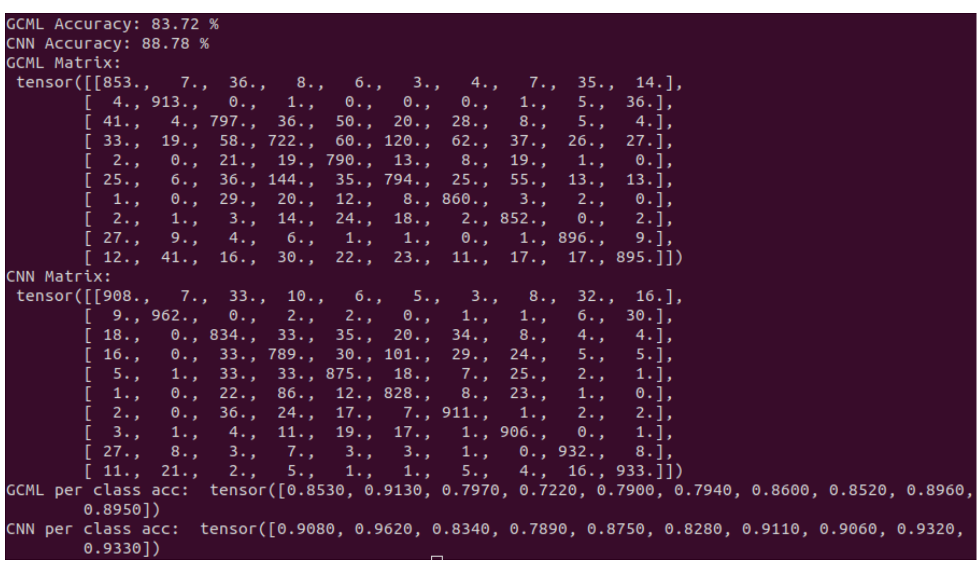
```
GCML Accuracy: 10.0 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([1., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 3:

TV:
0.0000000000005

GCML: 83.82%

CNN: 88.78%



```
GCML Accuracy: 83.72 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[853.,   7.,  36.,   8.,   6.,   3.,   4.,   7.,  35.,  14.],
        [  4., 913.,   0.,   1.,   0.,   0.,   0.,   1.,   5.,  36.],
        [ 41.,   4., 797.,  36.,  50.,  20.,  28.,   8.,   5.,   4.],
        [ 33.,  19.,  58., 722.,  60., 120.,  62.,  37.,  26.,  27.],
        [  2.,   0.,  21.,  19., 790.,  13.,   8.,  19.,   1.,   0.],
        [ 25.,   6.,  36., 144.,  35., 794.,  25.,  55.,  13.,  13.],
        [  1.,   0.,  29.,  20.,  12.,   8., 860.,   3.,   2.,   0.],
        [  2.,   1.,   3.,  14.,  24.,  18.,   2., 852.,   0.,   2.],
        [ 27.,   9.,   4.,   6.,   1.,   1.,   0.,   1., 896.,   9.],
        [ 12.,  41.,  16.,  30.,  22.,  23.,  11.,  17.,  17., 895.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8530, 0.9130, 0.7970, 0.7220, 0.7900, 0.7940, 0.8600, 0.8520, 0.8960,
        0.8950])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 4:

TV: 0.000000000000000000000005

GCML: 83.72%

CNN: 88.78%

```
GCML Accuracy: 83.82 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[854.,   7.,  36.,   9.,   6.,   3.,   4.,   7.,  36.,  14.],
        [  4., 913.,   0.,   1.,   0.,   0.,   0.,   1.,   5.,  36.],
        [ 46.,   5., 797.,  50.,  56.,  31.,  28.,  14.,   6.,   4.],
        [ 17.,   8.,  44., 719.,  48., 121.,  50.,  31.,  10.,   8.],
        [  2.,   0.,  22.,  20., 793.,  12.,   8.,  19.,   1.,   0.],
        [ 40.,  17.,  51., 147.,  49., 795.,  33.,  64.,  28.,  32.],
        [  1.,   0.,  29.,  20.,  13.,   5., 864.,   1.,   2.,   0.],
        [  4.,   1.,  10.,  18.,  28.,  26.,   9., 856.,   2.,   2.],
        [ 25.,   9.,   4.,   6.,   1.,   3.,   0.,   1., 896.,   9.],
        [  7.,  40.,   7.,  10.,   6.,   4.,   4.,   6.,  14., 895.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8540, 0.9130, 0.7970, 0.7190, 0.7930, 0.7950, 0.8640, 0.8560, 0.8960,
        0.8950])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 5:

TV: 0.000000000000000000000000000000000000000000000005

GCML: 83.83%

CNN: 88.78%

