Information Technology Course
Module Software Engineering
by Damir Dobric / Andreas Pech

Analyse Image Classification of MNIST Dataset

Alam Sher Khan
alam.khan@stud.fra-uas.de

Aiman Zehra
aiman.-@stud.fra-uas.de

Soundarya Talawai
soundarya.talawai@stud.fra-uas.de

Abstract—This paper gives the evaluation of HTM when applied for image classification and the prediction of images. HTM (hierarchical temporal memory) is a cognitive learning system that aims to replicate the neocortex's operating principles. The primary objective of this paper is to examine the HTM parameters which results in the highest similarity for the image of same label/class and lowest similarity between images of different classes for MNIST (Modified National Institute of Standards and Technology) dataset. Another objective is to provide the prediction code for image classification so that after learning, system can classify the image based on training dataset.

Keywords—Hierarchical Temporal Memory (HTM), Image classification, Sparse Distributed Representation (SDR), Prediction code, Local Area Density, Potential Radius, Local/Global Inhibition, Spatial Pooler (SP).

I. INTRODUCTION

Image classification has remained a challenge that puts a system's intellect to recognize the interpretation of visual information in an image and construct a model that can store such information to the test. The system must be able to extract crucial feature information that, when paired with information learned during training, allows the system to recognize the item in the image. As a result, picture categorization has the ability to make use of effective knowledge representation strategies. In this paper, the learning technique of Hierarchical temporal memory (HTM) is utilized.

Hierarchical temporal memory (HTM) is a machine learning technique inspired by the neocortex and developed to learn and anticipate sequences. It should be able to generate generalized representations for similar inputs in its empirical way. HTM depicts some of the neocortex's structures and functions at a high level. A HTM network is developed as a tree-shaped hierarchy. Although uncommon, multiple parents per node are allowed, therefore letting the receptive fields of the parents overlap. Its organization is similar to that of cortical mini columns, in which an HTM region is made up of numerous columns, each of which contains many cells. A level is made up of one or more regions. To build the whole network, levels are stacked hierarchically in a tree-like structure. Synapses are used to make connections in HTM, with both proximal and distal synapses being used to produce feedforward and adjacent connections, respectively as represented in Figure 1.

Multiple HTM networks can be mixed and matched. Efficiency is a benefit of hierarchical organization. Because patterns learnt at each level of the hierarchy are reused when combined in innovative ways at higher levels, it dramatically decreases training time and memory usage. Given the statistics of the input and the number of resources available, HTMs automatically learn the best feasible representations at each level. If you give a level greater memory, it will create representations that are larger and more sophisticated, which implies fewer hierarchical levels will be required. If you allot less memory to a level, it will construct smaller and simpler representations, which may necessitate the use of more hierarchical levels. In HTM Input temporal data generated from various data sources is semantically encoded as a sparse array called as sparse distributed representation (SDR). This encoded array is subjected to a procedure known as spatial pooling, which normalizes/standardizes the input data from diverse sources into a sparse output vector or mini-columns (columns of pyramidal neurons) with a defined size and sparsity.

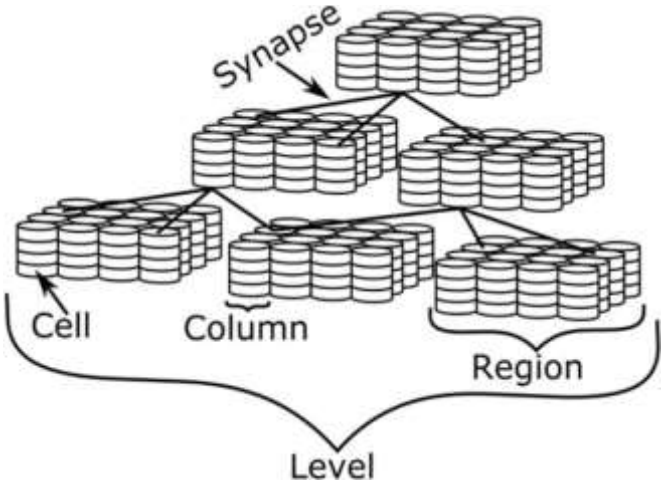


Figure 1: Depiction of HTM, showing the various levels of detail Source [1]

II. SPATIAL POOLER MODEL

The Spatial Pooler in an HTM-System which receives data from an encoder and sends it to the Temporal Memory. The primary goal is to convert the binary data into a semantically similar SDR with fixed sparsity. It is made up of a series of

mini columns. A mini-column is made up of a group of HTM-Neurons with the same receptive area.
The receptive field specifies which input bits a mini column can use. For each new input, the SP algorithm iterates through three phases after initialization:

1. Overlap calculation with Boosting
2. Inhibition
3. Learning by persistence adaption.

III. METHODOLOGY

Image classification project has been implemented previously in C# .NET Core in Microsoft Visual Studio 2022 IDE (Integrated Development Environment). The goal of this project is to implement a program that uses the existing solution as a library and find the spatial pooler parameter values which influence the learning of images resulting in the best correlation matrix and can also predict any input image based on the learning in HTM as shown in Figure 2.

