Abstract – Machine learning has been used intensively since the past 30 years to discriminate pixels from background or objects of interest from other classes of objects by training on a set of relevant features. As image sources are now producing more images that we can realistically cope with, the goal is to explore the limits of these approaches for ATD/ATR in order to optimally define the domains in which decisions can be left to automated processes or should require human intervention. With this objective in mind, this paper presents an assessment of the performances of the Holographic Neural Technology (AND Corporation) to support applications that would require incremental learning.

Keywords: Machine learning, neural networks, holographic memory, ATD/ATR.

1 Introduction

Within the past 30 years, tremendous efforts in image processing were devoted to solving the target detection and recognition problem in the most automated way. In front of the extreme complexity of the problem, and based on the knowledge and expertise gained over these early years of research, it is now largely accepted that human intervention still has to play a critical component in the decision process.

Among the powerful techniques proposed over the years, we will mention the matched filtering techniques with all the algorithms providing detection and recognition invariant under target rotation, scale and illumination [1-7]. Based on a correlation between the object of interest and the observed scene, these approaches yields fast detections without having to actually segment the object within the scene. The pitfall however is that detection performance highly depends on the matching between the object of interest and its occurrence within the scene and drops significantly when this occurrence shows a partially occluded or shadowed object.

Machine learning approaches were logically introduced to solve this generalisation problem by selecting significant object vs. context features, learn from their structural or natural variants, and use this knowledge in the detection/recognition process [8-12]. Our in-house investigation led us to conclude that one of the most promising software currently available commercially describes objects as the aggregation of polygonal shapes extracted by combining radiometric, geometric or contextual features with fuzzy-logic theory [13]. The difficulty resides in selecting the appropriate features and defining the membership functions and rules that drive the polygonal aggregation.

Another interesting and promising attempt proposed to solve the generalisation problem is to detect a set of object components rather than an instantiation of the object as a whole [14]. Object recognition is then defined as the process of associating the detected components into a consistent object of interest. While the algorithm currently proposed for object recognition from components is based on machine learning engines, a matched filtering approach can as easily be designed.

Conscious of the importance of using an optimized combination of computer programs and human expertise to solve the ATD/ATR problem, we revisited several machine learning architectures and selected the engines that could serve the purpose of an incremental learning tool, i.e. a tool capable of getting smarter as the human sorts out true detections from false alarms without having to be retrained.

This paper is focused on the performance evaluation of the first results obtained with an ATD/ATR tool based on an incremental implementation of the Holographic Neural Theory [15]. This technology has been investigated with a wide variety of signal, image processing and communication applications. Section 2 will give a brief recall of the principles and mathematics of this theory. Section 3 details the implementation of the software. Section 4 describes the experimentations conducted so far and Section 4 the results we obtained. A conclusion and future developments are given Section 5.

2 The holographic neural theory

This section provides an overview of the principles and formalism supporting the Holographic Neural Theory. We used for this study the Holographic Neural Technology (HNeT) software that has been developed and is commercialized by AND Corporation [15]. This technology has been applied to numerous fields ranging from medical, control systems, vision, voice or character
recognition, signal processing and communication. At the core of this technology is the representation of the information as complex-valued vectors where phase angles $\theta$ carry the real value of the measurements (radiometry, texture, temperature etc.) and the magnitudes $\lambda$ represent the confidence in that information. Confidence or magnitude values are bounded within $[0,1]$. A typical input or basic stimulus vector of dimension $m$ is represented as:

$$ S = [\lambda_j e^{i\theta_j}], \quad j = 1, m $$ (1)

### 2.1 Cerebellar and neo-cortical neural models

Unlike classical neural-networks, the holographic neural model achieves a high density storage of information by enfolding a large number of stimulus-response associations within a single neuron cell. HNeT permits to allocate seven types of neuron cells, four of which being modeled after the more predominant cells in the cerebellum and neo-cortex: the granule, stellate, pyramidal and purkinje cells [16]. Each cell may be individually configured with a set of parameters such as the learning rate, memory profile, plasticity etc and can be combined into more complex cell assemblies.

