Introduction

This project presents a pattern recognition task, based on multiple patterns. The proposed solution employs an Adaptive Resonance Theory (ART) network to recognize patterns, given certain initial conditions.

I have employed Jupyter Notebook, which is a web-based interactive environment that allows you to write and execute code. The platform is especially designed and optimized for data science, machine learning, and data analysis applications.

For implementation, I have made use of Python programming language along with data processing and manipulation libraries such as NumPy & pandas. In addition, for the purpose of data visualization – matplotlib library has been utilized.

The entire ML workflow has been adopted: dataset generation, model building, model evaluation, model optimization & results. These are discussed in depth in the subsequent sections.

Methodology & Justification

I. Adaptive Resonance Theory

Adaptive resonance theory (ART) is a type of neural network that employs unsupervised learning. They are open to new learning i.e., adaptive without discarding previous information i.e., resonance. This is phenomena is also referred to as stability-plasticity dilemma. In other words, they can learn new patterns without forgetting old ones. It is generally employed for pattern recognition and clustering applications.
The basic model consists of:

- F1 layer or the comparison layer (inputs are processed)
- F2 layer or the recognition layer (consists of clustering units/processing element)
- The Reset Module (acts as a control mechanism)

The F1 layer gets the external input and passes it to the F2 layer, which is responsible for coordinating it to a classification category. This outcome is given back to the F1 layer to find out whether the category coordinates the input vector. If there is a match, then a new input vector is read, otherwise a new category/cluster is formed.

There exist two sets of weighted interconnections for controlling the degree of similarity between the units in the F1 and the F2 layer. The reset unit makes the decision whether the cluster unit is allowed to learn the input pattern depending on how similar its top-down weight vector is to the input vector and to the decision. This is called the vigilance test. There are three classes of the ART Network: ART 1, ART 2 & ART 3. For this assignment, I have used ART 1, which is designed for sequences of binary input patterns.

II. Data Generation & Analysis

i. Clean Patterns

Dataset has been generated using list slicing to form each pattern. There are 20 clean patterns i.e., alphabets A-T, as represented in the problem statement, with each having dimensions of 8x8. The patterns are of gray-scale representation i.e., having a single channel for intensity values, and have been populated using binary values i.e., 1 & 0.
These are then converted to arrays and displayed as a matrix in a new figure window. Units with values 1 represent light regions and 0 depict dark areas.
Figure 1. represents 'clean' patterns generated.

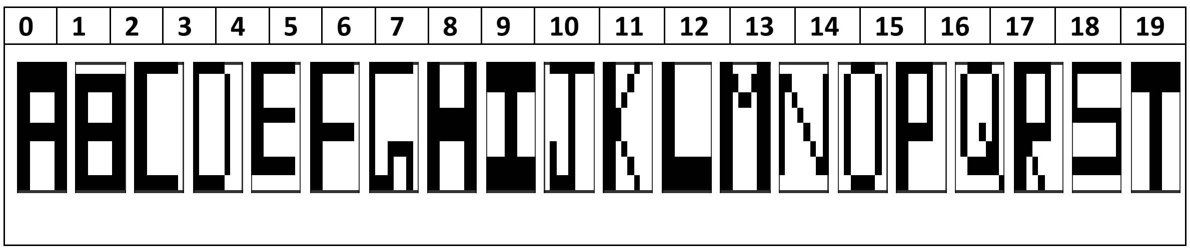

Figure 1.

The table below represents certain characteristics of each pattern that is vital for future reference.

Table 1. Pattern Characteristics

Number of Patterns	Rows per Pattern	Columns per Patterns	Number of Features/Pixels
20	8	8	64

Each pattern is independently produced and then stored in a single array to be later fed to the network for recognition.

ii. Noisy Patterns

As per specifications mentioned in the problem statement, I have generated noisy patterns associated with each clean pattern by introducing random noise. This is achieved by reversing values of 10 - 25% of pixels/units per pattern.
A similar process as that for the clean patterns is adopted. Once generated, they are collectively stored in a single array for further usage.