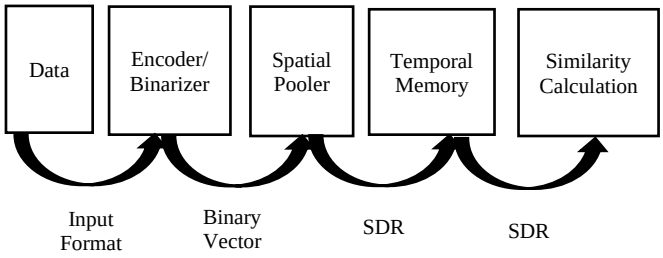


Figure 2: Overview of a typical HTM System

This section describes the reference to the methodology, which is already implemented for image classification, as it is used in this project for learning of images and thus for further experiments. The dataset used in this project is MNIST images dataset. It is a large database of handwritten digits that contains 60,000 training images and 10,000 testing images and is commonly used for training various image processing systems, a sample from MNIST dataset is shown in Figure 3. The database is also widely used for training and testing in the field of machine learning.

Figure 3: Sample images from MNIST dataset [2]

A. Encoder:

An HTM develops from the data that is given to it, as well as the presentation of that data. An encoder converts arbitrary input into a format that an HTM can understand. This input must be always an SDR. The activation state ('0' or '1' for inactive and active, respectively) of the columns from the previous area in the HTM is represented by each bit in the SDR. This input is then used as the feedforward input for the HTM's next area as shown in Figure 4.

Below example shows the binarization of the MNIST dataset images.

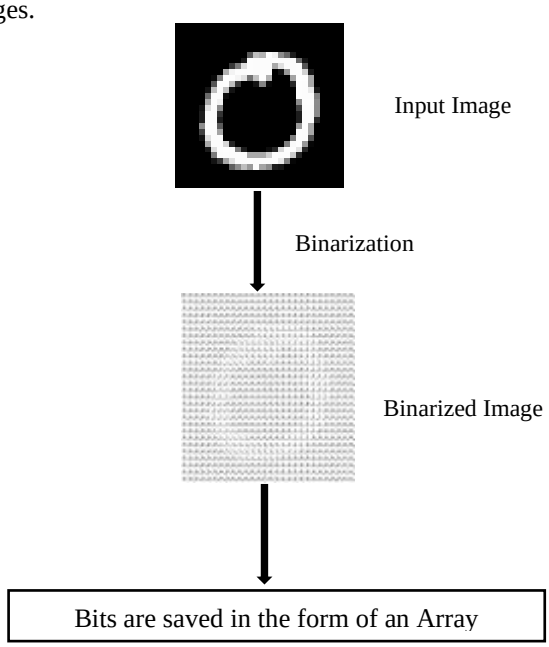


Figure 4: Image Binarization Process

B. Spatial Pooler:

The Spatial Pooler transforms input patterns into SDRs in a continuous online fashion. The HTM temporal memory learns temporal sequences of these SDRs and makes predictions for future inputs. The spatial pooler implies pools or clusters data in the spatial dimension. Each pattern that appears at the input during the spatial pooler's learning process is compared to the database of other patterns.

C. Sparse Distributed Representation:

In the HTM, SDR is an effective information organization system. Sparse means that only a tiny percentage of the big, interconnected cells are active at any given time. "Distributed" denotes that active cell are dispersed throughout the region and will be used to depict the region's activity. Because the binary representation is more biologically reasonable and extremely computationally efficient, HTM considers the binary SDR converted from a specific encoder. Even though the number of possible inputs exceeds the number of possible representations, the binary SDR does not result in a practical loss of information due to the following critical features of the SDR.

The HTM methods can be set with a variety of parameters and picking the proper parameters for investigations is crucial. Table 1 shows a list of Spatial Pooler parameters with their default values that are widely utilized in HTM investigations. Each of these variables influences the performance of HTM on its own; however, we will concentrate on the effect of potential radius and local area density, Global/Local Inhibition and NumActiveColumns PerInhArea.

The initialization of numerous parameters, which are defined by class htmconfig (HTM configuration), is the first and most crucial step in using any HTM configuration. Serialize the default configuration settings to the htmconfig.json file after initializing. The changed set of settings are loaded every time the user launches the HTM image categorization application. Below is the table consisting of all the HTM parameters which effect the classification of the Images.

Table 1: Default values of HTM parameters

Parameters	**Default value**
NumActiveColumnsPerInhArea	-1.0
MaxBoost	10.0
ColumnDimensions	(32,32)
CellsPerColumn	16
InputDimensions	(100,100)
ActivationThreshold	10
LearningRadius	10
MinThreshold	9
GlobalInhibition	false
LocalAreaDensity	0.1
DutyCyclePeriod	10
SynPermTrimThreshold	0.05
PotentialPct	0.5
StimulusThreshold	0.0
SynPermInactiveDec	0.01
SynPermActiveInc	0.1
SynPermConnected	0.1
DutyCyclePeriod	10

D. Similarity Calculation of SDRs

After training of images in each learning label is completed, similarity is calculated between active columns of input vectors and output SDRs of each image and a similarity matrix is generated to depict the similarity between images of same label (folder) called as **Micro Similarity** and between images of different labels called as **Macro Similarity.**

This is done by using CalculateSimilarityMatrix method from MathHelper class in the API. For example, images of all '0' are compared with each other and also compared with images of other MNIST digits like '1' and '9'.

