

Analyse Image Classification for Simple Shapes Dataset

Bolutife Oluyinka Adisa
bolutife.adisa@stud.fra-uas.de

Abstract— Several techniques for dealing with image classification have emerged over the years, in line with the introduction of various technological advancements, however, Hierarchical Temporary Memory (HTM) was utilized as Artificial Intelligence method in the case of this project. Hierarchical temporary memory which evolved from Neocortex neuroscience and is now one of the most widely used applications for image recognition and anomaly detection was utilized in this paper to conduct a series of tests to determine the optimal parameter combination for achieving good and accurate similarity between simple images (circle, square, star, and triangle), as well as an extra function for predicting a simple input image (circle, square, star and triangle). We were able to demonstrate excellent effectiveness and accuracy through several experiments and found the right optimal parameter combination

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 and recognition have proved to be difficult tasks in artificial intelligence for several decades. Image classification refers to the categorizing the images into one of a number of predefined classes. The Image Classification model must be able to extract critical feature information that will be learned during training phase, and store the information for further test, henceforth for the classification the trained data is used to categorize the images. In this paper Hierarchical Temporal Memory (HTM) is used as learning algorithm along with the C# as programming language.

Hierarchical temporal memory (HTM) [1], is a machine learning algorithm that was inspired by the neocortex, centre for higher brain functions of human brain, and designed to learn sequences and make predictions on spatiotemporal data. In its idealized form, it should be able to produce generalized representations for similar inputs. HTM can serve as a tool for intelligent data processing in edge computing devices.

HTM is divided into two parts, HTM Spatial Pooler (HTM SP) and HTM Temporal Memory (HTM TM). The HTM SP forms a spatial pooling on the Sparse Distributed Representation (SDR) of the input data by performing a feature encoding which is useful for visual data processing and

classification problems, whereas The HTM TM is responsible for the learning and processing of temporal patterns and can be used for the prediction taking into account previous experiences. The HTM network has a tree shaped hierarchical structure, as shown in Figure. 1. Each level of HTM is made up of distinct areas with columns, each of which is made up of cells. In HTM, the columns represent neurons. The connections between the columns and the input space are made through a dendritic segment with multiple synapses. Each synapse has a certain weight called synaptic permanence.

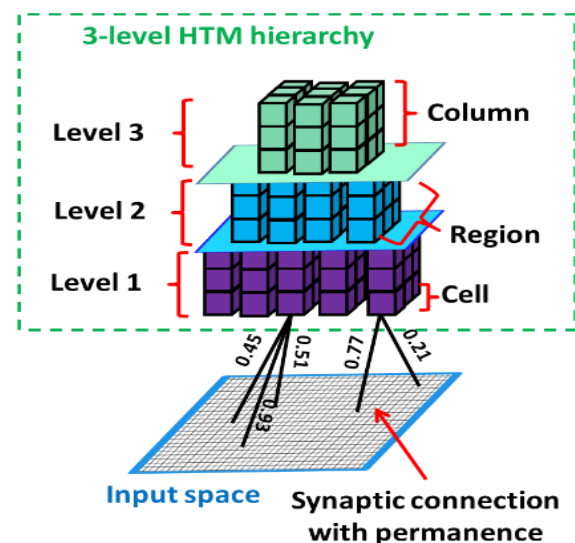


Figure 1: The three-level hierarchical structure of HTM

II. METHODOLOGY

The Image Classification project was implemented [2] in C# .NET Core in Microsoft visual studio 2022 Integrated Development Environment (IDE). To achieve the desired classification result, the project utilizes the NeoCortex API [3] for implementing the HTM. The goal of this project is to implement a program that uses the existing NeoCortex API solution as a library to find the best parameter values for the spatial pooler which influence the training of the different

images (simple shapes), thereby resulting in the identification of a best correlation matrix. As a result, the model should be able to predict any input image belonging to the class of chosen simple shapes dataset which were used during the training phase.

This section outlines the methodology employed in a previous image classification project [3], which is the same methodology that will be used to train images in the current project. After the training process is completed and the optimal correlation matrix is determined, the model will have the ability to predict the class of any input image, as depicted in Figure 2.

