INVESTIGATING IMAGE RECONSTRUCTION USING HIERARCHICAL TEMPORAL MEMORY AND K-NEAREST NEIGHBOURS CLASSIFIERS

Anoushka Piplai (1566664)
anoushka.piplai@stud.fra-uas.de

Avradip Mazumdar (1566651)
avradip.mazumdar@stud.fra-uas.de

Raka Sarkar (1567153)
raka.sarkar@stud.fra-uas.de

Somava Ganguly (1566916)
somava.ganguly@stud.fra-uas.de

Abstract- Input reconstruction plays a vital role in machine learning and pattern recognition, allowing for accurate recovery of encoded data. This study explores the use of Hierarchical Temporal Memory (HTM) and K-Nearest Neighbors (KNN) classifiers for reconstructing input images by Sparse Distributed Representations (SDRs) generated by the HTM Spatial Pooler. The research leverages NeoCortexAPI for encoding input images, training classifier on stable SDRs generated by Spatial Pooler, and reconstructing predicted inputs from SDRs. The dataset is divided into 80% for training and 20% for testing, where both classifiers learn from the SDRs and attempt to reconstruct original inputs. The reconstructed images are compared to their original binarized versions using Jaccard Index and Hamming Distance similarities to measure reconstruction accuracy. Additionally, a comparative analysis is conducted to evaluate the similarities between HTM and KNN reconstructed images. The experimental results demonstrate that both classifiers successfully reconstruct input patterns, with HTM showing higher accuracy in pattern recovery.

Keywords: *Hierarchical Temporal Memory (HTM), K-Nearest Neighbors (KNN), Input Reconstruction, Sparse Distributed Representations (SDRs), Spatial Pooler, Jaccard Index Similarity, Hamming Distance Similarity, Machine Learning, Pattern Recognition*

I. INTRODUCTION

Sparse Distributed Representations (SDRs) play a pivotal role in Hierarchical Temporal Memory (HTM) systems, particularly in pattern recognition and anomaly detection tasks. SDRs, characterized by their high-dimensional and sparse nature, provide a biologically inspired method of encoding information, enabling robust learning and generalization in artificial intelligence (AI) models [1]. These properties make SDRs particularly suitable for applications in image recognition, where preserving spatial and temporal dependencies is crucial.

This research explores the process of training and utilizing both HTM and k-Nearest Neighbors (KNN) classifiers on binarized image data, focusing on predicting and reconstructing images through SDRs, as illustrated in Figure 1. The ability to reconstruct original images from predicted representations is integral for evaluating the effectiveness of these classifiers. Image reconstruction serves as a critical metric for assessing the retention of essential image features and the fidelity of the classification process [7].

Our approach begins by encoding input images into SDRs using the Spatial Pooler (SP), a core component of HTM responsible for converting raw input data into sparse, stable representations. Each SDR is a large binary vector, where active elements are represented by 1s, and inactive elements by 0s. Both HTM and KNN classifiers are trained on these SDRs to learn patterns and make predictions. Performance evaluation is conducted by reconstructing images from predicted SDRs and comparing them to their original binarized counterparts. This process involves two key components:

- ImageReconstructor, which translates predicted SDRs back into image form.

- Similarity Metrics, which assess reconstruction accuracy using measures such as Jaccard Index Similarity and Hamming Distance Comparison. [10].

Additionally, this study explores the visualization of reconstruction accuracy through similarity plots and performance graphs. These visualizations provide quantitative insights into the fidelity of reconstructed images, highlighting the comparative performance of HTM and KNN classifiers. Previous research has demonstrated that HTM, inspired by neocortical learning principles, exhibits superior performance in handling spatially correlated data, whereas KNN, a non-parametric method, is effective in instance-based learning [3]. By analyzing the reconstruction quality of both classifiers, this research contributes to a deeper understanding of SDR-based learning systems in image recognition tasks.

Advancements in biologically inspired computing and sparse encoding techniques have led to improved methodologies for processing visual data in AI systems. The findings of this study aim to provide insights into the potential advantages of

SDR-based classification models, particularly in applications where robustness and interpretability are essential, such as medical imaging, automated surveillance, and autonomous systems [5].

