Analyse Image Classification of Fruits360 dataset

Anu Maria Varghese
anu.varghese@stud.fra-uas.de

Tiniya Vinod Puthanpurayil
tiniya.puthanpurayil@stud.fra-uas.de

Veena Alphonsa Jose
veena.alphonsajose@stud.fra-uas.de

Abstract: **Image classification is the process of categorizing and labeling groups of pixels or vectors within an image based on specific rules. It is the most important part of digital image analysis. This paper aims to change various Hierarchical Temporal Memory (HTM) learning parameters and to find the best fit that shows the image classification and to demonstrate how these parameters influence learning. Our objective is to improve the input prediction and to implement the highest similarity of the input images. The HTM Cortical Learning Algorithm (HTM CLA) Spatial Pooler (SP) is a neocortical-inspired Cortical Learning Algorithm for learning. Its purpose is to understand the spatial pattern by creating the input's Sparse Distributed Representation code (SDR). HTM is based on the biological functions of the brain as well as its learning mechanism. HTM can be described as the theory that attempts to describe the functioning of the neocortex, as well as the methodology that intends to provide machines with the capacity to learn in a human way.**

Keywords: Spatial Pooler, Hierarchical Temporal Memory, Sparse Distributed Representation, neocortex.

I. Introduction

Over the years, and with the emergence of various technological innovations, the relevance of automatic learning methods has increased exponentially, and they now play a key role in society. This paper aims to change various learning parameters and to find the best fit that shows the image classification and to demonstrate how these parameters influence learning. Our objective is to improve the input prediction and to implement the highest similarity of the input images. HTM is based on the biological functions of the brain as well as its learning mechanism. HTM can be described as the theory that attempts to describe the functioning of the neocortex, as well as the methodology that intends to provide machines with the capacity to learn in a human way [1]. The neocortex is defined as the portion of the human cerebral cortex from which comes the highest cognitive functioning, occupying approximately half the volume of the human brain. The neocortex is understood by four main lobes with specific functions of attention, thought, perception, and memory. These four regions of the cortex are the frontal, parietal, occipital, and temporal lobes. The frontal lobe's responsibilities are the selection and coordination of behavior. The parietal lobe is qualified to make decisions in numerical cognition as well as in the processing of sensory information. The occipital lobe, in turn, has a visual function. Finally, the temporal lobe has the functions of sensory as well as emotional processing and dealing with all significant memory. Thus, the algorithm that is presented intends to create a transposition of this portion of the brain, creating a machine with true intelligence [2].

Starting with the first region, the encoder deals with all the sensory components. This will receive the data in their raw form, converting them into a set of bits that will later be transformed into a Sparse Distributed Representation (SDR). Transposing into the human organism, the SDRs correspond to the active neurons of the neocortex. Thus, a 1 bit represents an active neuron while a 0 bit represents an inactive neuron. This transformation is achieved by transforming the data into a set of bits while maintaining the semantic characteristics essential to the learning process. One of the characteristics that proved to be quite interesting is those similar data entries, when submitted to the encoding process, create overlapping SDRs; that is, with the active bits placed in the same positions. Another important characteristic is that all SDRs must have a similar dimensionality and sparsity (the ratio between the number of bits at 1 and the total number of bits) [3]. A certain percentage of sparsity will result in a system's ability to handle noise and under-sampling.

The second region, Spatial Pooler (SP), is responsible for assigning the columns according to a fixed number, where each column corresponds to a dendritic segment of the neuron that connects to the input space created by the region described above, the encoder. Each segment has a set of synapses that can be initialized at random, with a permanence value. Some of these synapses will be active (when connected to a bit with value 1) and consequently will be driven in such a way as to inhibit other columns in the vicinity. Therefore, the SP is responsible for creating an SDR of active columns. This transformation follows the Hebbian learning rule that for each input, the active synapses are driven by inhibiting the inactive synapses. The thresholds dictate whether a synapse is active or not.

The third region, Temporal Memory (TM), starts from the result of the previous two, finding patterns in the sequence of SDRs to determine a prediction for the next SDR. At the beginning of the process, all the cells of the active column are also active; however, the region TM is responsible for activating a subset of cells of those same columns when a context is predicted. In case there is no forecast, all the cells remain active. The activation of the previously mentioned subsets of cells is carried out because only in this way can the same entry be represented according to different contexts.

Finally, the classifier is the region in which a decoder calculates the overlap of the predicted cells of the SDR obtained, selecting the one with more overlaps and comparing it with the actual value (if known) [4] [5].