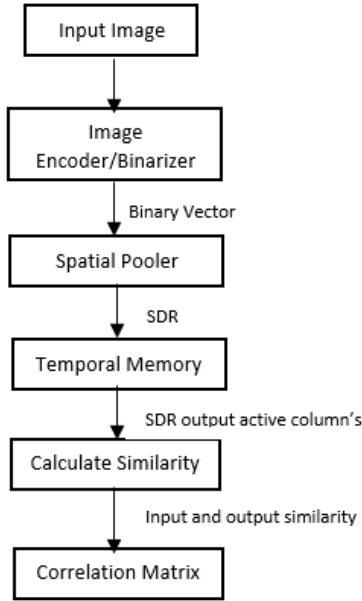


Figure 2: Methodology of the HTM image classification

A. Project structure

The image classification project includes two internal classes, as well as a .json file containing parameters for the Spatial pooler. The classes, which are shown in Figure 3, are called *HelpersTemp.cs* and *Experiment.cs*. Additionally, the project includes a .json file named *htmconfig.json*.

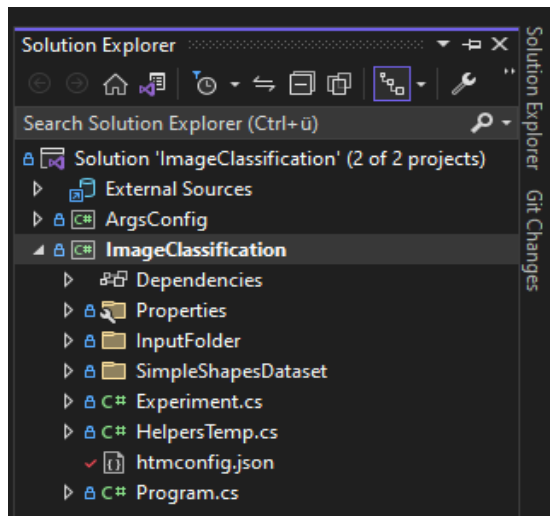


Figure 3: Project structure

B. Input Dataset:

When selecting a dataset for image analysis, various factors need to be considered to ensure optimal results. These factors include the position of the object within the images, the orientation or angle of the object within the images, and the size of the object within the images. The selection of these factors is dependent on the depth of analysis required. For instance, if an object with similar size [4] and orientation is used in multiple images, there is a higher probability of achieving good results. However, in our study, images with varying object sizes, positions, and orientations were selected to assess their impact on the results and minimize any such impact to obtain good results as shown in figure 4.

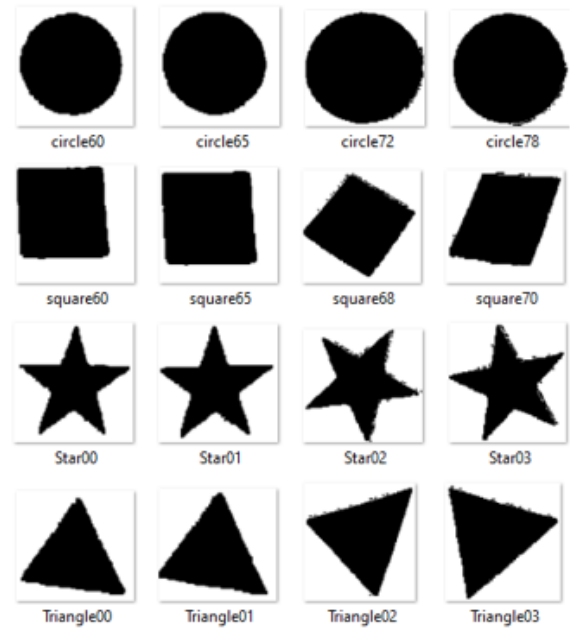


Figure 3: 64 x 64 pixels input dataset

C. Image Encoder:

The Image Encoder is responsible for reading image data and converting it into binary form by generating an integer array. To accomplish this, the Image Encoder calculates the pixel values of the image and uses a threshold value to segment the image into binary form. Typically, in grayscale images, white pixels are assigned a value of one, while dark pixels are assigned a value of zero, as illustrated in Figure 5.