Two neural assemblies were used for this study: the cerebellar (Figure 1) and the so-called neo-cortical architecture (Figure 2). In each cases, stimuli are presented at the receptor level, the cerebellar assembly supporting the sequential presentation of simple stimulus vectors (typically raw images) while the neo-cortical supports the sequential presentation of a more complex vectors made from a stack of data (typically raw images supplemented by features).

#### 2.2 Input processing

The conversion of an input real-valued measurement vector $X$ to a complex-valued stimulus vector $S$ is realized through the application of a non-linear transformation. A typical example for Gaussian distributed measurements is to use a sigmoid form such as:

$$ x_j \rightarrow \lambda_j e^{i\theta_j} $$

$$ \theta_j = 2\pi / (1 + \exp \left( \frac{\mu - x_j}{\sigma} \right) ) $$

where $\mu$ and $\sigma$ are respectively the mean and variance of $X$. Table 1 gives the class of functions that can be used.

<table>
<thead>
<tr>
<th>Group A</th>
<th>Group B</th>
<th>Group C</th>
<th>Group D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real</td>
<td>Real, imaginary normalisation</td>
<td>Sigmoid</td>
<td>Fourier</td>
</tr>
<tr>
<td>Phase, Magnitude</td>
<td>Phase, Magnitude normalisation</td>
<td>Linear</td>
<td>Wavelet</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Histogram</td>
</tr>
</tbody>
</table>

#### 2.3 Basic unit of association

The basic unit of association represents the stimulus-response association within the neuron cell. Mathematically represented as a complex number:

$$ \lambda e^{i\theta_{\text{diff}}} = \lambda e^{(i\theta_2-i\theta_1)} $$

its phase angle measures the difference between the phase orientation of the learning and recall stimuli and its magnitude measures the level of confidence in this association.

#### 2.4 Basic learning

Holographic memory $W$ is formed by the product between the complex valued stimulus vector $S$ and the associated response vector $R$. The result is stored in the cortical memory array $W$ following Hebbian formalism:

$$ W_{t+1} = W_t + S_{t+1} R_{t+1} $$

Given the set of vectors $S$ and $R$ being written as:

$$ S^T = \begin{bmatrix} \lambda_{1,1} e^{i\theta_{1,1}} & \lambda_{1,2} e^{i\theta_{1,2}} & \ldots & \lambda_{n,1} e^{i\theta_{n,1}} \\ \lambda_{1,2} e^{i\theta_{1,2}} & \lambda_{2,2} e^{i\theta_{2,2}} & \ldots & \lambda_{n,2} e^{i\theta_{n,2}} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{1,n} e^{i\theta_{1,n}} & \lambda_{2,n} e^{i\theta_{2,n}} & \ldots & \lambda_{n,n} e^{i\theta_{n,n}} \end{bmatrix} $$

and

$$ R^T = \begin{bmatrix} \lambda_{1,1} e^{i\theta_{1,1}} & \lambda_{2,2} e^{i\theta_{2,2}} & \ldots & \lambda_{n,1} e^{i\theta_{n,1}} \\ \lambda_{1,2} e^{i\theta_{1,2}} & \lambda_{2,2} e^{i\theta_{2,2}} & \ldots & \lambda_{n,2} e^{i\theta_{n,2}} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{1,n} e^{i\theta_{1,n}} & \lambda_{2,n} e^{i\theta_{2,n}} & \ldots & \lambda_{n,n} e^{i\theta_{n,n}} \end{bmatrix} $$

The conversion of an input real-valued measurement vector $X$ to a complex-valued stimulus vector $S$ is realized through the application of a non-linear transformation. A typical example for Gaussian distributed measurements is to use a sigmoid form such as:

$$ x_j \rightarrow \lambda_j e^{i\theta_j} $$

$$ \theta_j = 2\pi / (1 + \exp \left( \frac{\mu - x_j}{\sigma} \right) ) $$

where $\mu$ and $\sigma$ are respectively the mean and variance of $X$. Table 1 gives the class of functions that can be used.