```
GCML Accuracy: 83.83 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[855.,   7.,  36.,   9.,   6.,   3.,   4.,   7.,  36.,  14.],
        [  4., 914.,   0.,   1.,   0.,   0.,   0.,   1.,   5.,  35.],
        [ 39.,   3., 796.,  37.,  50.,  18.,  28.,   8.,   3.,   3.],
        [ 17.,   8.,  44., 721.,  49., 121.,  49.,  32.,  11.,   8.],
        [  2.,   0.,  22.,  21., 793.,  13.,   8.,  20.,   1.,   0.],
        [ 40.,  16.,  51., 148.,  47., 794.,  37.,  63.,  28.,  32.],
        [  1.,   0.,  29.,  20.,  13.,   7., 863.,   1.,   2.,   0.],
        [  8.,   3.,  10.,  22.,  30.,  35.,   7., 857.,   4.,   4.],
        [ 27.,   9.,   4.,   6.,   1.,   1.,   0.,   1., 896.,  10.],
        [  7.,  40.,   8.,  15.,  11.,   8.,   4.,  10.,  14., 894.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8550, 0.9140, 0.7960, 0.7210, 0.7930, 0.7940, 0.8630, 0.8570, 0.8960,
        0.8940])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 6:

TV: 10000.0

GCML: 10.0%

CNN: 88.78%

```
GCML Accuracy: 10.0 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([1., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 7:

TV: 100000000.0

GCML: 10.0%

CNN: 88.78%

```
GCML Accuracy: 10.0 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([1., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 8:

TV: 0.009

LR: 0.005

GCML: 83.77%

CNN: 88.78%

```
GCML Accuracy: 83.77 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[849.,   6.,  36.,   8.,   6.,   3.,   4.,   7.,  39.,  14.],
        [  4., 914.,   0.,   1.,   0.,   0.,   0.,   1.,   5.,  34.],
        [ 42.,   2., 797.,  32.,  48.,  20.,  32.,   8.,   3.,   3.],
        [ 34.,  17.,  60., 726.,  60., 125.,  66.,  42.,  28.,  26.],
        [  4.,   2.,  23.,  25., 795.,  15.,   8.,  19.,   2.,   1.],
        [ 23.,   6.,  37., 144.,  36., 796.,  25.,  55.,  13.,  11.],
        [  1.,   0.,  27.,  22.,  13.,   7., 855.,   1.,   2.,   0.],
        [  3.,   0.,   2.,  13.,  25.,  14.,   2., 855.,   0.,   2.],
        [ 28.,  10.,   3.,   6.,   1.,   2.,   0.,   0., 892.,  11.],
        [ 12.,  43.,  15.,  23.,  16.,  18.,   8.,  12.,  16., 898.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8490, 0.9140, 0.7970, 0.7260, 0.7950, 0.7960, 0.8550, 0.8550, 0.8920,
        0.8980])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 9:

TV: 0.499

GCML: 73.56%

CNN: 88.78%

```
GCML Accuracy: 73.56 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[711.,   8.,  42.,  14.,   6.,   7.,   8.,  11.,  28.,   9.],
        [ 20., 802.,   4.,   7.,   1.,   1.,   9.,   2.,  30., 134.],
        [ 62.,   3., 718.,  41.,  50.,  27.,  42.,   8.,  11.,   5.],
        [ 34.,   9.,  59., 659.,  92., 126.,  56.,  52.,  22.,  16.],
        [ 10.,   1.,  23.,  23., 619.,  15.,  11.,  20.,   2.,   1.],
        [ 64.,  61.,  79., 175., 148., 770.,  71., 105.,  97.,  68.],
        [  8.,   4.,  25.,  29.,  18.,   7., 778.,   2.,   5.,   1.],
        [ 20.,   1.,  24.,  22.,  55.,  32.,   7., 793.,   2.,   6.],
        [ 29.,  24.,   4.,   7.,   1.,   0.,   1.,   2., 760.,  14.],
        [ 42.,  87.,  22.,  23.,  10.,  15.,  17.,   5.,  43., 746.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.7110, 0.8020, 0.7180, 0.6590, 0.6190, 0.7700, 0.7780, 0.7930, 0.7600,
        0.7460])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 10:

TV: 0.499

NE_train: 200

GCML: 73.74%

CNN: 88.78%

```
GCML Accuracy: 73.74 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[719.,   8.,  43.,  14.,   6.,   6.,   9.,  11.,  30.,   9.],
        [ 20., 807.,   3.,   6.,   1.,   2.,   9.,   2.,  30., 135.],
        [ 63.,   3., 721.,  37.,  50.,  28.,  42.,   7.,  13.,   6.],
        [ 34.,   9.,  63., 656., 100., 128.,  60.,  56.,  22.,  16.],
        [ 10.,   1.,  22.,  21., 622.,  16.,  12.,  22.,   2.,   1.],
        [ 55.,  55.,  73., 176., 132., 767.,  66., 100.,  83.,  64.],
        [  8.,   4.,  25.,  27.,  19.,   7., 778.,   2.,   6.,   1.],
        [ 21.,   1.,  23.,  24.,  57.,  31.,   6., 791.,   4.,   7.],
        [ 29.,  24.,   4.,   8.,   1.,   0.,   1.,   2., 766.,  14.],
        [ 41.,  88.,  23.,  31.,  12.,  15.,  17.,   7.,  44., 747.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.7190, 0.8070, 0.7210, 0.6560, 0.6220, 0.7670, 0.7780, 0.7910, 0.7660,
        0.7470])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 11:

TV: 0.499

NE_test: 20

GCML: 73.51%

CNN: 88.78%

```
GCML Accuracy: 73.51 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[718.,   7.,  44.,  13.,   6.,   6.,   8.,  13.,  30.,   9.],
        [ 21., 804.,   3.,   6.,   1.,   1.,   9.,   3.,  31., 136.],
        [ 61.,   3., 719.,  36.,  46.,  26.,  42.,   5.,  10.,   5.],
        [ 35.,  10.,  61., 655.,  99., 128.,  61.,  60.,  24.,  17.],
        [ 10.,   1.,  23.,  22., 615.,  16.,  12.,  21.,   2.,   1.],
        [ 60.,  58.,  73., 182., 144., 769.,  68., 101.,  89.,  68.],
        [  8.,   3.,  26.,  27.,  19.,   8., 775.,   2.,   6.,   1.],
        [ 20.,   0.,  24.,  24.,  55.,  31.,   6., 788.,   2.,   6.],
        [ 27.,  26.,   4.,   8.,   1.,   0.,   1.,   2., 763.,  12.],
        [ 40.,  88.,  23.,  27.,  14.,  15.,  18.,   5.,  43., 745.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.7180, 0.8040, 0.7190, 0.6550, 0.6150, 0.7690, 0.7750, 0.7880, 0.7630,
        0.7450])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 12:

TV: 0.999

GCML: 70.74%

CNN: 88.78%

```
GCML Accuracy: 70.74 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[750.,  12.,  61.,  24.,  21.,  11.,  16.,  22.,  47.,  18.],
        [ 24., 616.,   5.,   7.,   2.,   6.,  12.,   7.,  55.,  59.],
        [ 60.,   1., 658.,  45.,  55.,  24.,  55.,   9.,  10.,   5.],
        [ 25.,   7.,  48., 591.,  69., 135.,  60.,  36.,  18.,   9.],
        [ 12.,   1.,  21.,  26., 629.,  12.,   9.,  32.,   4.,   2.],
        [  3.,   6.,  53., 201., 109., 732.,  24., 155.,  12.,   9.],
        [ 19.,  15.,  56.,  42.,  28.,  23., 789.,  13.,   5.,  14.],
        [ 24.,   3.,  41.,  34.,  71.,  43.,  12., 712.,  15.,  16.],
        [ 33.,  31.,  15.,  19.,   6.,   2.,   5.,   2., 753.,  24.],
        [ 50., 308.,  42.,  11.,  10.,  12.,  18.,  12.,  81., 844.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.7500, 0.6160, 0.6580, 0.5910, 0.6290, 0.7320, 0.7890, 0.7120, 0.7530,
        0.8440])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 13:

TV: 9.999

GCML: 10.0%

CNN: 88.78%

```
GCML Accuracy: 10.0 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([1., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 14:

TV: 0.999999999

GCML: 10.0%

CNN: 88.78%

```
GCML Accuracy: 10.0 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000., 1000.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.],
        [   0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([1., 0., 0., 0., 0., 0., 0., 0., 0., 0.])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 15:

TV: 0.0009

GCML: 83.66%

CNN: 88.78%

```
GCML Accuracy: 83.66 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[853.,   7.,  36.,   9.,   6.,   3.,   4.,   7.,  37.,  14.],
        [  4., 912.,   0.,   1.,   0.,   0.,   0.,   1.,   5.,  36.],
        [ 42.,   4., 795.,  35.,  50.,  21.,  28.,   7.,   6.,   4.],
        [ 17.,   8.,  45., 721.,  51., 120.,  50.,  30.,   9.,   9.],
        [  2.,   1.,  23.,  21., 789.,  15.,   9.,  20.,   1.,   0.],
        [ 41.,  17.,  50., 146.,  46., 792.,  36.,  63.,  28.,  32.],
        [  1.,   0.,  28.,  21.,  13.,   7., 861.,   1.,   2.,   0.],
        [  2.,   0.,   3.,  13.,  26.,  18.,   2., 855.,   0.,   2.],
        [ 26.,   9.,   4.,   6.,   1.,   1.,   0.,   0., 895.,  10.],
        [ 12.,  42.,  16.,  27.,  18.,  23.,  10.,  16.,  17., 893.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8530, 0.9120, 0.7950, 0.7210, 0.7890, 0.7920, 0.8610, 0.8550, 0.8950,
        0.8930])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 16:

TV: 0.0009

NE_train: 200

GCML: 83.72%

CNN: 88.78%

```
GCML Accuracy: 83.72 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[853.,   7.,  36.,   9.,   6.,   3.,   4.,   7.,  36.,  14.],
        [  4., 912.,   0.,   1.,   0.,   0.,   0.,   1.,   5.,  36.],
        [ 42.,   4., 795.,  34.,  51.,  23.,  28.,   6.,   5.,   5.],
        [ 17.,   9.,  45., 721.,  50., 119.,  50.,  30.,  11.,   8.],
        [  2.,   0.,  22.,  21., 791.,  12.,   9.,  20.,   1.,   0.],
        [ 39.,  16.,  50., 146.,  45., 794.,  35.,  63.,  28.,  34.],
        [  1.,   0.,  29.,  21.,  13.,   5., 862.,   1.,   2.,   0.],
        [  2.,   1.,   4.,  14.,  25.,  20.,   2., 858.,   0.,   2.],
        [ 27.,   9.,   4.,   6.,   1.,   2.,   0.,   0., 895.,  10.],
        [ 13.,  42.,  15.,  27.,  18.,  22.,  10.,  14.,  17., 891.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8530, 0.9120, 0.7950, 0.7210, 0.7910, 0.7940, 0.8620, 0.8580, 0.8950,
        0.8910])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 17:

tv: 0.0011

GCML: 83.8%

CNN: 88.78%

```
GCML Accuracy: 83.8 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[853.,   7.,  36.,   9.,   6.,   3.,   4.,   7.,  36.,  14.],
        [  4., 914.,   0.,   1.,   0.,   0.,   0.,   1.,   4.,  36.],
        [ 41.,   3., 796.,  35.,  50.,  19.,  28.,   6.,   3.,   3.],
        [ 18.,   8.,  44., 719.,  49., 119.,  50.,  30.,  10.,   8.],
        [  2.,   0.,  22.,  21., 792.,  12.,   9.,  20.,   1.,   0.],
        [ 39.,  15.,  51., 148.,  46., 795.,  35.,  62.,  29.,  31.],
        [  1.,   0.,  28.,  21.,  13.,   5., 861.,   1.,   2.,   0.],
        [  3.,   2.,   4.,  14.,  27.,  23.,   3., 859.,   2.,   3.],
        [ 27.,   9.,   4.,   6.,   1.,   3.,   0.,   0., 896.,  10.],
        [ 12.,  42.,  15.,  26.,  16.,  21.,  10.,  14.,  17., 895.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8530, 0.9140, 0.7960, 0.7190, 0.7920, 0.7950, 0.8610, 0.8590, 0.8960,
        0.8950])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 18:

TV: 0.0001

GCML: 83.79%

CNN: 88.78%

```
GCML Accuracy: 83.79 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[854.,   7.,  36.,   9.,   6.,   3.,   4.,   7.,  36.,  14.],
        [  4., 915.,   0.,   1.,   0.,   0.,   0.,   1.,   5.,  36.],
        [ 39.,   3., 798.,  34.,  50.,  19.,  28.,   8.,   3.,   3.],
        [ 17.,   8.,  43., 720.,  47., 121.,  50.,  30.,   9.,   9.],
        [  2.,   0.,  22.,  21., 790.,  13.,   8.,  19.,   1.,   0.],
        [ 40.,  14.,  50., 145.,  47., 792.,  33.,  64.,  28.,  31.],
        [  1.,   0.,  29.,  20.,  14.,   7., 864.,   1.,   2.,   0.],
        [  4.,   2.,   3.,  14.,  27.,  22.,   3., 854.,   2.,   3.],
        [ 27.,   9.,   4.,   8.,   1.,   1.,   0.,   1., 897.,   9.],
        [ 12.,  42.,  15.,  28.,  18.,  22.,  10.,  15.,  17., 895.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8540, 0.9150, 0.7980, 0.7200, 0.7900, 0.7920, 0.8640, 0.8540, 0.8970,
        0.8950])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

Test 19:

TV: 0.0001

NE_train: 500

GCML: 83.74%

CNN: 88.78%

```
GCML Accuracy: 83.74 %
CNN Accuracy: 88.78 %
GCML Matrix:
 tensor([[854.,   7.,  36.,   9.,   6.,   3.,   4.,   7.,  36.,  14.],
        [  4., 912.,   0.,   1.,   0.,   0.,   0.,   1.,   5.,  36.],
        [ 39.,   3., 797.,  36.,  51.,  18.,  28.,   9.,   3.,   3.],
        [ 17.,   8.,  44., 719.,  47., 122.,  50.,  30.,  10.,   9.],
        [  2.,   0.,  22.,  20., 792.,  12.,   8.,  20.,   1.,   0.],
        [ 41.,  17.,  50., 145.,  47., 795.,  35.,  63.,  28.,  33.],
        [  1.,   0.,  29.,  20.,  13.,   7., 862.,   1.,   2.,   0.],
        [  4.,   2.,   3.,  14.,  27.,  21.,   3., 854.,   2.,   3.],
        [ 27.,   9.,   4.,   7.,   1.,   1.,   0.,   1., 896.,   9.],
        [ 11.,  42.,  15.,  29.,  16.,  21.,  10.,  14.,  17., 893.]])
CNN Matrix:
 tensor([[908.,   7.,  33.,  10.,   6.,   5.,   3.,   8.,  32.,  16.],
        [  9., 962.,   0.,   2.,   2.,   0.,   1.,   1.,   6.,  30.],
        [ 18.,   0., 834.,  33.,  35.,  20.,  34.,   8.,   4.,   4.],
        [ 16.,   0.,  33., 789.,  30., 101.,  29.,  24.,   5.,   5.],
        [  5.,   1.,  33.,  33., 875.,  18.,   7.,  25.,   2.,   1.],
        [  1.,   0.,  22.,  86.,  12., 828.,   8.,  23.,   1.,   0.],
        [  2.,   0.,  36.,  24.,  17.,   7., 911.,   1.,   2.,   2.],
        [  3.,   1.,   4.,  11.,  19.,  17.,   1., 906.,   0.,   1.],
        [ 27.,   8.,   3.,   7.,   3.,   3.,   1.,   0., 932.,   8.],
        [ 11.,  21.,   2.,   5.,   1.,   1.,   5.,   4.,  16., 933.]])
GCML per class acc:  tensor([0.8540, 0.9120, 0.7970, 0.7190, 0.7920, 0.7950, 0.8620, 0.8540, 0.8960,
        0.8930])
CNN per class acc:  tensor([0.9080, 0.9620, 0.8340, 0.7890, 0.8750, 0.8280, 0.9110, 0.9060, 0.9320,
        0.9330])
```

5. Works Cited

Clarkson University. (2021). Clarkson University Courses for the Academic Year 2020-2021. clarkson.edu. https://www.clarkson.edu/sites/default/files/2020-06/Final%20Catalog%202020-2021.pdf.

Edward Verenich, Tobias Martin, Alvaro Velasquez, Nazar Khan, & Faraz Hussain. (2021). Pulmonary Disease Classification Using Globally Correlated Maximum Likelihood: an Auxiliary Attention mechanism for Convolutional Neural Networks.