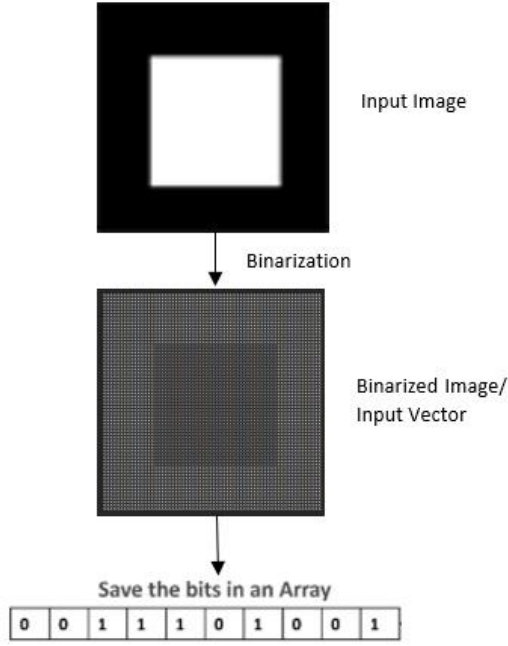


Figure 5. Image Binarization Process

D. Spatial Pooler and Sparse Distributed Representation:

In order to comprehend and investigate the HTM software for conducting image classification tests, it is essential to understand the program's core, which involves the Spatial Pooler. The Spatial Pooler Learning Method is an unsupervised machine learning algorithm inspired by the neocortex that is used to learn spatial patterns. It receives input in the form of Sparse Distributed Representations (SDRs) generated by the encoder (binarized data) and generates a set of active columns. The Spatial Pooler groups similar spatial patterns represented as activated neurons into highly sparse representations of cortical micro-columns. SDRs are achieved when the number of active mini-columns in each HTM zone is limited to between two and six percent of the total mini-columns [4].

Parameters	Default value
InputDimensions	(64,64)
ColumnDimensions	(32,32)
InhibitionRadius	1
PotentialRadius	30
PotentialPct	0.5
GlobalInhibition	False
LocalAreaDesity	0.1
NumActiveColoumnPerInhArea	-1
StimulusThreshold	0.0
SynPermInactiveDec	0.01
SynPermActiveInc	0.1
SynPermConnected	0.1
DutyCyclePeriod	10
MaxBoost	10

Table 1: Default values of Spatial Pooler parameters

E. Calculation of Similarities

During the learning phase of image processing, after each image has been trained, the next step involves computing the similarity between the Sparse Distributed Representations (SDRs) of each trained image. Two distinct methods are used to measure the similarity between the active columns of an image's SDR: Micro similarity and Macro similarity. Micro similarity corresponds to the similarity between images with the same shape, whereas Macro similarity refers to the similarity between images with different shapes. For example, to calculate Micro similarity for all circle-shaped images, each circle-shaped image is compared to itself. On the other hand, to obtain Macro similarity between circle and Square shaped images, a circle-shaped image is compared to a Square-shaped image. The resulting values of Micro and Macro similarity are then combined to construct a similarity matrix, which is used to evaluate the accuracy of the learning process and to predict the class of the image..

III. EXPERIMENT

This section discusses a series of experiments that have been conducted with the aim of exploring how image similarity varies with changes in different parameters.

The overarching objective of these experiments is to determine the optimal set of parameters that can yield satisfactory results and accurately predict the shapes of the images in the dataset, which are simple shapes such as squares, rectangles, circles, triangles, and stars. For each shape category, there are four images, each varying in terms of size, position, and orientation. The similarity of the images is measured with respect to different parameters, and the resulting values are plotted on a graph to visualize the variations.