<table>
<thead>
<tr>
<th>Group A</th>
<th>Group B</th>
<th>Group C</th>
<th>Group D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real</td>
<td>Real, imaginary normalisation</td>
<td>Sigmoid</td>
<td>Fourier</td>
</tr>
<tr>
<td>Phase, Magnitude</td>
<td>Phase, Magnitude normalisation</td>
<td>Linear</td>
<td>Wavelet</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Histogram</td>
</tr>
</tbody>
</table>

#### 2.3 Basic unit of association

The basic unit of association represents the stimulus-response association within the neuron cell. Mathematically represented as a complex number:

$$ \lambda e^{i\theta_{\text{diff}}} = \lambda e^{(i\theta_2-i\theta_1)} $$

its phase angle measures the difference between the phase orientation of the learning and recall stimuli and its magnitude measures the level of confidence in this association.

#### 2.4 Basic learning

Holographic memory $W$ is formed by the product between the complex valued stimulus vector $S$ and the associated response vector $R$. The result is stored in the cortical memory array $W$ following Hebbian formalism:

$$ W_{t+1} = W_t + S_{t+1} R_{t+1} $$

Given the set of vectors $S$ and $R$ being written as:

$$ S^T = \begin{bmatrix} \lambda_{1,1} e^{i\theta_{1,1}} & \lambda_{1,2} e^{i\theta_{1,2}} & \ldots & \lambda_{n,1} e^{i\theta_{n,1}} \\ \lambda_{1,2} e^{i\theta_{1,2}} & \lambda_{2,2} e^{i\theta_{2,2}} & \ldots & \lambda_{n,2} e^{i\theta_{n,2}} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{1,n} e^{i\theta_{1,n}} & \lambda_{2,n} e^{i\theta_{2,n}} & \ldots & \lambda_{n,n} e^{i\theta_{n,n}} \end{bmatrix} $$

and

$$ R^T = \begin{bmatrix} \lambda_{1,1} e^{i\theta_{1,1}} & \lambda_{2,2} e^{i\theta_{2,2}} & \ldots & \lambda_{n,1} e^{i\theta_{n,1}} \\ \lambda_{1,2} e^{i\theta_{1,2}} & \lambda_{2,2} e^{i\theta_{2,2}} & \ldots & \lambda_{n,2} e^{i\theta_{n,2}} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{1,n} e^{i\theta_{1,n}} & \lambda_{2,n} e^{i\theta_{2,n}} & \ldots & \lambda_{n,n} e^{i\theta_{n,n}} \end{bmatrix} $$
Equations 5 and 6 represent the stimulus and response fields (i.e. the stack of transposed stimulus and responses vectors \( R \) and \( S \)). Each line of Eq. 5 represents a transposed stimulus vector (raw data possibly augmented with higher order terms generated at the granule cell level). Stimulus vector may consist in the sampling of a time-varying process or may represent another type of measurement taken as a stimulus vector. In image processing, each element of the stimulus vector \( S \) represents the complex value associated to each pixel of the input image. Stimulus vectors can be time varying signals (for video imagery), features computed from raw data or images taken from an image dataset. The term “holographic memory” was used in that context as images are “superimposed” in the cortical memory \( W \) (Eq. 4).

### 2.5 Response recall

The response generated by the presentation of a new stimulus to the receptor cell of a trained memory can be written as:

\[
R = \frac{1}{c} S \cdot W \tag{7}
\]

where the proposed normalisation coefficient \( c \) is defined as:

\[
c = \sum_{i=1}^{n} \lambda_i \tag{8}
\]

### 2.6 Reinforcement learning

Response recall performances obtained with a basic learning on a given training set can be improved to generate almost exact mapping with a few reinforcement learning trials using new stimuli. Reinforcement learning is performed as a three-stage process.