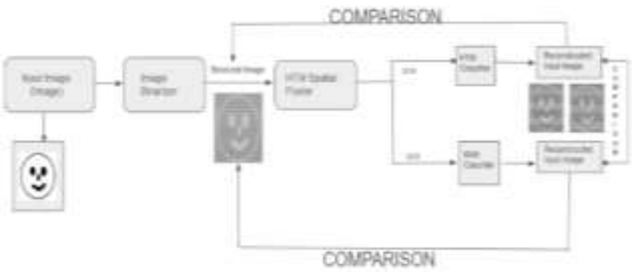

FIGURE 1: PROJECT OVERVIEW

II. METHODS

This section describes the methods used to train and evaluate two classifiers, Hierarchical Temporal Memory (HTM) and k-Nearest Neighbors (KNN), for image recognition and reconstruction using Sparse Distributed Representations (SDRs). The image is first binarized then processed into SDRs by Spatial Pooler, and both classifiers are trained using these representations. Subsequently, the system performs predictions and image reconstructions, comparing the results to the original images.

A. IMAGE BINARIZER

Function:
Converts raw image data into a binarized format before generating Sparse Distributed Representations (SDRs) using the Spatial Pooler (SP).
Details:
The Image Binarizer preprocesses input images by converting them into binary (black-and-white) representations, where pixel values are either 0 or 1 based on a predefined threshold [6]. This step is crucial for ensuring that the images are suitable for SDR generation and further processing by HTM (Hierarchical Temporal Memory) and KNN (K-Nearest Neighbors) classifiers. By simplifying the image structure while preserving key features, the binarization process enhances the ability of classifiers to recognize patterns effectively. The binarized images serve as input to the Spatial Pooler, which then produces the corresponding SDR representations for classification and reconstruction tasks [3].

B. SPATIAL POOLER (SP)

Function: Encodes the raw input data (image) using the Spatial Pooler (SP) to produce a Sparse Distributed Representation (SDR).
Details: This method activates specific columns based on the input image's pattern, turning on certain neurons (cells) in the pool. These active columns form the SDR, which is used for pattern recognition tasks. The SP serves as a feature extractor, transforming the input image into a compact and efficient binary representation [3]. The active array is then used to predict the class or label of the image in the subsequent steps as shown in Figure 2.

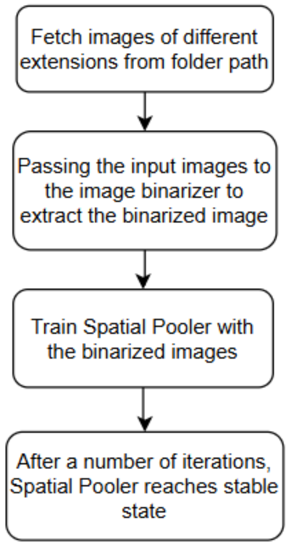

FIGURE 2: SPATIAL POOLER LEARNING PHASE

C. SDR (SPARSE DISTRIBUTED REPRESENTATION)

Function: Trains both the HTM and KNN classifiers using the Sparse Distributed Representations (SDRs) of training images.
Details: This method takes a dictionary of images and their corresponding SDRs as input, where each image is represented by an array of Cell objects (SDR). It then trains both HTM (Hierarchical Temporal Memory) and KNN (K-Nearest Neighbors) classifiers using the provided SDRs. Each image's SDR is fed to the classifiers, which learn the patterns and features of the images for future predictions. The method logs the training process and tracks the images used for training [2]. Few values of the different HTM parameters are as shown in Table 1.

Parameters	Values
CellsPerColumn	10
InputDimensions	new int[]{64, 64}
NumInputs	64*64
ColumnDimensions	new int[] { 64, 64 }
MaxBoost	5.0
DutyCyclePeriod	100
MinPctOverlapDutyCycles	1.0
GlobalInhibition	False

TABLE 1: VALUES OF HTM PARAMETERS

D. CLASSIFIER TRAINING

Function: Trains both the HTM and KNN classifiers using binarized images represented as Sparse Distributed Representations (SDRs).
Details: The system processes training images by converting them into binary format and transforming them into SDRs. The HTM classifier learns by adjusting synaptic permanence to recognize spatial patterns, while the KNN classifier stores SDR representations and associates them with labels for

nearest-neighbor classification. The system logs the trained image names and records training time for performance evaluation [3].