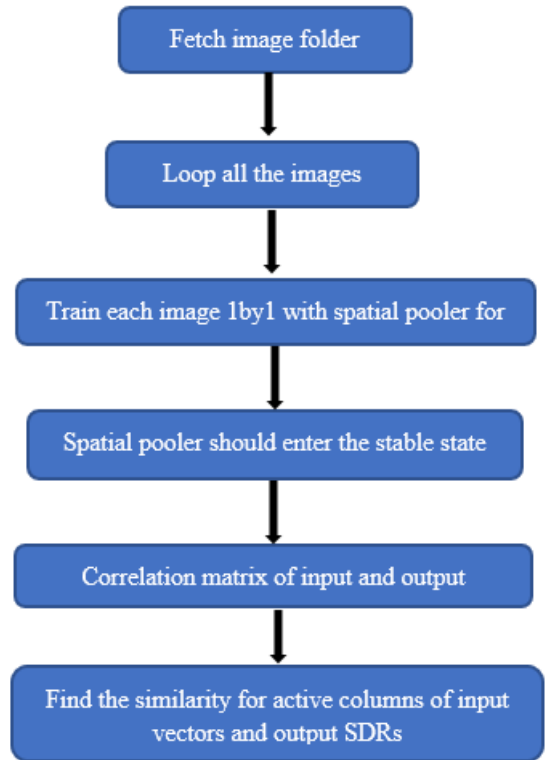


Figure 6. Learning Phase Flow Chart

Learning Phase:

Image (circle, square, triangle, and star) dimensions of 64x64 were taken, then these images were binarized (0 for the object in the image and one for the background), these binarized images are now the input. In the initial phase of the experiment, the same instance of spatial pooler was used with the default parameters as shown in Table 1, from the *htmconfig.json* file.

The argument *iterationsteps* specifies how many iteration steps are used to train each image. The images are trained until the spatial pooler achieves a stable state, which is managed by the *HomeostaticPlasticityController* class. When the learning process begins, the purpose of the *HomeostaticPlasticityController* is to place the spatial pooler in a new-born state. At this moment, the boosting is highly active, but the spatial pooler is unstable. The HPC will emit an event that alerts the code that the spatial pooler is stable after the SDR generated for each input becomes stable. $trainingImagePathLength \times newBornStageIteration$ automatically calculates the minimum number of cycles.

Local area density: density of active columns inside of local inhibition. It should be more than zero and less one if it is less than 0 the *numActiveColumnsPerInhArea* will be used.

Potential radius: this defines the radius in number of input cells visible to columns cells. It is important to choose this value, so every input neuron is connected to at least a single column.

GlobalInhibition: If true, then during inhibition phase the winning columns are selected as the most active columns from the region. Otherwise, the winning columns are selected with respect to their local neighbourhoods.

numActiveColumnsPerInhArea: An alternate way to control the density of the active columns. If *numActiveColumnsPerInhArea* is specified then *localAreaDensity* must be less than 0, and vice versa.

Micro: this is the similarity between two images in the same category e.g., circle to circle or Square to Square

Macro: this is the similarity between two images from different category e.g., circle to Square or Square to triangle

Prediction Phase:

After the completion of the learning phase and the generation of a similarity matrix with Micro and Macro correlations for all classes, the prediction process involves setting the learn flag to a false state. The input image that needs to be predicted is then converted to binary and passed through the spatial pooler, resulting in an SDR. The CalculateSimilarity Method is then used to compare the SDR of the input image with the SDRs of each trained image. Afterwards, the correlation matrix is searched to find the best match, and the prediction image's similarity with all trained shapes is displayed, including the maximum, average, and minimum percentage of similarity. The flowchart depicting this process is shown in Figure 7.

The first step of the process involves calculating the percentage similarity between the SDR of the input image and the SDRs of the trained images.

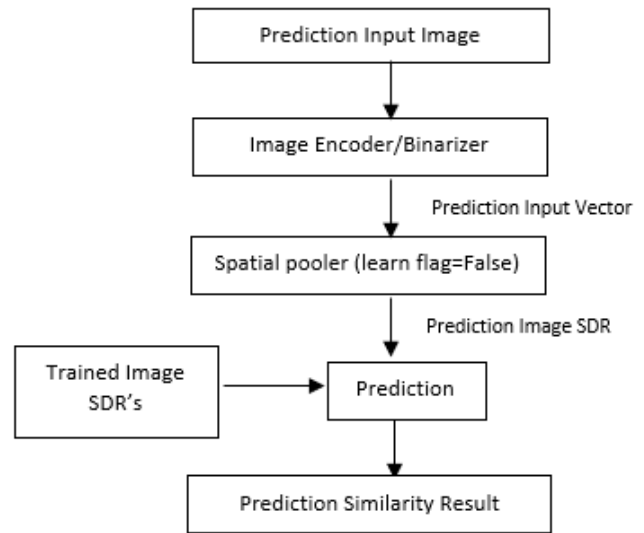


Figure 7. Prediction Phase Flow Chart

IV. RESULT AND DISCUSSION

In this section, the results of the above experiments conducted will be discussed.