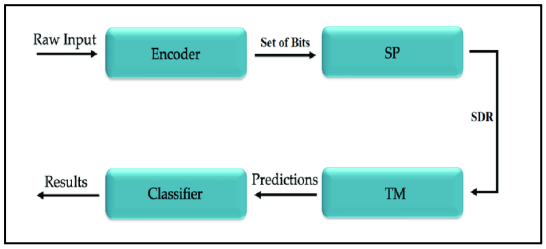

Figure 1: HTM Network Topology

Figure 1 describes the typical process of an HTM network. HTM is built on three main features of the neocortex: it is a memory system with temporal patterns and its regions are organized in a hierarchical structure. There are many biological details that the theory ignores in case they have no relevance for learning. In short, this approach includes Sparse Distributed Representation (SDR)s, its semantical and mathematical operations, and neurons along the neocortex capable of learning sequences and enabling predictions; these systems learn in a continuous way, with new inputs through time and with flows of information top-down and bottom-up between its hierarchical layers, making them efficient in detecting temporal anomalies. The theory relies on the fact that by mimicking the neocortex, through the encoding of data in a way that gives it a semantic meaning, activating neurons sparsely in an SDR through time will give these systems a power to generalize and learn, not achieved to date with other classic approaches of AI.

II. Methods

A) HTM

HTM is based on the biological functions of the brain as well as its learning mechanism. The results are of significant relevance and show a low percentage of errors in the predictions made over time. The Hierarchical Temporal Memory Cortical Learning Algorithm (HTM CLA) is a theory and machine learning technology that aims to capture the cortical algorithm of the neocortex [6].

HTM consists of 2 different components: Spatial Pooler and Temporal Memory. Figure 2

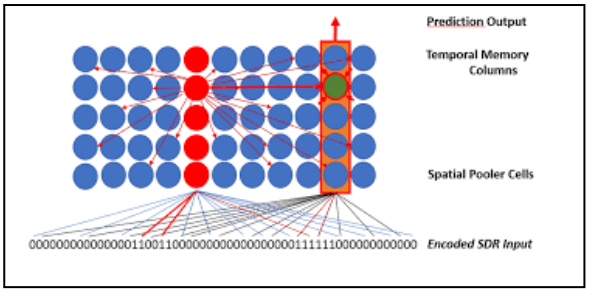

Figure 2: Data Processing in HTM

depicts the principle network connectivity and data processing in the Hierarchical Temporal Memory. The HTM implements technical equivalents for sensory organs and learning of spatiotemporal sequences in the neocortex. Raw sensory data is processed by encoders and fed into the Cortical Learning algorithm consisting of Spatial Pooler and Temporal Memory cells.

B) Sparse Distributed Representations (SDR)

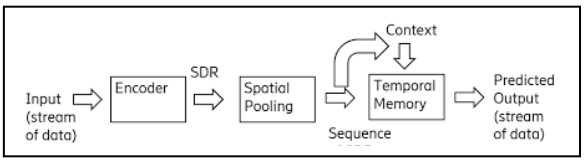

Figure 3: SDR Dataflow

An SDR is a vast array of bits with most of them turned off (0s) and only a few turned on (1s) (1s). Each SDR represents some meaning since two SDRs are judged to have equivalent meaning if they have several overlapping places on bits. The data is more comparable or the gap between two SDRs is smaller the more bits they share. Figure 3 shows the SDR Dataflow [6].

C) Temporal Memory

Temporal memory is an algorithm which learns sequences of Sparse Distributed Representations (SDRs) formed by the Spatial Pooling algorithm and makes predictions of what the next input SDR will be. Temporal Memory algorithm predicts what the next input SDR will be based on sequences of Sparse Distributed Representations (SDRs) produced by the Spatial Pooling algorithm. Each column in the SDR consists of many cells. Each cell can have three states: active, predictive, and inactive. These cells should have one proximal segment and many distal dendrite

segments. The proximal segment is the connection of the column and several bits in the input space [6].

D) Spatial pooler

Spatial Pooler (SP) is a learning algorithm that is designed to replicate the neuron functionality of the human brain. Essentially, if a brain sees one thing multiple times, it is going to strengthen the synapses that react to the specific input result in the recognition of the object. Similarly, if several similar SDRs are presented to the SP algorithm, it will reinforce the columns that are active according to the on bits in the SDRs. If the number of training iterations is big enough, the SP will be able to identify the objects by producing different sets of active columns within the specified size of SDR for different objects [6].