E. Prediction & Reconstruction

Function: Performs image prediction and reconstruction using both HTM and KNN classifiers, comparing the reconstructed images to the original images.

Details: This method takes encoded input (SDR) from the Spatial Pooler (SP) and performs prediction using both HTM and KNN classifiers. After obtaining the predicted images, it uses an Image Reconstructor to reconstruct the predicted images and compares them with the original image for similarity. The reconstruction is evaluated using a similarity metric, and the method generates similarity graphs and plots to visually represent how well the classifiers have reconstructed the image [8]. This whole process is pictorially described in Figure 3.

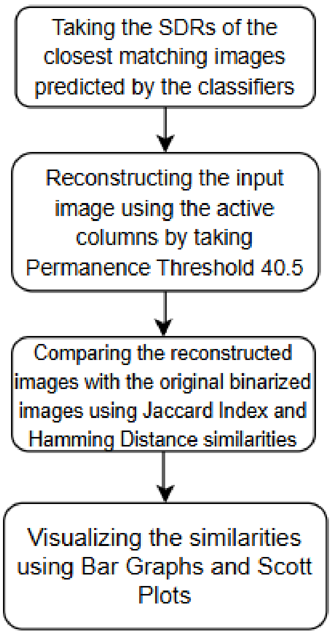

Figure 3: Prediction and Reconstruction Workflow

F. Similarity Calculation of SDRs

Function: The system calculates the similarity between reconstructed images and their original versions using Jaccard Index Similarity and Hamming Distance Similarity to assess reconstruction accuracy.

Details: The effectiveness of the classifiers is measured by comparing the reconstructed images with their original binarized versions. Jaccard Index Similarity evaluates how much overlap exists between the two images by calculating the ratio of common active pixels to the total unique pixels in both images. A higher Jaccard Index Similarity score indicates better reconstruction accuracy as shown in Figure 4. Hamming Distance Similarity, on the other hand, measures bitwise differences between the original and reconstructed images as shown in Figure 5. This metric provides insight into structural deviations and the level of distortion introduced during the reconstruction process. Both similarity measures are recorded and stored for further performance evaluation and analysis [4].

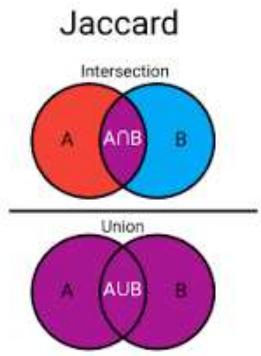

Figure 4: Visual Representation of Jaccard Similarity Index (reference: https://www.maartengrootendorst.com/blog/distances/)

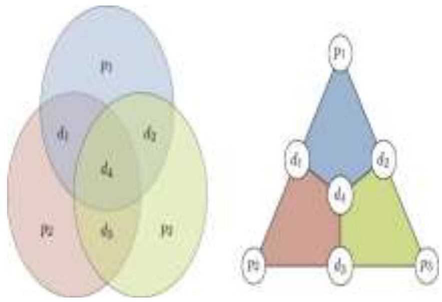

Figure 5: Visual Representation of Hamming Distance (reference: https://qubit.guide/14.1-the-hamming-code)

G. Visualization of Results

Function: The system generates graphical representations of the similarity results using Jaccard Similarity Graphs and Hamming Distance Similarity Plots to visualize classifier performance

Details: The Jaccard Similarity Graph presents a side-by-side comparison of the accuracy scores of HTM and KNN classifiers for each test image. This allows for an easy assessment of how well each classifier preserves image structures. Additionally, a Hamming Distance Similarity Plot is created using ScottPlot, illustrating the variation in similarity scores across different test images. These plots visually highlight the strengths and weaknesses of each classifier, making it easier to interpret their effectiveness in reconstructing images. The visualizations are saved as images for reference and further analysis [4].

G. Final Evaluation & Classifier Reset

Function: The system performs a final assessment to determine which classifier performed better in image prediction and reconstruction.

Details: After computing similarity scores and generating visualizations, the system evaluates the overall performance of HTM and KNN classifiers. It determines how often one classifier outperforms the other based on internal similarity metrics. By analyzing the results, the system identifies which classifier is more reliable in different scenarios. Once the evaluation is complete, the classifiers are reset, preparing them for future experiments [9].