In this experiment, an image dimensions of 64x64 and 32x32 column is used for over 100 experiments conducted. The entire learning process took a minimum of 55 cycles and maximum of 204 cycles depending on the configured parameters in the *htmconfig.json* file. The result of the experiment suggests the optimal local area density and potential radius that give optimal similarity between the images.

Case 1: Micro Similarity between images

From the input dataset mentioned in the Figure 4, considering the two square shape images as shown in figure 8 and comparing the similarity between them for various values of parameters.

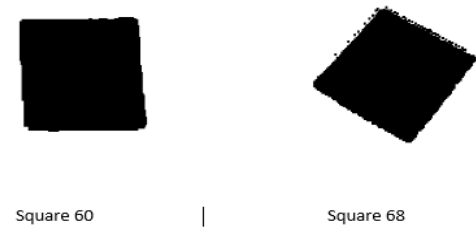


Figure 8. Input images for micro similarity calculation

Figure 9 and Table 3 shows the result we get on the similarity between two squares, when we run different experiments on various local area density and potential radius values.

Local Area Density	Output Similarity between Square60 & Square68				
	Potential Radius = 1	Potential Radius = 5	Potential Radius = 10	Potential Radius = 20	Potential Radius = 30
0.1	50.8	20.32	31.61	75.44	27.7
0.2	72.55	28.39	41.02	84.91	30.14
0.3	81.96	41.65	48.25	78.09	42
0.4	88.03	48.97	60.77	81.49	49.19
0.5	95.74	81	79.1	90.56	65.75
0.6	93.88	91.34	81.38	85.71	77.82
0.7	97.69	97.44	86.93	89.34	82.86
0.8	97.69	98.18	93.08	90.48	89.29
0.9	97.69	99.65	96.05	98.69	97.88
1	100	100	100	100	100

Table 3: Micro Similarity between Square60 & Square68

Local Area Density	Output Similarity between Square60 & Triangle00				
	Potential Radius = 1	Potential Radius = 5	Potential Radius = 10	Potential Radius = 20	Potential Radius = 30
0.1	62.31	23.4	25.2	49.37	20.7
0.2	68.06	32.14	27.65	59.85	21.58
0.3	75.95	42.85	30.42	65.23	27.3
0.4	88.03	49.08	40.85	70	43.55
0.5	95.8	82.24	80.87	71.26	46.46
0.6	95.55	92.58	83.39	73.92	60.02
0.7	98.66	97.56	88.82	78.39	80.98
0.8	98.66	97.94	94.92	85.43	83.9
0.9	98.66	99.06	98.43	93.82	99.08
1	100	100	100	100	100

Table 4: Macro Similarity between Square60 & Triangle00

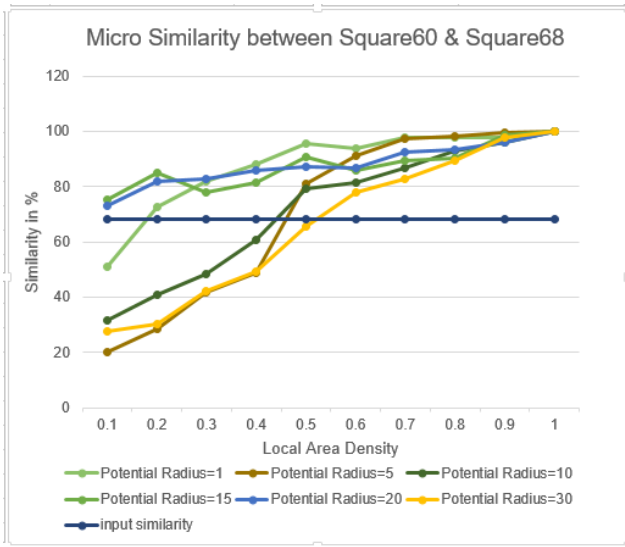


Figure 9. Graph depicting the change in micro similarity for change in PotentialRadius and LocalAreaDensity