Evaluate the response \( R \) to \( S \):

\[
R = \frac{1}{c} S \cdot W \tag{7}
\]

Evaluate the difference between the generated response \( R \) and the desired response \( R_0 \):

\[
R_{diff} = R_0 - R \tag{8}
\]

Inject the vector difference in the learning process:

\[
W_{new} = W_{prev} + S^T \cdot R_{diff} \tag{9}
\]

The mathematical expression of the reinforcement is obtained by injecting Eq. (8) in Eq. (9):

\[
W_{new} = W_{prev} + S^T \cdot R_0 - S^T \left[ \frac{1}{c} S \cdot W \right] \tag{10}
\]

which can be expressed as:

\[
W_{new} = W_{prev} + S^T \cdot R_0 - H \cdot W_{prev} \tag{11}
\]

with:

\[
H = \frac{1}{c} S^T \cdot S \tag{12}
\]

### 3 Implementation

We implemented the HNeT software within Erdis Imagine 8.7. Several GUIs were developed to simplify the user interaction with the target/no-target datasets, the selection of stimuli, the various settings required by the learning processes and finally the performance analysis of the results.

### 3.1 Image dataset

The image dataset used for this study was built by S. Agarwal, A. Awan and D. Roth from the Dept of Computer Science of the University of Illinois at Urbana-Champaign. This dataset features side views of cars in various environments.

It contains a training set of 1050 images of size 100x40 pels (550 car and 500 non-car images); 170 single-scale test images (200 cars at a scale similar to the training database) and 108 multi-scale images (139 cars at various scales).

Figure 2 shows images extracted from these three datasets.

Figure 2. Object (upper left), non-object (upper right) samples from the training set.

The non-object dataset does not necessarily contain images showing a content completely different from the objects dataset. As an example Figure 3 shows that the non-object dataset can also contain images of car parts or other related objects (such as motorcycles).

Figure 3. Images extracted from the non-object dataset

Figure 4 shows two images (left: 140x266 pels, right: 155x252 pels) extracted from the single scale dataset. One or several cars at a scale similar to those appearing in the training data set may be present in the image and they may be partially occluded or shadowed.

Figure 4 shows two images (left: 140x266 pels, right: 155x252 pels) extracted from the single scale dataset. One or several cars at a scale similar to those appearing in the training data set may be present in the image and they may be partially occluded or shadowed.
3.2 Image dataset explorer

Objects selected by the analyst to populate the target, no-target databases as well as the various features computed for each of them are maintained with the Firebird 1.5 Database Server. Firebird is a relational database offering many ANSI SQL-99 features that runs on Linux, Windows, and a variety of Unix platforms. Firebird offers excellent concurrency, high performance, and powerful language support for stored procedures and triggers. Figure 5 shows the Database Explorer GUI.

<table>
<thead>
<tr>
<th>Table 2. Features available as stimulus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Raw Data</td>
</tr>
<tr>
<td>Law's R5R5</td>
</tr>
<tr>
<td>Sobel R5B</td>
</tr>
<tr>
<td>Mean Euclidian Distance</td>
</tr>
<tr>
<td>LAWD</td>
</tr>
<tr>
<td>Variance</td>
</tr>
<tr>
<td>LAWF</td>
</tr>
<tr>
<td>Skewness</td>
</tr>
<tr>
<td>LAWH</td>
</tr>
<tr>
<td>Kurtosis</td>
</tr>
<tr>
<td>Mean</td>
</tr>
<tr>
<td>Median</td>
</tr>
<tr>
<td>Red, Green, Blue</td>
</tr>
<tr>
<td>Standard Deviation</td>
</tr>
<tr>
<td>I, H, S</td>
</tr>
<tr>
<td>Edge Detection</td>
</tr>
<tr>
<td>Sobel with threshold</td>
</tr>
<tr>
<td>Cross Edge Detection</td>
</tr>
<tr>
<td>Geometric moments</td>
</tr>
<tr>
<td>Laplacian Edge Detection</td>
</tr>
<tr>
<td>Central, Norm. Central</td>
</tr>
</tbody>
</table>

3.3 Classes of stimuli

Upon selection of a target/no-target object and their inclusion in the target/no-target databases, a variety of features capturing texture, spectral, geometric measurements are calculated in addition to the raw data to be potentially used as additional stimulus vectors. Table 2 gives the list of the features currently available, Figure 5 illustrates the results obtained for a few of them.