III. RESULTS

A. IMAGE BINARIZATION

The image binarization process plays a fundamental role in preparing images for classification. The binarization algorithm, implemented in the binarizeImage function, processes input images, as shown in Figure 6, from a specified training folder by resizing them to 64x64 pixels and converts them into binary representations. This was achieved through the BinarizeImages function, which applies thresholding techniques to distinguish between black (0) and white (1) pixels. The dataset is randomly shuffled and divided into 80% training and 20% testing subsets to ensure a balanced learning process. Binarized images, as shown in Figure 7, are then stored in a designated folder named BiarizedImages, maintaining a mapping between actual images and their binarized versions for accurate reconstruction. The success of this step is evident in the structured SDR representations produced, ensuring robust feature extraction for the Spatial Pooler (SP) and subsequent classification tasks.

FIGURE 6: EXAMPLE OF AN INPUT IMAGE

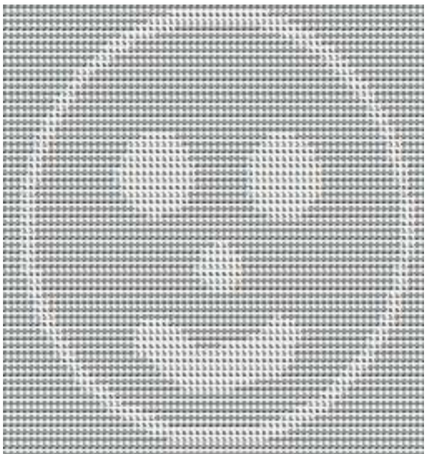

FIGURE 7: BINARIZED VERSION OF THE INPUT IMAGE

B. SPATIAL POOLER TRAINING

In the Spatial Pooler training phase, these binarized images are then encoded into sparse distributed representations (SDRs) by feeding them into the Spatial Pooler. As mentioned in Section II B, the Spatial Pooler learns spatial patterns by identifying active columns based on input overlap, updating synaptic connections, and adjusting permanence values over multiple training cycles. Stability is monitored using a Homeostatic Plasticity Controller, ensuring that the model reaches a stable state where the learned representations remain consistent as shown in Figure 8. Once the Spatial Pooler stabilizes, the SDRs generated for training images are stored and used for classification and reconstruction tasks.

FIGURE 8: OUTPUT OF SPATIAL POOLER TRAINING

C. CLASSIFIER TRAINING

The training phase involved utilizing two different classifiers: Hierarchical Temporal Memory (HTM) and K-Nearest Neighbors (KNN), both trained using Sparse Distributed Representations (SDRs) of binarized training images. A dictionary, training Images SDRs, was used where each key represents an image identifier (filename), and the value is the binarized SDR of the image.

For each training image, both classifiers processed the associated SDR to update their respective models, as discussed in Section II D. The training process for both classifiers was completed successfully without any errors as shown in Figure 9. Upon completion, the system logged the names of the images used for training.

Training Time: The total training time, as measured by the stopwatch, was recorded at approximately 7 ms.

FIGURE 9: OUTPUT OF CLASSIFIER TRAINING

D. PREDICTION AND RECONSTRUCTION

The prediction phase was carried out using a set of binarized testing images. For each test image, the Spatial Pooler (SP) processed the input vector to identify active columns, which were then used as input to both the HTM and KNN classifiers for prediction, as mentioned in section II E. The classifiers then output predicted labels based on the similarity of the test image's SDR to those learned during training.

The similarity of the reconstructed images was evaluated by comparing them to the original binarized images. Both HTM and KNN classifiers were able to make predictions and reconstruction of images based on the input SDRs, producing reconstructed images for each test case, as described in Section II E**.** The images reconstructed by the HTM classifier, as shown in Figure 10, are stored in a folder named ReconstructedHTM. Similarly, the images reconstructed by the KNN classifier, as shown in Figure 11, are stored in the ReconstructedKNN folder.