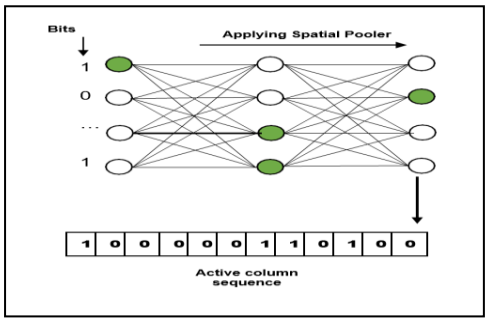

Figure 4: Spatial Pooler Applying Process

In the Figure 4 we can see after applying Spatial Pooler we get active column sequence. In the Figure 4 the green nodes indicate the active bits. We use spatial pooler to train our binarize data. Then we get an active column sequence and store it for further use. Figure 5 shows the columns in Spatial Pooler and Temporal memory.

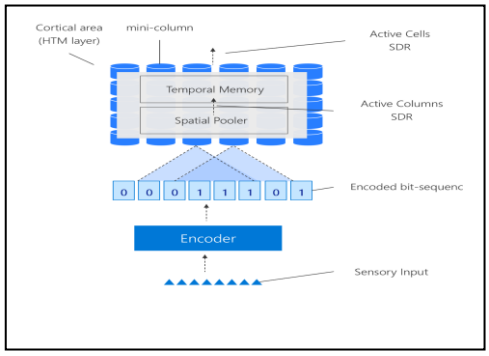

Figure 5: Spatial Pooler and Temporal memory

E) Image Binarization

A binary image is a digital image that has only two uses for a binary image, black and white, through any two colors can be used. The color used for the objects in the image is the foreground color while the rest of the image is the background color [7].

- Get an image from a file.
- Create equivalent binary

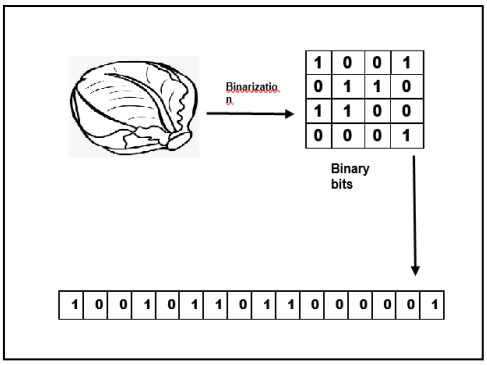

Figure 6:Image Binarization Process

Figure 6 demonstrates the process how an image converts into binary format. This is the process of taking a grayscale image and converting it to black and white, essentially reducing the information contained within the image from 256 shades of gray to 2: black and white, a binary image. This is sometimes known as image thresholding, although thresholding may produce images with more than 2 levels of gray. It is a form or segmentation, whereby an image is divided into constituent objects.

F. Dataset

A dataset is a collection of data pieces that can be treated by a computer as a single unit for analytic and prediction purposes. The selection of a dataset as well as the features to be used may be determinant for the success of the research work. Datasets is not just a simple data repository. Each dataset is a community where discussion of data, discovery of public code and techniques, and creation of projects in notebooks. Dataset used for experiments are from Fruit 360 of Kaggle.

Kaggle provides a vast container of datasets, sufficient for the enthusiast to the expert. Kaggle supports a variety of dataset publication formats, but dataset publishers encourage to share their data in an accessible, non-proprietary format if possible [8]. Not only are open, accessible data formats better supported on the platform, but they are also easier to work with for more people regardless of their tools. Figure 7 shows the datasets obtained from Kaggle.

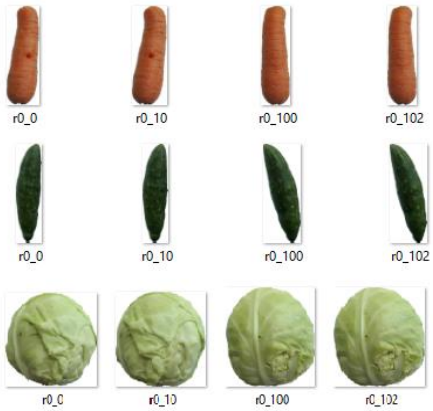

Figure 7: Dataset

III. Results

Results in this paper show the correlation validation between the input images and provide the correlation matrix. In Figure 8 shows the similarity matrix of the input images.

Parameter Name	Values
POTENTIAL_RADIUS	30
POTENTIAL_PCT	0.5
GLOBAL_INHIBITION	False
INHIBITION_RADIUS	1
LOCAL_AREA_DENSITY	0.1
NUM_ACTIVE_COLUMNS_PER_INH_AREA	-1
STIMULUS_THRESHOLD	0.0
SYN_PERM_INACTIVE_DEC	0.01
SYN_PERM_ACTIVE_INC	0.1
SYN_PERM_CONNECTED	0.1
DUTY_CYCLE_PERIOD	10
MAX_BOOST	10.0

Table 1: HTM Input Parameters

prediction code. New method called PredictLabel is implemented to predict the label of the image which is imputed by comparing the binarized values of the images.