<table>
<thead>
<tr>
<th>Table 3. Network configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameters</td>
</tr>
<tr>
<td>Network type</td>
</tr>
<tr>
<td>Distribution</td>
</tr>
<tr>
<td>Learning rate</td>
</tr>
<tr>
<td>Epochs</td>
</tr>
<tr>
<td>Transfer function</td>
</tr>
<tr>
<td>Normalization</td>
</tr>
<tr>
<td>Memory decay</td>
</tr>
<tr>
<td>Pruning</td>
</tr>
<tr>
<td>Neural plasticity</td>
</tr>
</tbody>
</table>

When stimuli are formed by raw images only, the network configuration is based on the cerebellar assembly. Table 3 gives the holographic neural configuration used for this experiment.

When stimuli are formed by raw images and features, the network configuration is based on the neo-cortical assembly. Table 3 gives the holographic neural configuration used for this experiment.
An evaluation of the best training parameters (learning rate, number of epochs) has been conducted. The optimisation of two other parameters used in the network configuration (neural plasticity, memory decay) has been left for further work.

4 Experiments

The investigation of the holographic neural technology for incremental learning has been organized in two experiments. Each of these experiments is conducted in two phases; first with a stimulus vector formed with raw images only (using a cerebellar assembly) and second with a stimulus vector formed with raw images supplemented with two features: topological (Edge Detect) and textural (LAWF). This second phase uses a neo-cortical assembly.

The first experiment is focused on object recognition. The incremental learning performances are tested by cross-validation using subsets extracted from the training dataset. All images have the same size and are presented at the receptor cells as a whole (no detection with a sliding window).

The second experiment deals with object detection and recognition. The capability of the network to learn from false and true positive identified after a recall on the test dataset is evaluated with the two learning models (batch and incremental).

Tests are conducted to verify the level of similarity between the different learning models. The persistence of the learned patterns in memory is evaluated by making separate recalls before and after the addition of new patterns in the learning process.

4.1 ATR by cross-validation

The training set is divided into ten subsets. One of the subset is used for validation, the other nine are used for incremental training (each subset contains 105 samples).

This experiment evaluates the two learning models (batch and incremental) separately.

To conduct a cross-validation [16] with batch learning, one subset is designated as the test set TE, the nine others as the training set TR. The classifier is trained with TR and tested with TE. Then, we designate a new subset as the test set TE, and reiterate the training and test steps until the 10 subsets are used as the test set TE. This is called 10-fold cross-validation.

To conduct a cross-validation with incremental learning, one subset is designated as the test set TE, the others as the training sets TRi (i=1,..., 9). The classifier is trained with TR1, trained with TR2, trained with TR3, and so on, and tested with TE. This is repeated until each subset has been designated as the test set. This is a 10-fold cross-validation with incremental learning steps.

A second interesting test is to compare the two learning models as the learning progresses.

The cross-validation with batch learning is obtained by training the network with TRi and using it to classify TE. A new network is trained with (TR1 ∪ TR2) and used to classify TE. The same procedure is repeated for all TRi. The network learns by using all the data available at each time, which is batch learning.

The cross-validation by incremental learning is obtained by training the network with TRi and using it to classify TE. The same network is then trained with TRj, and used to classify TE. The same was done with all TRj. Thus the network has to learn new data without having access to previous training data, which is incremental learning.

These steps are repeated ten times; each time with a different subset selected as the test set (10-fold cross-validation).

4.2 Incremental vs. batch learning for ATD/ATR

The experiment was structured in two parts:

First, the network is trained with the entire training dataset and the response-recall is performed on the training dataset.

Second, on the trained network a response recall is performed on the single-scale test dataset. The true positive (TP) and false positive (FP) detections are identified and added to the objects and non-objects datasets. Two learning methods were tested to train the network to recognize these new samples. The first approach consists in building a new training set by adding to the original training dataset the TP and FP and training a new network in batch mode. The second approach is to incrementally train the network (previously trained with the training dataset) by presenting the TP or FP one at a time, without reusing the original training dataset. The learning optimization performed during the epochs is applied on the entire dataset in the first approach and to each new sample in the second approach. In both approaches, the training data (the training dataset, the TP and the FP) is randomly sorted before being presented to the network.