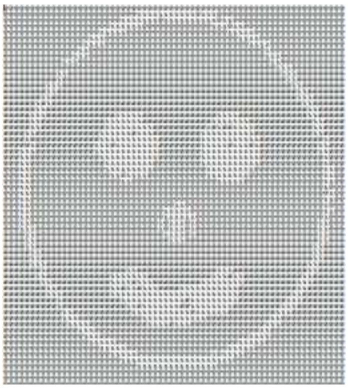

FIGURE 10: IMAGE RECONSTRUCTED BY HTM CLASSIFIER

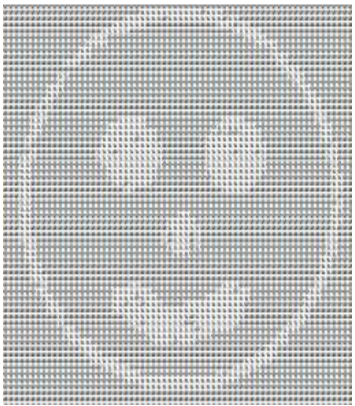

FIGURE 11: IMAGE RECONSTRUCTED BY KNN CLASSIFIER

For each test image:

- HTM: The predicted label was compared, and the reconstructed image's similarity to the original was calculated.
- KNN: Similarly, the KNN predicted label was compared, and the reconstructed image's similarity was calculated.

Also, the performance of both classifiers is compared based on their internal similarity. The system tracked which classifier produced a higher similarity score for each test image, and results were logged for analysis, as mentioned in Section II F**.**

- Best Prediction: For each test image, the classifier that produced the higher similarity was flagged as the "better" classifier for that test case. In instances where both classifiers performed equally well, a message indicating equality was logged.
- Prediction and Reconstruction Time: The total time taken for prediction and reconstruction, as measured by the stopwatch, was 3513 ms as shown in Figure 12.

FIGURE 12: OUTPUT OF CLASSIFIER PREDICTION AND RECONSTRUCTION TIME

E. SIMILARITY EVALUATION

The system generated similarity plots for both HTM and KNN classifiers. The similarity values were obtained from the reconstruction process, with higher values indicating a better match between the reconstructed image and the original binarized image.

- HTM Similarity Plot: A graph depicting the reconstruction similarity for HTM across all test images was generated using Jaccard Similarity Index and saved as HTM_Similarity_Graph.png in the designated folder. This plot illustrates the variation in HTM reconstruction performance across different test images as shown in Figure 13.

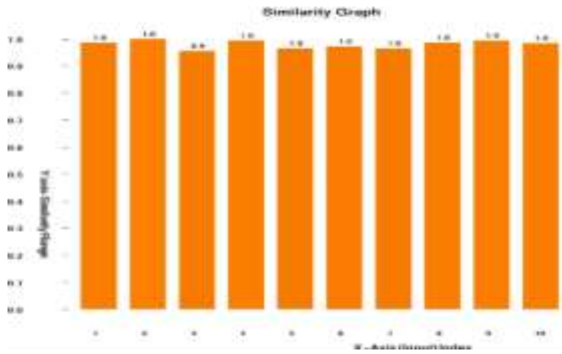

FIGURE 13: GRAPH DEPICTING THE SIMILARITY BETWEEN THE ORIGINAL BINARIZED IMAGE AND THE IMAGE RECONSTRUCTED BY HTM CLASSIFIER

- KNN Similarity Plot: Similarly, a graph for KNN reconstruction similarity was created and saved as KNN_Similarity_Graph.png in the designated folder. This plot provides a visual comparison of KNN's performance across all test images as shown in Figure 14.

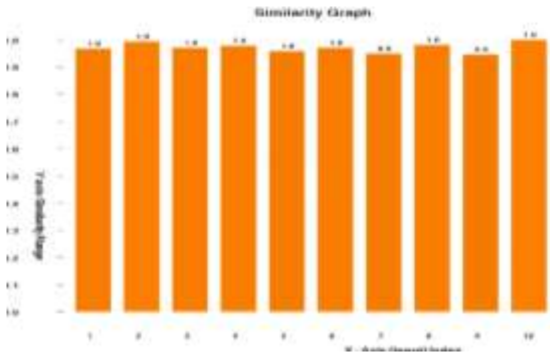

FIGURE 14: GRAPH DEPICTING THE SIMILARITY BETWEEN THE ORIGINAL BINARIZED IMAGE AND THE IMAGE RECONSTRUCTED BY KNN CLASSIFIER

Additionally, Scott Plots were generated for both HTM (as shown in Figure 15) and KNN similarity (as shown in Figure 16) results using Hamming Distance to further highlight the reconstruction similarity distribution. These plots were saved in their respective folders and can be used to assess the classifier's consistency in image reconstruction.