The correlation matrix inside of the same image class defines the micro-accuracy and the correlation matrix between different image classes is called macro-accuracy. Further, the similarity data of all experiments is saved in a CSV file (Correlation.csv) which can be read by excel to analyse the best HTM parameters for image classification and prediction.

IV. EXPERIMENT

In this section, we have described different cases of experiments done on images of handwritten digits obtained from MNIST dataset. The experiments are performed to analyze the variation in similarity on changing the HTM parameters. Later, experiments are also performed to check the accuracy of image prediction after obtaining the best similarity matrix. The result of all experiments presented in this work is done by using local inhibition.

Learning Phase: Figure 5 shows the flow of image training steps in HTM image classification system.

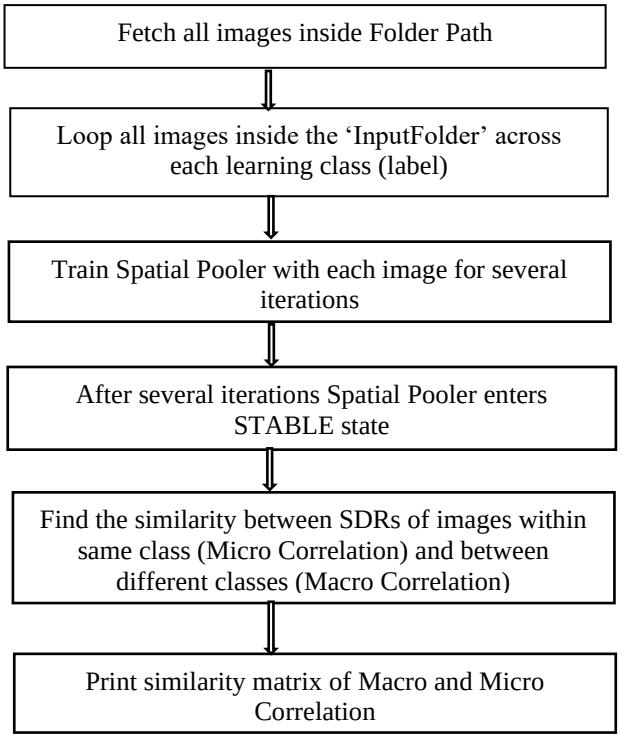


Figure 5: HTM Image Training Process Flow

For training process, we have taken MNIST images of dimensions 28x28 pixels. Initial experiments were performed with default parameters from htmconfig.json file. Every image is trained in several iteration steps specified by the argument iteration Steps. The images are trained until the spatial pooler enters a stable state which is controlled by the class HomeostaticPlasticityController (HPC). Its goal is to place the Spatial Pooler in a new-born state at the start of the learning process. The boosting is very active at this point, but the spatial pooler is instable. Once the SDR generated for each input becomes stable, the HPC will fire an event that notifies the code that spatial pooler is stable. The minimum number of cycles (iteration Steps) are automatically calculated by trainingImagesPath.Length × newBornStageIterations.

During the learning process the number of cycles taken by spatial pooler to become stable were between 90 to 380 depending on the number of images in different learning classes and on the HTM parameters given in the htmconfig.json file. In these experiments we have analysed how the classification of image is influenced by changing the spatial pooler parameters, **local area density** and **potential**

radius on images of same class and between images of different class. Local area density defines the number of active columns within a local inhibition area, we specify the desired density of active columns by changing **LocalAreaDensity** parameter in the htmconfig.json file. We have conducted experiments by varying this parameter between 0.1 to 1, keeping potential radius constant. Potential radius determines the extent of the input that each column can potentially be connected to. This can be thought of as the input bits that are visible to each column. This number determines how far a column's effect spreads over the HTM layer. We have conducted experiments by varying the parameter **PotentialRadius** in the htmconfig.json file by selecting the values from among 1,10,20 and 30.

Prediction Phase: The process followed for prediction is similar like learning phase but the difference here is that the flag learn is set to false state. After the learning phase is completed and similarity matrix of Macro and Micro correlation is generated for all the classes the SDR of the image to be predicted is compared with each of the SDRs of the images learned during the training phase and similarity percentage is calculated using *CalculateSimilarity* function.

Further the best match is searched in correlation matrix and displays the image class (label) with which it is matching including the maximum, minimum and average percentage of similarity. To make the predictions even more accurate, we have also included a ***similarity threshold*** comparison parameter in the prediction code which decides based on the HTM parameters whether the image to be predicted belongs to any image class used in the training process or not. So, for prediction of any image our prediction code involves two steps mentioned below:

In the first step, we calculate the percentage similarity between the SDR of the image to be predicted with SDRs of training images.