5 Results

For each of the three experiments, results will be given for each of the two phases (raw data, raw data and features).
5.1 ATR by cross-validation

Figure 4 (resp. 5) shows the recognition results obtained with raw data (resp. raw and features data) for batch learning and incremental learning.

For a perfectly recognized object (resp. non-object) the network will generate an output of 1 (resp. –1). The threshold values, used to generate the ROC curves, correspond to an output value at which we can make a decision of classifying the stimulus as object or non-object.

Figure 4 does not show significative differences between the two learning models.

Figure 5 shows the recognition results obtained with raw data and the two features with a batch and an incremental learning.

As mentioned earlier, Figure 5 does not show significative differences between the two learning models. We can see however that the addition of the two features in the stimulus vectors contributes to increase the number of recognized object and decrease the number of misclassifications.

Figure 6 (resp. 7) shows the comparison of the classification accuracy ((TP+TN)/(TP+FN+FP+TN)) of the two learning models as the learning progress (each learning step is associated to the learning of a new subset) for raw data (resp. raw data plus the two features).

Figures 6 and 7 do not show significant differences between the two models as new patterns are learned. As mentioned earlier better recognition accuracy is achieved when supplementing the raw data with the two features.

5.2 ATD/ATR- Incremental vs. batch learning

Table 5 shows the detection results obtained on the test dataset after training the network N1 on the training dataset in batch mode after five epochs (TP: True Positive, FP: False Positive, FN: False Negative, DR: Detection Rate, FAR: False Alarm Rate, P: Precision).

A new training dataset is formed at this time. It contains the original training dataset to which are added the FP and TP images the network failed to recognize.

Table 6 shows the detection results obtained on the test dataset after training the network N2 on the new training dataset in batch mode after five epochs.
### Table 5. Detection results with network N1

<table>
<thead>
<tr>
<th>Threshold</th>
<th>TP</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.3</td>
<td>196</td>
<td>4</td>
<td>55915</td>
<td>153</td>
</tr>
<tr>
<td>-0.2</td>
<td>195</td>
<td>5</td>
<td>55963</td>
<td>105</td>
</tr>
<tr>
<td>-0.1</td>
<td>194</td>
<td>6</td>
<td>56002</td>
<td>66</td>
</tr>
<tr>
<td>0</td>
<td>193</td>
<td>7</td>
<td>56029</td>
<td>39</td>
</tr>
<tr>
<td>0.1</td>
<td>186</td>
<td>14</td>
<td>56047</td>
<td>21</td>
</tr>
<tr>
<td>0.2</td>
<td>178</td>
<td>22</td>
<td>56058</td>
<td>10</td>
</tr>
<tr>
<td>0.3</td>
<td>164</td>
<td>36</td>
<td>56063</td>
<td>5</td>
</tr>
<tr>
<td>0.4</td>
<td>143</td>
<td>57</td>
<td>56066</td>
<td>2</td>
</tr>
<tr>
<td>0.5</td>
<td>126</td>
<td>74</td>
<td>56066</td>
<td>2</td>
</tr>
<tr>
<td>0.6</td>
<td>100</td>
<td>100</td>
<td>56066</td>
<td>2</td>
</tr>
<tr>
<td>0.7</td>
<td>75</td>
<td>125</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>52</td>
<td>148</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>39</td>
<td>161</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>25</td>
<td>175</td>
<td>56068</td>
<td>0</td>
</tr>
</tbody>
</table>

Tables 5 and 6 show an improvement of the object detection rate (TP increased) and a lower false alarm rate by adding the FP and TP to the training dataset. The network has successfully learned the new objects.