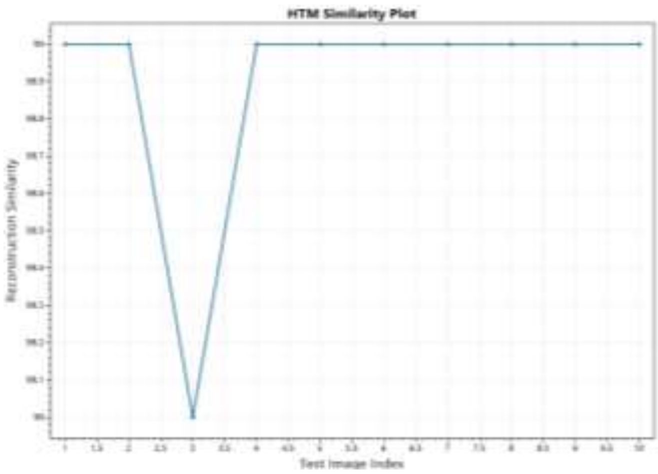

FIGURE 15: SCOTT PLOT DEPICTING THE SIMILARITY BETWEEN THE ORIGINAL BINARIZED IMAGE AND THE IMAGE RECONSTRUCTED BY HTM CLASSIFIER

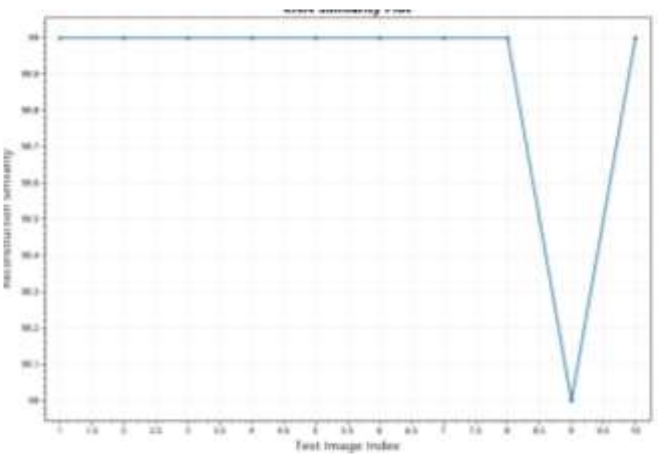

FIGURE 16: SCOTT PLOT DEPICTING THE SIMILARITY BETWEEN THE ORIGINAL BINARIZED IMAGE AND THE IMAGE RECONSTRUCTED BY KNN CLASSIFIER

F. COMPARISON OF RECONSTRUCTED IMAGES

A direct comparison of the reconstructed images from HTM and KNN was carried out. The reconstruction process involves both classifiers generating reconstructed versions of the predicted images, which were then compared for similarity. This comparison is essential in determining the performance of each classifier in reconstructing the original images.

For each test image, the reconstructed images from HTM and KNN were evaluated for their visual similarity and logged for further analysis.

G. HTM VS. KNN PERFORMANCE:

There are two scenarios encountered in our project:

1. HTM Classifier performed better than KNN Classifier as shown in Figure 17.

FIGURE 17. CONSOLE OUTPUT OF CLASSIFIER PERFORMANCE (CASE 1)

2. Both the Classifiers performed equally as shown in Figure 18.

FIGURE 18 : CONSOLE OUTPUT OF CLASSIFIER PERFORMANCE (CASE 2)

6. FINAL SUMMARY AND RESET

After completing the predictions, reconstructions, and evaluations, the system logged the final results. For overall predictions and reconstructions which classifier performed better is flagged as better classifier. As shown in Figure 19, the overall performance of HTM Classifier was better than KNN Classifier. The HTM and KNN classifiers were then reset, clearing any internal states, to prepare for any subsequent experiments or further testing as shown in Figure 19.