The we compare the highest similarity obtained in step 1 with the ***similarity threshold value*** and of the highest similarity is above the threshold value then only the prediction will classify that the image belongs to the class with which it has the highest similarity. **If the highest similarity is lower than the similarity threshold value, then the prediction code will classify that the image does not belong to any image class used in the training phase.**

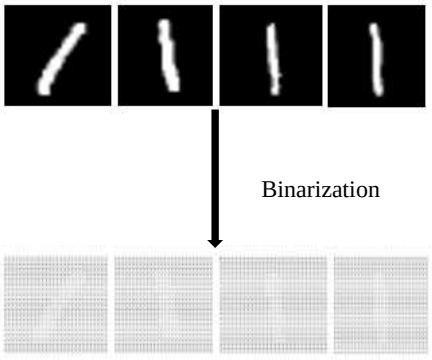
We calculated the similarity threshold value after conducting many training experiments at various values of **LocalAreaDensity** and **PotentialRadius** and observed the similarity between images of different classes (Macro Correlation) and found that images of different classes are also similar up to a certain minimum percentage even in the best case of micro correlation in which there is no overlapping between minimum of micro correlation and maximum of macro correlation in the similarity matrix.

Throughout the experiments, the input dimensions are taken as 28 x 28 and columns dimensions as 64 x 64.

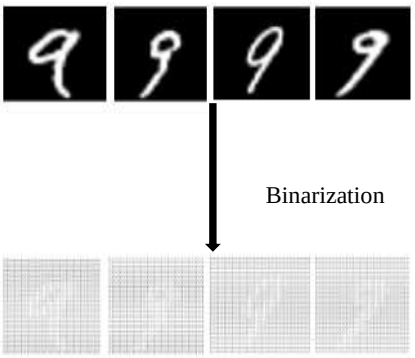
V. RESULT AND DISCUSSION

For our experiments in the training phase, we used the images of digits 0,1 and 9 from the MNIST dataset. Four random images of each of these digits were taken as shown in Figure 2. and three sub folders were created with names same as the digit name inside the **InputFolder.**

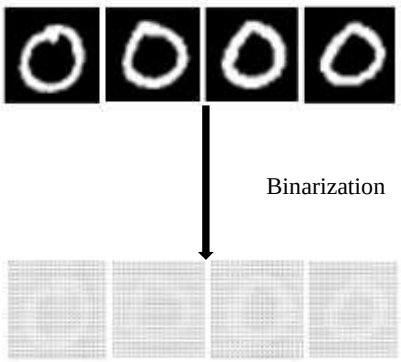
Label Name: 'One'



Label Name: 'Nine'



Label Name: 'Zero'



From all the experiments conducted in the training phase, we analysed the similarities between images of same class and between images of different class and have explained our findings in the below given cases:

Similarity between Images of Same Label

1. *Case 1: Micro Similarity of Images in Label 'Nine'*

In this case we analyzed variations in similarity between images of same class and observed the micro similarity

between images of digit 9. Table 2 shows the parameter values which we used in this case.

Table 2: Parameters used in Initial Experiment 1

Parameters	Default Value
InputDimensions	(28,28)
ColumnDimensions	(64,64)
PotentialRadius	1
GlobalInhibition	False
LocalAreaDensity	0.1

LocalAreaDensity was varied from 0.1 to 1, keeping potential radius and other parameters constant. Later more experiments were also conducted for the same images of digit 9 at potential radius 10,20 and 30 respectively at various local area density values.

Figure 6 shows variation in similarity between images of digit 9 at various local area and density and potential radius.

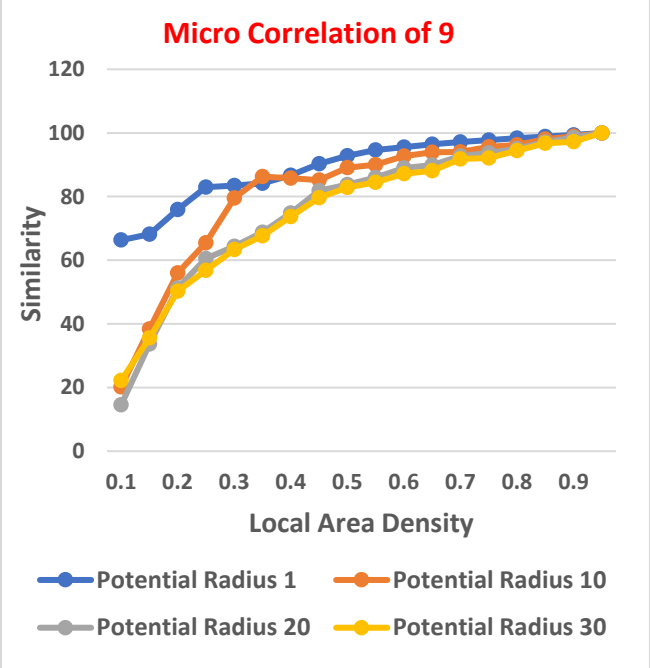


Figure 6: Micro Similarity of Label 9 with local area density at potential radius 1,10,20 and 30

Local area density value is varied from 0.1 to 1.0 in steps of 0.05 and the results are as follows. The similarity is always 100% irrespective of whatever the image is taken when local area density is above 1.0(e.g., 1, 2, 3, 10... so on) for any value of potential radius.