Since our goal is to find an optimal value for the local area density that maximizes both micro and macro similarity, it can be observed in Figure 9 that the micro similarity increases with increasing local area density. The optimal value for local area density would be the one that maximizes both micro and macro similarity simultaneously.

Case 2: Macro Similarity between images

Macro similarity is a measure of similarity between two different classes of images. In this case, the similarity between a square image and a triangle image is being considered as shown in Figure 10. The similarity values obtained for this comparison are presented in Table 4 and also visualized in Figure 11.

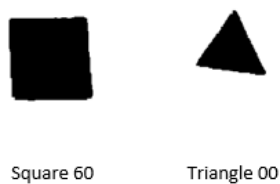


Figure 10. Input images for macro similarity calculation

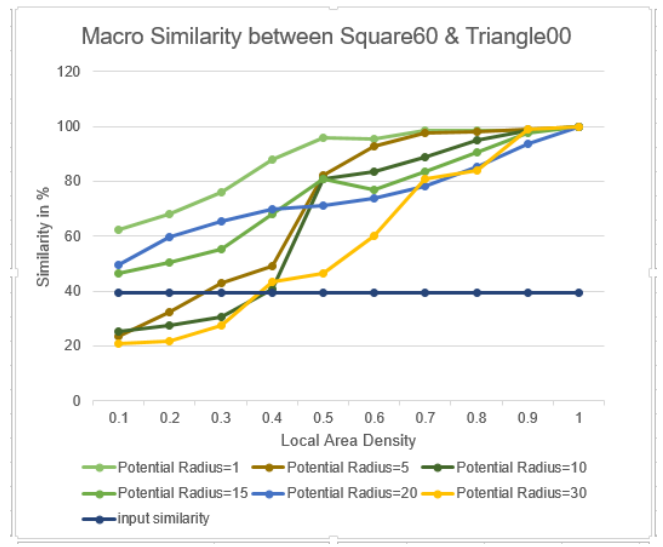


Figure 11. Graph Depicting the change in macro similarity for change in PotentialRadius and LocalAreaDensity

Figure 11 shows the output result graphically on the similarity between Square and triangle, when we run different experiment on various local area density and potential radius values. From figure 11, it can be understood that similarity between Square and triangle increase with respect to the increase in local area density, which will eventually give bad result.

Similarity Matrix:

The output similarity of the micro and macro photos is shown in Figure 12 which is the best fit similarity matrix obtained from the values depicted in table 5

LocalAreaDensity	0.3
PotentialRadius	10
NumActiveColomnsPerInhArea	-1
GlobalInhibition	False

Table 5: Optimal Parameters for 64x64 input dataset

The analysis of our results indicate that for all categories, the micro similarity ranges from 98% maximum to an average of 78% to 82%, and a minimum range of 67% to 74%. These findings suggest a satisfactory outcome for micro similarity, given that higher similarity scores indicate better results. Based on this, it is recommended that for micro similarity, the maximum similarity score should be 80% or higher, the average similarity score should be between 70% and 85%, and the minimum similarity score should be within the range of 40% to 80%.

As per the analysis, the macro similarity across all categories ranges from 45% to a maximum of 66%, with an average range of 40% to 58%, and a minimum range of 36% to 51%. It is noteworthy that lower similarity scores indicate better outcomes. Therefore, the current results suggest a favourable outcome for macro similarity. Based on this, it is recommended that only the maximum similarity score should be considered, and the ideal range for maximum similarity is between 20% and 70%.

```

cycle ** 269 ** Stability: False
cycle ** 270 ** Stability: False
TABLE STATE
Printing statistics for experiment in micro(diagonal line from top left to bottom right) and macro (rest):