Case 1: By changing various HTM parameter to find the best fit correlation matrix

HTM setting of the project can be inputted to the program by means of a .json file htmconfig.json. Multiple experiments can therefore be conducted via changes of parameters in the json file. Most important learning parameters are: Global/Local Inhibition, Potential Radius, Local Area Density and NumofActiveColumnsPerInArea and we found how these parameters influenced learning. Table 1 shows the HTM parameters default values. After conducting various tests, we have been able to find the parameters at which we get the least overlapping in between Micro and Macro and thus the best

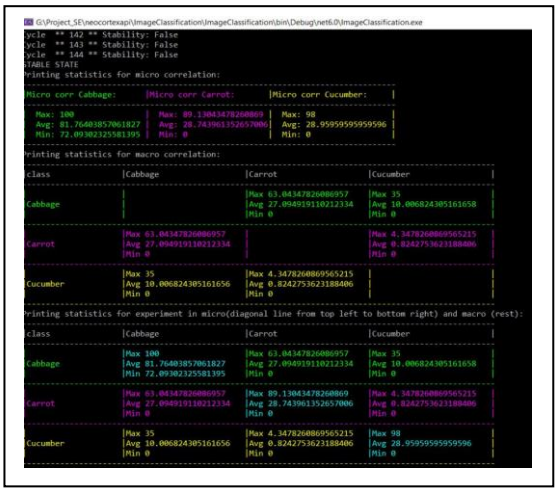

Figure 8: Output of Similarity Matrix

Case 2: By modifying the prediction code to calculate the highest similarity of the input images

To test the quality of learning we have improved the prediction code to calculate the highest similarity of the input images. The prediction code provides a set of predicting results like: “Cabbage – 87%, Carrot 7%, Cucumber - 3%”. In Figure 9 the input label prediction and highest similarity matrix is displayed.

Listing 1 shows the prediction code. Select one image path from input folder. The image is binarized into an integer array with ReadImageData. Using Compute method of API, the SDR of the image is calculated and is then compared with the SDR values of other images which is available in the input folder using the PredictLabel function. We can get the similarity of the images using CalcArraySimilarity.

```
string PredictLabel(int[] sdrOfInputImage,
Dictionary<string, int[]> sdrs)

        {

            //Dictionary<string, List<string>>
inputsPath = new Dictionary<string,
List<string>>();

            string label = "Could not able to
predict the label";

            double similarityWithEachSDR = 0;
```

Listing 1: Prediction code

Figure 9: Similarity Matrix with Input Prediction

IV. Discussion

We have used the process and properties of image classification based on Spatial pooler. The Spatial pooler has a flexible coding schema that can be used in real life machine learning applications. We did not use any encoder. We just binarize the image and give it to the spatial pooler. If we look in our image and the output table, we can see exactly matched images are showing 100%. Further, some images are quite similar but there may be some little dissimilarity.

Now in this era, there are many smart machines that use image recognition for the identification process. All of this may not be 100% accurate. Though our project has some limitations, we can apply it to various real-life work. We can improve result accuracy by using more images.

References

[1] C. Neto, M. Brito, H. Peixoto, V. Lopes, A. Abelha and J. Machado, "Prediction of Length of Stay for Stroke Patients Using Artificial Neural Networks," *Information Systems and Technologies*, vol. 1159, pp. 212-221, 2020.

[2] A. Ghazanfar and C. Schroeder, "Is neocortex essentially multisensory?," *Trends in Cognitive Sciences*, vol. 10, no. 6, pp. 278-285, 2006.

[3] Y. Cui, S. Ahmad and J. Hawkins, "Continuous Online Sequence Learning with an Unsupervised Neural Network Model," *Neural Computation*, vol. 28, no. 11, p. 2474–2504, 2016.

[4] D. Maltoni, "Pattern Recognition by Hierarchical Temporal Memory," SSRN, 2011. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3076121. [Accessed March 2022].

[5] S. Purdy, "Encoding Data for HTM Systems," *Neural and Evolutionary Computing*, no. arXiv:1602.05925, 2016.

[6] D. Dobric, "Neocortexapi," [Online]. Available: https://ddobric.github.io/neocortexapi/.

[7] M. Wirth, "Craftofcoding," WordPress, 17 2 2017. [Online]. Available: https://craftofcoding.wordpress.com/2017/02/13/image-binarization-1-introduction/. [Accessed 3 2022].

[8] "Kaggle," Kaggle, [Online]. Available: https://www.kaggle.com/. [Accessed 2021].