Table 3 shows how the similarity changes with the change in local area density, when the potential radius is 1, 10,20 and 30 in case of images of same class (micro similarity).

Table 3: Variation in micro similarity with local area density for potential radius = 1,10,20,30

Local Area Density	Output Similarity at Potential Radius 1	Output Similarity at Potential Radius 10	Output Similarity at Potential Radius 20	Output Similarity at Potential Radius 30
0.1	66.32	20.2	14.51	22.19
0.15	68.13	38.4	33.71	35.61
0.2	75.95	56.03	51.27	50.25
0.25	82.92	65.43	60.54	56.82
0.3	83.5	79.59	64.36	63.38
0.35	84.19	86.3	68.77	67.66
0.4	86.64	85.8	74.78	73.71
0.45	90.35	85.23	81.88	79.68
0.5	92.82	89.11	83.78	82.86
0.55	94.63	90.06	85.98	84.43
0.6	95.56	92.7	88.97	87.15
0.65	96.48	93.9	89.95	88.1
0.7	97.16	94.04	92.69	91.87
0.75	97.74	95.67	93.78	92.12
0.8	98.33	96.21	95.05	94.44
0.85	98.88	98.01	97.18	96.79
0.9	99.37	99.01	98.53	97.28
1	100	100	100	100

2. *Case 2: Micro Similarity of Images in Label 'One'*

In this case we analyzed the variations in micro similarity of another label 'One'. Like the previous case keeping other spatial pooler parameters same, we did experiments by changing local area density and potential radius.

Local area density value is varied from 0.1 to 1.0 in steps of 0.05 and the results are as follows. The similarity is always 100% irrespective of whatever the image is taken when local area density is above 1.0(e.g., 1, 2, 3, 10... so on) for any value of potential radius.

Figure 7 shows variation in similarity between images of digit '1' at various local area and density and potential radius.

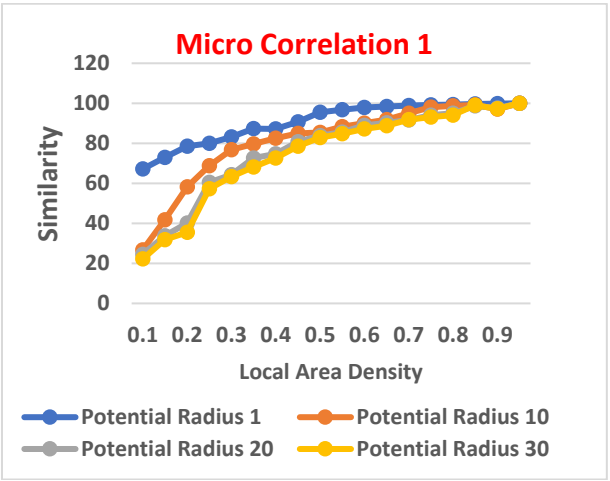


Figure 7: Micro Similarity of Label 'One' with local area density at potential radius 1,10,20 and 30

Table 4 shows how the similarity changes with the change in local area density, when the potential radius is 1, 10,20 and 30 in case of images of same class (micro similarity).

Table 4: Variation in micro similarity of Digit '1' with local area density for potential radius = 1,10,20,30

Local Area Density	Output Similarity at Potential Radius 1	Output Similarity at Potential Radius 10	Output Similarity at Potential Radius 20	Output Similarity at Potential Radius 30
0.1	67.15	26.74	24.55	22.19
0.15	72.98	41.67	33.71	31.89
0.2	78.48	58.29	40.14	35.43
0.25	79.92	68.79	60.54	57.34
0.3	83.1	76.78	64.36	63.38
0.35	87.35	79.63	72.59	68.2
0.4	87.24	82.57	74.78	72.62
0.45	90.72	84.88	80.71	78.54
0.5	95.48	85.44	83.78	82.86
0.55	96.79	88.23	85.62	84.74
0.6	97.95	89.98	88.97	87.11
0.65	98.38	91.87	90.19	88.78
0.7	98.85	94.97	91.69	91.87
0.75	99.16	97.99	93.98	93.11
0.8	99.4	98.74	95.05	94.03
0.85	99.59	99.11	98.65	98.98
0.9	99.74	97.15	97.53	97.28
1	100	100	100	100

3. *Case 3: Micro Similarity of Images in Label 'Zero'*

In this case we analyzed the variations in micro similarity of another label '0'. Like the previous case keeping other spatial pooler parameters same, we did experiments by changing local area density and potential radius. Local area density value is varied from 0.1 to 1.0 in steps of 0.05 and the results are as follows. The similarity is always 100% irrespective of whatever the image is taken when local area density is above 1.0(e.g., 1, 2, 3, 10... so on) for any value of potential radius. Figure 8 shows variation in similarity between images of digit '0' at various local area and density and potential radius.