### Table 6. Detection results with network N2

<table>
<thead>
<tr>
<th>Threshold</th>
<th>TP</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.3</td>
<td>198</td>
<td>2</td>
<td>55944</td>
<td>124</td>
</tr>
<tr>
<td>-0.2</td>
<td>197</td>
<td>3</td>
<td>56007</td>
<td>61</td>
</tr>
<tr>
<td>-0.1</td>
<td>197</td>
<td>3</td>
<td>56040</td>
<td>28</td>
</tr>
<tr>
<td>0</td>
<td>197</td>
<td>3</td>
<td>56057</td>
<td>11</td>
</tr>
<tr>
<td>0.1</td>
<td>196</td>
<td>4</td>
<td>56061</td>
<td>7</td>
</tr>
<tr>
<td>0.2</td>
<td>194</td>
<td>6</td>
<td>56064</td>
<td>4</td>
</tr>
<tr>
<td>0.3</td>
<td>189</td>
<td>11</td>
<td>56067</td>
<td>1</td>
</tr>
<tr>
<td>0.4</td>
<td>182</td>
<td>18</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.5</td>
<td>166</td>
<td>34</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.6</td>
<td>150</td>
<td>50</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.7</td>
<td>125</td>
<td>75</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>94</td>
<td>106</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>65</td>
<td>135</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>49</td>
<td>151</td>
<td>56068</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 7. Recall with N1

<table>
<thead>
<tr>
<th>Threshold</th>
<th>TP</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.3</td>
<td>550</td>
<td>0</td>
<td>492</td>
<td>8</td>
</tr>
<tr>
<td>-0.2</td>
<td>550</td>
<td>0</td>
<td>498</td>
<td>2</td>
</tr>
<tr>
<td>-0.1</td>
<td>550</td>
<td>0</td>
<td>498</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>549</td>
<td>1</td>
<td>499</td>
<td>1</td>
</tr>
<tr>
<td>0.1</td>
<td>549</td>
<td>1</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.2</td>
<td>547</td>
<td>3</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.3</td>
<td>543</td>
<td>7</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.4</td>
<td>532</td>
<td>18</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.5</td>
<td>511</td>
<td>39</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.6</td>
<td>488</td>
<td>62</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.7</td>
<td>443</td>
<td>107</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>377</td>
<td>173</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>289</td>
<td>261</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>205</td>
<td>345</td>
<td>500</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 8. Recall with N2

<table>
<thead>
<tr>
<th>Threshold</th>
<th>TP</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.3</td>
<td>550</td>
<td>0</td>
<td>494</td>
<td>6</td>
</tr>
<tr>
<td>-0.2</td>
<td>550</td>
<td>0</td>
<td>498</td>
<td>2</td>
</tr>
<tr>
<td>-0.1</td>
<td>549</td>
<td>1</td>
<td>499</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>549</td>
<td>1</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.1</td>
<td>547</td>
<td>3</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.2</td>
<td>546</td>
<td>4</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.3</td>
<td>540</td>
<td>10</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.4</td>
<td>522</td>
<td>28</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.5</td>
<td>508</td>
<td>42</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.6</td>
<td>474</td>
<td>76</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.7</td>
<td>431</td>
<td>119</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>361</td>
<td>189</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>284</td>
<td>266</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>205</td>
<td>345</td>
<td>500</td>
<td>0</td>
</tr>
</tbody>
</table>

### Table 9. Detection results with incremental learning on network N1

<table>
<thead>
<tr>
<th>Threshold</th>
<th>TP</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.3</td>
<td>197</td>
<td>3</td>
<td>55892</td>
<td>176</td>
</tr>
<tr>
<td>-0.2</td>
<td>197</td>
<td>3</td>
<td>55958</td>
<td>110</td>
</tr>
<tr>
<td>-0.1</td>
<td>197</td>
<td>3</td>
<td>55998</td>
<td>70</td>
</tr>
<tr>
<td>0</td>
<td>197</td>
<td>3</td>
<td>56039</td>
<td>29</td>
</tr>
<tr>
<td>0.1</td>
<td>195</td>
<td>5</td>
<td>56053</td>
<td>15</td>
</tr>
<tr>
<td>0.2</td>
<td>193</td>
<td>7</td>
<td>56060</td>
<td>8</td>
</tr>
<tr>
<td>0.3</td>
<td>188</td>
<td>12</td>
<td>56065</td>
<td>3</td>
</tr>
<tr>
<td>0.4</td>
<td>180</td>
<td>20</td>
<td>56067</td>
<td>1</td>
</tr>
<tr>
<td>0.5</td>
<td>169</td>
<td>31</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.6</td>
<td>152</td>
<td>48</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.7</td>
<td>124</td>
<td>76</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>88</td>
<td>112</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>64</td>
<td>136</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>47</td>
<td>153</td>
<td>56068</td>
<td>0</td>
</tr>
</tbody>
</table>

Tables 7 and 8 show that the memory persistence of the network is intact.