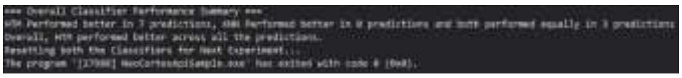

FIGURE 19: CONSOLE OUTPUT OF OVERALL CLASSIFIER PERFORMANCE

IV. DISCUSSION

The project successfully implemented and evaluated two classification methods, Hierarchical Temporal Memory (HTM) and K-Nearest Neighbors (KNN), for image recognition and reconstruction using Sparse Distributed Representations (SDRs). The classifiers were trained on a set of binarized images and tested on unseen data to assess their predictive and reconstructive performance. The prediction phase involved prediction of the best matched SDRs and the reconstruction phase involved reconstructing the predicted images and calculating similarity using Jaccard Index and Hamming Distance metrics. The results indicated that both classifiers performed well in different scenarios, with HTM excelling in some cases and KNN in others. The overall performance summary highlighted instances where each classifier was more effective, demonstrating the strengths and limitations of both approaches.

The reconstruction process provided a valuable visual analysis of classification accuracy, supported by similarity plots. The integration of similarity graphs and Scott plots further enhanced the analysis, allowing for an in-depth comparison of classifier performance. The project's methodology ensures a comprehensive evaluation, making it a valuable approach for applications involving pattern recognition and SDR-based learning models.

For future work, improvements can be made by refining feature extraction techniques, incorporating hybrid models that combine the strengths of HTM and KNN, and exploring alternative distance metrics to enhance classification accuracy. Additionally, extending the framework to support larger datasets and more complex image structures could provide further insights into the scalability and robustness of the models. The implementation of real-time prediction and reconstruction capabilities would also be an interesting direction for practical applications in computer vision and artificial intelligence.

V. References

[1] Ahmad, S., & Hawkins, J. (2015). Properties of Sparse Distributed Representations and their Application to Hierarchical Temporal Memory. *arXiv preprint arXiv:1503.07469*. Available at: https://arxiv.org/abs/1503.07469. Accessed on: March 28, 2025.

[2] Balasubramaniam, J., Krishnaa, C. B. G., & Zhu, F. (2015). Enhancement of Classifiers in HTM-CLA Using Similarity Evaluation Methods. *Procedia Computer Science*, 60, 1516–1523. Retrieved on March 29, 2025, from https://www.sciencedirect.com/science/article/pii/S1877050915023881:contentReference[oaicite:1]{index=1}

[3] Cui, Y., Ahmad, S., & Hawkins, J. (2017). The HTM Spatial Pooler—A Neocortical Algorithm for Online Sparse Distributed Coding. *Frontiers in Computational Neuroscience*, 11, 111. Available at: https://www.frontiersin.org/articles/10.3389/fncom.2017.00111/full. Accessed on: March 28, 2025.

[4] Jaccard, P. (1912). *Étude comparative de la distribution florale dans une portion des Alpes et des Jura*. *Bulletin de la Société vaudoise des sciences naturelles*, 47, 241–272. Retrieved on March 29, 2025, from https://en.wikipedia.org/wiki/Jaccard_index

[5] Olshausen, B. A., & Field, D. J. (1997). Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1?. *Vision Research*, 37(23), 3311–3325. Available at: https://www.sciencedirect.com/science/article/pii/S0042698997001697. Accessed on: March 28, 2025.

[6] Otsu, N. (1979). *A Threshold Selection Method from Gray-Level Histograms*. IEEE Transactions on Systems, Man, and Cybernetics. Available at: https://engineering.purdue.edu/kak/computervision/ECE661.08/OTSU_paper.pdf. Retrieved on March 28, 2025.

[7] Yang, J., Wright, J., Huang, T. S., & Ma, Y. (2010). Image Super-Resolution via Sparse Representation. *IEEE Transactions on Image Processing*, 19(11), 2861–2873. Available at: https://www.columbia.edu/~jw2966/papers/YWHM10-TIP.pdf. Accessed on: March 28, 2025.

[8] Yilmaz, E., & Gunal, S. (2020). Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets. *SN Computer Science*, 1(3). Retrieved on March 29, 2025, from https://link.springer.com/article/10.1007/s42452-019-1356-9

[9] Yilmaz, E., Krishnaa, C. B. G., & Zhu, F. (2020). Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets using similarity measures. *Scientific Reports*, 10(1). Retrieved on March 29, 2025, from https://link.springer.com/article/10.1007/s42452-019-1356-9

[10] Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. (2017). Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. *IEEE Transactions on Image Processing*, 26(7), 3142–3155. Available at: https://ieeexplore.ieee.org/document/7839189. Accessed on: March 28, 2025.