```

class	Circle	Square	Star	Triangle
Circle	Max 90.83547782846 Avg 86.9783897289834 Min 80.97087378648776	Max 70.481089581222 Avg 58.645501877009515 Min 50.0970737864878	Max 84.46801941747572 Avg 74.77732313978818 Min 66.60840511627907	Max 86.40221317829486 Avg 73.01518862709103 Min 52.038834951456316
Square	Max 78.48618095512573 Avg 58.64550187700951 Min 50.0970737864878	Max 90.89108918091809 Avg 69.78588015068122 Min 50.09071779883946	Max 71.14780818421597 Avg 52.558428886718814 Min 44.18084651162791	Max 81.23751181141807 Avg 53.04631914881877 Min 33.59841116478008
Star	Max 84.46801941747572 Avg 74.77732313978818 Min 66.60840511627907	Max 72.14700193423597 Avg 52.34862888671882 Min 44.18084651162791	Max 96.31782547736434 Avg 85.8624728473688 Min 79.96188949416343	Max 80.81395348837289 Avg 75.753672511393143 Min 70.23346383581945
Triangle	Max 85.85273317829486 Avg 73.01518862709102 Min 52.038834951456316	Max 81.23751181141807 Avg 53.04631914881877 Min 33.52941170478008	Max 80.81395348837289 Avg 75.753672511393145 Min 70.23346383581945	Max 84.18084651162791 Avg 70.471236230811015 Min 57.17804263565892

Figure 12. Similarity matrix for the parameters shown in table 5

Prediction Testing

Following the training phase, the model will possess the capability to forecast a unique set of images. The Prediction phase, as discussed in the experiment section, involves creating the Sparse Distributed Representation (SDR) of the image to be predicted. Subsequently, the active columns of the predicted image's SDR are compared to the active columns of training image SDRs, thereby providing a set of similarity values expressed in percentages between different classes.

Figure 13 displays the images from the Circle, Square, and Star classes, which have been utilized to evaluate the prediction performance of the image classification model. Based on the output presented in Figure 12, the Circle and Square classes are expected to belong to the trained image model dataset and should achieve a high prediction accuracy. Conversely, the Star class is expected to have a lower percentage similarity score for the trained dataset, indicating a relatively weaker performance for this class.

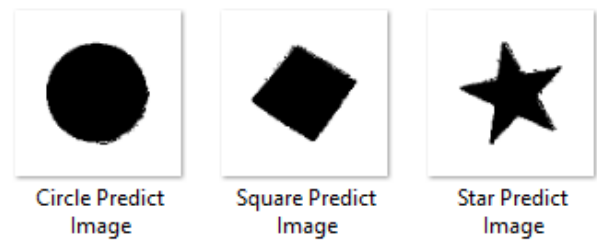


Figure 13: Images used for prediction

For the prediction testing, the following cases are considered for an optimal test of the learning model.

Case 1: Predict image belongs to same class used in training

For the following test case, consider the images 'Circle Predict Image' and 'Square predict Image' form the Figure 16 for the input dataset consisting of circle, Square, and triangle as the dataset.

The Figure 14 shows the prediction percentages for the 'Circle Predict Image'. The prediction percentage for the circle class has the maximum of 99.1% and a minimum of 69.77%. For the Square class, the 'Circle Predict Image' has the maximum of 70.08% and a minimum of 64.86% and for the triangle class it has a maximum similarity percentage of 62.9% and minimum of 53.33%. By looking into the overall average percentages of all the three class, the circle class has the highest average percentage with 83.58% which gives that the image 'Circle Predict Image' belongs to the Circle class.

```

The predicted results for the input image "Circle Predict Image" is

```

Circle:	Max:99.1%	Avg:83.58%	Min:69.77%
Square:	Max:70.08%	Avg:66.31%	Min:64.86%
Star:	Max:62.9%	Avg:57.58%	Min:53.33%

Figure 14: Predicted results for image 'Circle Predict Image'