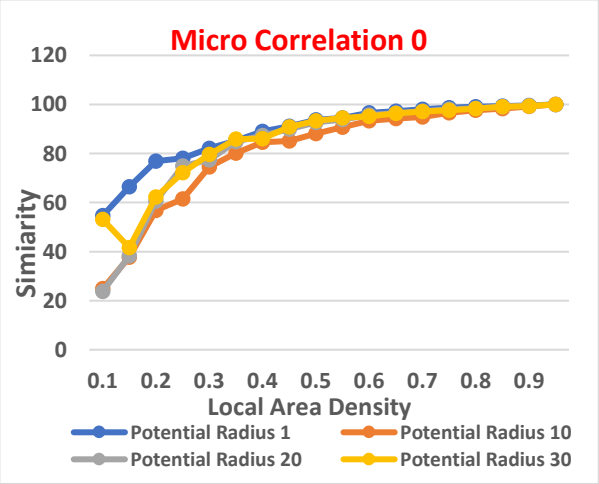


Figure 8: Micro Similarity of Label 'Zero' with local area density at potential radius 1,10,20 and 30

Table 5 shows how the similarity changes with the change in local area density, when the potential radius is 1, 10,20 and 30 in case of images of same class (micro similarity).

Table 5: Variation in micro similarity of Digit '0' with local area density for potential radius = 1,10,20,30

Local Area Density	Output Similarity at Potential Radius 1	Output Similarity at Potential Radius 10	Output Similarity at Potential Radius 20	Output Similarity at Potential Radius 30
0.1	54.64	24.88	23.79	53.17
0.15	66.46	37.76	38.27	41.65
0.2	76.86	56.72	60.56	62.31
0.25	78.09	61.43	74.9	72.22
0.3	82.05	74.53	77.46	79.57
0.35	85.26	80.15	84.89	85.9
0.4	89.07	84.52	87.3	86.05
0.45	91.08	85.15	89.97	90.89
0.5	93.67	88.15	92.67	93.31
0.55	94.47	90.75	93.87	94.54
0.6	96.58	93.3	95.64	95.11
0.65	97.22	94.22	96.4	96.3
0.7	98.05	94.91	97.2	97.1
0.75	98.64	96.63	98.01	97.65
0.8	99.09	97.68	98.47	98.32
0.85	99.38	98.28	99.1	99.01
0.9	99.54	99.24	99.4	99.34
1	100	100	100	100

***Discussion*:** From Case 1 to Case 3, we found that similarity between images of same class of digit increase with increase in local area density at any potential radius and similarity is 100% at local area density of 1 and above. And similarity is higher at potential radius 1 even at local area density of 0.1. Similarity starts from range of 55% to 65% when potential radius is 1 and local area density 0.1. From local area density range of 0.75 to 0.9 is the similarity at any potential radius reaches in the range of 93%-99%.

Similarity between Images of Different Labels

4. *Case 4: Macro Similarity 9 vs 1 at Potential Radius=1,10,20 and 30*

In this case we observed the effect on similarity between images in label 'Nine', with respect to images in labels 'One' potential radius 1,10,20 and 30. Local area density value is varied from 0.1 to 1.0 in steps of 0.05 and plots of micro similarity of digit '9 vs 1' was observed.

Error! Reference source not found. shows the extent to w hich images of digits in different labels are like each other at various values at various local area densities and potential radius.

Table 6 shows how the similarity changes with the change in local area density, when the potential radius is 1, 10,20 and 30 in case of images of different classes (macro similarity).

From **Error! Reference source not found.** and Table 6 we o bserved that at local area density 0.1, similarity between

images of labels 'Nine' and 'One' the similarity is only 46.68%. As the local area density is increased the macro similarity also increases and reaches 100% at local area density 0.1 which means as the local area density increases different objects appear to be similar.

Table 6: Variation in macro similarity of Digits 9 vs 1 with local area density for potential radius = 1,10,20,30

Local Area Density	9 vs 1 at Potential Radius=1	9 vs 1 at Potential Radius= 10	9 vs 1 at Potential Radius=20	9 vs 1 at Potential Radius=30
0.1	48.68	17.74	14.57	9.26
0.15	54.61	28.94	16.44	12.55
0.2	58.9	40.32	27.77	25.14
0.25	58.8	47.56	37.3	36.61
0.3	66.87	57.48	44.26	45.89
0.35	71.92	60.43	53.59	52.22
0.4	75.64	66.82	59.76	56.01
0.45	81.85	71.94	64.15	63.72
0.5	86.37	77.47	71.82	70.94
0.55	89.54	79.23	76.86	74.08
0.6	91.92	82.22	81.85	80.66
0.65	93.8	86.46	84.71	83.45
0.7	95.26	89.67	88.07	86.95
0.75	96.56	91.85	90.88	89.89
0.8	97.42	92.25	92.4	93.2
0.85	98.43	93.36	94.19	94.8
0.9	99.07	94.26	95.98	95.87
1	100	100	100	100

Discussion: Images of different digits are tested during the work. However, as an illustration, results of only two classes of digits Nine vs One are mentioned in this paper. After training, the number of active columns from SP for potential radius of 1 and local area density of 0.1 is 48.68%. For a potential radius value of 10 the similarity slightly decreases and stands at 17.74%. It further decreases for potential radius 20 to 14.57% When potential radius is 30, the output similarity is 9.26% for minimum value of local area density. As the density of active columns (local area density) increases, Nine and One appear to be similar.