It would be now be interesting to verify this property when adding incrementally the objects the network failed to recognize in the first place.

Table 9 shows the detection results obtained on the test dataset after training incrementally the network N1 with the FP and TP.

### Table 10. Recall after the incremental learning on network N1

<table>
<thead>
<tr>
<th>Threshold</th>
<th>TP</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.3</td>
<td>197</td>
<td>3</td>
<td>55892</td>
<td>176</td>
</tr>
<tr>
<td>-0.2</td>
<td>197</td>
<td>3</td>
<td>55958</td>
<td>110</td>
</tr>
<tr>
<td>-0.1</td>
<td>197</td>
<td>3</td>
<td>55998</td>
<td>70</td>
</tr>
<tr>
<td>0</td>
<td>197</td>
<td>3</td>
<td>56039</td>
<td>29</td>
</tr>
<tr>
<td>0.1</td>
<td>195</td>
<td>5</td>
<td>56053</td>
<td>15</td>
</tr>
<tr>
<td>0.2</td>
<td>193</td>
<td>7</td>
<td>56060</td>
<td>8</td>
</tr>
<tr>
<td>0.3</td>
<td>188</td>
<td>12</td>
<td>56065</td>
<td>3</td>
</tr>
<tr>
<td>0.4</td>
<td>180</td>
<td>20</td>
<td>56067</td>
<td>1</td>
</tr>
<tr>
<td>0.5</td>
<td>169</td>
<td>31</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.6</td>
<td>152</td>
<td>48</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.7</td>
<td>124</td>
<td>76</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>88</td>
<td>112</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>64</td>
<td>136</td>
<td>56068</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>47</td>
<td>153</td>
<td>56068</td>
<td>0</td>
</tr>
</tbody>
</table>

Tables 5 and 9 show an improvement of the object detection rate (TP increased) and a lower false alarm rate when the network learns incrementally the objects it initially failed to recognize. Moreover, Tables 6 and 9 show that the detections capability of the network that has incrementally learned the new objects are similar to the performance of a network initially trained with these new objects.

Table 10 shows the results of the recall of the training set by N1 after the incremental learning of new objects. The purpose of this Table is to verify that the network N1 kept the same recall capability it had before learning the new samples.
Table 10. Training set recall after incremental learning

<table>
<thead>
<tr>
<th>Threshold</th>
<th>TP</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.3</td>
<td>550</td>
<td>0</td>
<td>484</td>
<td>16</td>
</tr>
<tr>
<td>-0.2</td>
<td>550</td>
<td>0</td>
<td>493</td>
<td>7</td>
</tr>
<tr>
<td>-0.1</td>
<td>549</td>
<td>1</td>
<td>498</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>549</td>
<td>1</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.1</td>
<td>547</td>
<td>3</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.2</td>
<td>540</td>
<td>10</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.3</td>
<td>529</td>
<td>21</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.4</td>
<td>515</td>
<td>35</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.5</td>
<td>489</td>
<td>61</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.6</td>
<td>453</td>
<td>97</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.7</td>
<td>402</td>
<td>148</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.8</td>
<td>331</td>
<td>219</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>0.9</td>
<td>256</td>
<td>294</td>
<td>500</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>171</td>
<td>379</td>
<td>500</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 10 and Table 7 do not show significant changes in the recall performances.

Figure 8 shows typical FP and TP detections obtained by the network that has not learned them yet (left) and after having learned them (right).

![Figure 8. Detection before (left) and after (right) incremental learning.](image)

6 Conclusion

The objective of this paper was to evaluate the capability of the Holographic Neural Technology to incrementally learn from its successes and from its failures in order to get increasingly smarter while keeping its initial recall capability intact. The results presented in this paper are conclusive and positively support this hypothesis. Improvements can certainly be made by searching for the network parameters and the feature set that would optimize the detection and recognition performances. This investigation is the object of our current activities.

References