Consider the second image for the prediction, 'Square Predict Image'. The following figure 15 shows the prediction similarity results for all the three trained classes. The prediction percentage for the circle class has the maximum of 81.32% and a minimum of 70.49%. For the Square class, the 'Square Predict Image' has the maximum of 99.1% and a minimum of 67.39% and for the star class it has a maximum similarity percentage of 78.37% and minimum of 65.96%. From the average similarity percentages, the Square class has the highest of 81.32% which depicts that the image for prediction used does belong to the Square dataset.

```

The predicted results for the input image "Square Predict Image" is

```

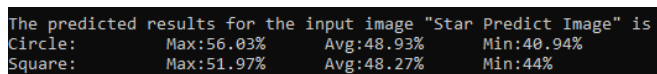
Circle:	Max:81.32%	Avg:76.55%	Min:70.49%
Square:	Max:100%	Avg:78.06%	Min:67.39%
Star:	Max:78.37%	Avg:71.5%	Min:65.96%

Figure 15: Predicted results for image 'Square Predict Image'

Case 2: Predict image belongs to different class

For this test case, involving the 'Star Predict Input' image from Figure 17, and a training dataset consisting of Circle, Square, and triangle classes, it is theoretically expected that the trained image model would yield low similarity percentage scores when compared to the input image. This is due to the fact that the input image belongs to a different class (Star) than that of the trained dataset, implying that the model has not been trained to accurately classify this type of image. As a result, the similarity percentages between the input image and the trained dataset are expected to be relatively lower than those achieved for images belonging to the trained classes.

Figure 16 displays the prediction percentages for the 'Star Predict Image'. The maximum and minimum prediction percentages for the Circle class are 56.03% and 40.94%, respectively. For the Square class, the 'Star Predict Image' has a maximum similarity percentage of 51.97% and a minimum of 44%. The average similarity values for the trained image classes are 48.93% and 48.27% for the Circle and Square classes, respectively. These findings suggest that the 'Star Predict Image' exhibits a lower level of similarity with the trained image classes (Circle, Square), thereby indicating that the features of the 'Star Predict Image' do not align well with those of the trained image classes.



The predicted results for the input image "Star Predict Image" is			
Circle:	Max:56.03%	Avg:48.93%	Min:40.94%
Square:	Max:51.97%	Avg:48.27%	Min:44%

Figure 16: Predicted results for image 'Star Predict Image'

V. CONCLUSION

The aim of this project was to explore different image categories to determine the best approach for classifying and predicting these images based on a trained model. By conducting 102 experiments, it was determined that achieving a maximum micro similarity of 80% or higher, a minimum similarity of 40-80%, and an average macro similarity between 0-70% was a favorable outcome. Additionally, it was observed that utilizing a smaller local area density (less than 1) led to improved results, and that the output converged to 100% when the local area density was 1. These findings indicate that certain image characteristics, such as micro similarity, local area density, and macro similarity, can impact the accuracy of image classification and prediction models.

As a suggestion for future work, it could be beneficial to incorporate an additional function that indicates the appropriate potential radius for an image based on its dimensions. This is because the potential radius can vary depending on the image dimensions, and selecting an appropriate radius is crucial for achieving optimal results in image classification and prediction tasks. By implementing such a function, the model could automatically determine the appropriate potential radius for each image based on its dimensions, thereby improving the accuracy and efficiency of the model.

VI. REFERENCES

- [1] "Numenta," [Online]. Available: <https://numenta.com/resources/biological-and-machine-intelligence/spatial-pooling-algorithm/>.
- [2] D. S. D. Sundari, "Hierarchical Temporal Memory Image Classification".
- [3] D. Dobric, "Github," [Online]. Available: <https://github.com/ddobric/neocortexapi-classification>.
- [4] O. Rukundo, "Effects of image size on deep learning," [Online]. Available: <https://arxiv.org/ftp/arxiv/papers/2101/2101.11508.pdf>.
- [5] Z. C. Y. Q. Z. Y. Y. X. Wen Zhuo, "Image classification using HTM cortical learning algorithms," [Online]. Available: <https://ieeexplore.ieee.org/document/6460663>.
- [6] H. J. a. G. D., "Hierarchical Temporal Memory: Concepts, Theory, and Terminology.," [Online]. Available: http://www.numenta.com/Numenta_HTM_Concepts.pdf.