Prediction Testing

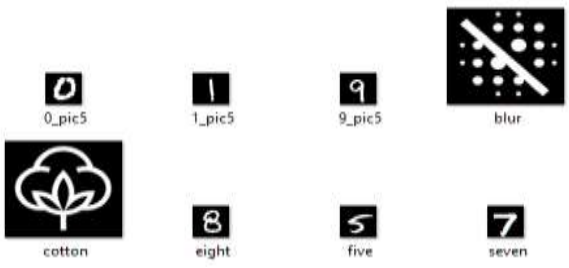


Figure 9: Prediction Images

After training, the model is used to make a prediction of different images. The active columns of the prediction method are compared with the SDRs (active columns) saved in a text file during Spatial Pooler training process.

Figure 9 shows the images and their names which were used for prediction and testing the quality of learning. For testing, different images of MNIST digits with class same as the classes used in training as well as other MNIST digits which were not used in training and two other random black and white images of objects were used as mentioned below:

- Different images with class same as class of images used in training **0,1** and **9**.
- Images of other classes of MNIST dataset which were not used in training **8,5** and **7**.
- Black and white images of random object which are not a part of MNIST dataset and not used in training **blur** and **cotton** as shown in Figure 9.

The similarity of each image in prediction is calculated with active columns from the training. After searching for the best match in the correlation matrix it gives the label name of the digit to which it has maximum similarity and displays the minimum, maximum and average similarity with other images classes used in the training. Prediction results for different type of images used for prediction are discussed in two cases given below:

Case 1: When prediction image belongs to any class which is used for training

In this case we conducted prediction tests for images belonging to MNIST digits '0', '1' and '9' as shown in Figure 7. However, in this paper we have taken only one of those tests to explain the results.

Image to be predicted:

Images used in the training:

Label 'Nine'

Label 'One'

Label 'Zero'

Figure 10 shows the output of training and prediction at **local area density 0.2** and **potential radius 10**, when images of MNIST digits 9,1 and 0 were used in training with each label 'Nine', 'One' and 'Zero' consisted of four images of their own class each. For prediction, we used an image of MNIST digit '0' and it displayed the prediction status. The similarity

of tested image is highest with images in training label ‘Zero’ and stands at 75.72% and it has a similarity of 58% and 53.11% with images in training labels ‘Nine’ and ‘One’ respectively. **The prediction status is correct for the selected local area density and potential radius.**

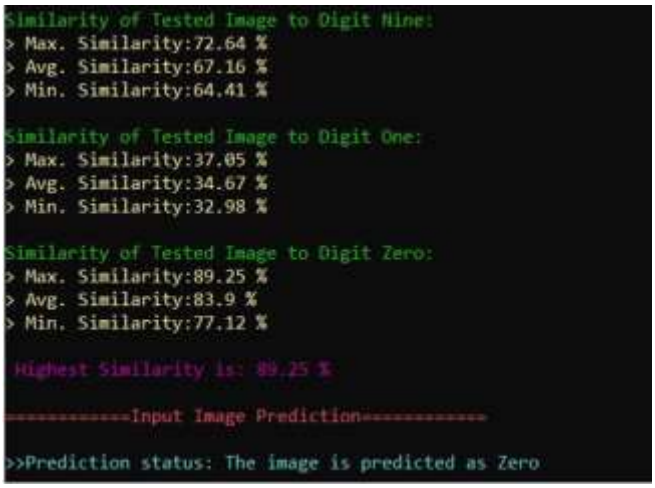


Figure 10: Output of Training and Prediction

Case 2: When prediction image does not belong to any class which is used for training

This case is an improvement of the prediction code which enables it to detect if the tested image does not belong to any image classes which were used for training.
Image to be predicted:

Images used in the training:

Label ‘Nine’

Label ‘One’

Label ‘Zero’

In this case we conducted prediction tests for images which do not belong to any MNIST digits which were used for training like image names ‘cotton’, ‘blur’, ‘eight’, ‘five’ and ‘seven’ as shown in Figure 9. However, in this paper we have taken only one of those tests to explain the results.

Figure 11 shows the output of training and prediction at **local area density 0.2** and **potential radius 10**, when images of MNIST digits 9,1 and 0 were used in training with each label ‘Nine’, ‘One’ and ‘Zero’ consisted of four images of their own class each. For prediction, we used an image of MNIST digit ‘8’ as shown in Figure 9 and it displayed the prediction status.

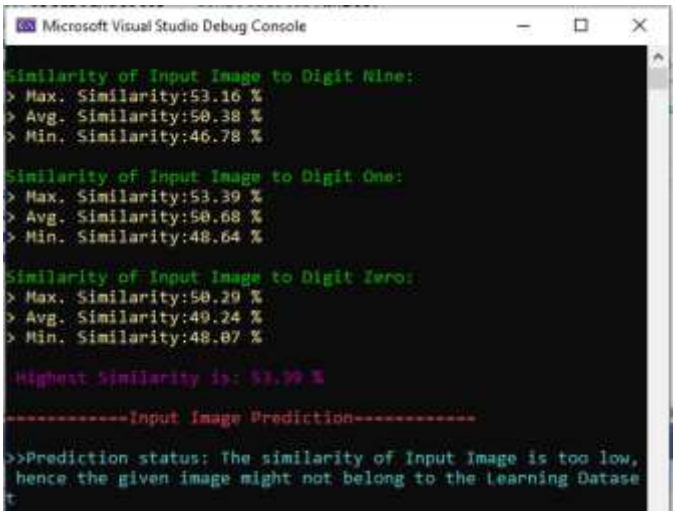


Figure 11: Prediction Result for Image ‘Eight’

The similarity of tested image is highest with images in training label ‘One’ and stands at 53.39% and it has a similarity of 53.16% and 50.29% with images in training labels ‘Nine’ and ‘Zero’ respectively. But even if the predicted input image has higher similarity with training label ‘One’, we know that we used image of digit ‘8’ for prediction and in this case the prediction result would be false if only the amount of similarity will be considered for prediction, so to avoid false predictions we incorporated a certain ‘**Similarity Threshold**’ in our prediction code in cases when the similarity of the image to be predicted is relatively low as compared to the similarity which we usually observe when images belong to the same class.

Discussion: In this case, though the image ‘Eight’ which we wanted to test has a similarity of 53.39% with images of training label ‘One’ but still this similarity is not sufficient to qualify it as an image of label ‘One’ as from experiments conducted during training phase at local area density 0.2 and potential radius 1, we found that images in label ‘One’ are similar to each other in the range of 65% to 85% which is much higher than 53.39%, so it displays the prediction status that ‘similarity of input image is too low’. The use of similarity threshold makes the prediction more efficient and prevents false predictions.

The threshold value used for similarity threshold criteria is selected after carefully observing the experiments conducted during the training phase and analyzing the similarity matrix for the usual similarity percentage ranges in case of micro correlation and macro correlation. We observed that at local area density 0.2 to 0.3 and potential radius 1 and 10, the similarity matrix shows no or very small overlapping between macro and micro similarity and there the similarity threshold was also selected at this local area density and potential radius.

As we know that the similarity gets effected with the change of local area density and potential radius, the similarity threshold also needs to be changed in the prediction code to get efficient and accurate results.

VI. Conclusion

In this paper, Neocortex API is adapted to the image classification problem with MNIST dataset. The results show how potential radius and local area density parameters influence the similarity. The similarity increases as the local area density is increased and images of same image class and different image class appears to be similar. From the results, we can see that for local area density value of 1, the image matches 100% irrespective of whatever the image is taken.

The experiments were also helpful in understanding how local area density and potential radius can influence the prediction of images based on training. We were able to get accurate predictions at local area density ranging from 0.2 to 0.3 and potential radius 1 and 10. We also incorporated a similarity threshold in the prediction code which **prevented false predictions** in case if the image to be predicted does not belong to training dataset. We also made the prediction code more efficient and time saving as it can predict multiple testing images in the same execution. We also improvised the prediction code to enable to fetch the absolute path and name of all testing images automatically without any manual input to the code.

From the discussion in the above section, we can conclude that if large number of columns remain active within a local area inhibition, the similarity between the images will be more and they appear to me similar. So, for the different class of images not to appear similar, the local area density must be as low as possible.

VII. References

[1] J. Mnatzaganian, E. Fokoué and D. Kudithipudi, "frontiers in Robotics and AI," [Online]. Available: https://www.frontiersin.org/articles/10.3389/frobt.2016.00081/full.

[2] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/MNIST_database.

[3] D. Dobric, "Github," [Online]. Available: https://github.com/ddobric/neocortexapi-classification.

[4] G. D, "How the brain might work: a hierarchical and temporal model for learning and recognition," Stanford University, Stanford, CA, USA, 2008.

[5] J. Hawkins, S. Ahmad and D. Dubinsky, "Hierarchical Temporal Memory including HTM Cortical Learning".

[6] "Numenta," [Online]. Available: https://numenta.com/resources/biological-and-machine-intelligence/spatial-pooling-algorithm/.

[7] J. Hawkins and C. Maver, "Biological and Machine Intelligence," BAMI, 2019. [Online]. Available: https://numenta.com/assets/pdf/biological-and-machine-intelligence/BaMI-HTM-Overview.pdf.