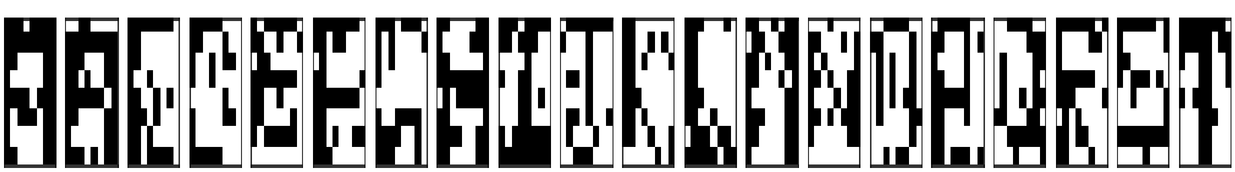

Figure 2. Noise induced pattern i.e., 12.5% noise.

Table 2. represents the set of noisy patterns generated and impacted unit values in comparison to the base model.

Table 2. Noisy Pattern Characteristics

Index	Noise Added (%)	Units/Elements Reversed	Total Features/Elements
Noisy_Pattern#1	12.5%	8	64
Noisy_Pattern#2	25%	16	64

.

III. Data Preprocessing

Under this section, we look at the preprocessing techniques that were employed to transform the dataset into clean and standardized representation for feeding the network.

I. Binary Nature

ART 1 only works with binary input sequences and requires the dataset be transformed accordingly, if not in binary format. In other words, it can work with discrete data. The following command ensures this is achieved in my code:

```
if np.any((X != 0) & (X != 1)):
raise ValueError("ART1 Network works only with binary matrices")
```

II. Flattening

Using the *np.flatten() or np.reshape(-1)* method of the NumPy library, I have reshaped the original arrays of shape (8,8) to (64,). This enables easier interpretation by the network and allows for faster processing due to its single dimensionality.

```
arr_A = pattern_A.reshape(-1)
```

IV. Model Building & Evaluation

Under this section, the network is built and evaluated for several scenarios. The learning process is unsupervised and based on the principles of clustering algorithm. In summary, input is presented to the network and the algorithm checks whether it fits into one of the already stored clusters. If it fits, then the input is added to the cluster that matches the most else a new cluster is formed.

We evaluate the performance based on correct recognition/association of the input pattern with the stored patterns, otherwise, the creation of new category/cluster. An important parameter is the ‘Vigilance’ parameter, defined as the similarity between the top-down template and the input vector. It can be represented using the following equation:

$$\frac{|S|}{|I|} > \rho$$

Where S = T-D Template, I = Input Vector & 0 < p > 1

There are two scenarios for the vigilance test:

a. Vigilance test is passed. Update T-D and B-U weights for both layers, for the associated cluster.
b. Vigilance test is failed. Repeat the earlier steps and compare the input with remaining clusters to choose one. If none passes the test, create a new category, and do not disrupt weights for the other stored clusters.

For higher values of vigilance, the network can distinctly categorize input patterns into new categories. This is a desired attribute but may lead to overfitting. On the other hand, a smaller value means patterns can be incorrectly associated with similar but different patterns. Thus, it is imperative to choose the value of the vigilance parameter according to the problem specifications.

I employ the accuracy metric for evaluation of the network for clean patterns. Accuracy is one of the metrics employed for evaluating classification models.
It is represented as:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

The following sub-sections highlight model training and evaluation, and the subsequent analysis of different aspects.

i. Clean Patterns

I now conduct a series of tests (model runs) for the originally generated patterns. The performance of the model is evaluated for all clean patterns. Note: Apart from the vigilance level, all other model hyperparameters have been kept constant.

Table 3. represents the metrics recorded:

Table 3. Clean Pattern Recall

Input Pattern Number	**Clustering Vigilance Level**								**Desired Resonance Cluster**
	0.3	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**0.95**	
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	1
2	0	0	1	1	2	2	2	2	2
3	1	1	2	2	3	3	3	3	3
4	2	2	3	3	4	4	4	4	4
5	0	0	4	4	5	5	5	5	5
6	3	3	5	5	6	6	6	6	6
7	0	0	0	6	7	7	7	7	7
8	4	4	6	7	8	8	8	8	8
9	5	5	7	8	9	9	9	9	9
10	6	6	8	9	10	10	10	10	10
11	7	7	9	10	11	11	11	11	11
12	0	8	10	6	7	12	12	12	12
13	8	9	11	11	12	13	13	13	13
14	9	10	12	12	13	14	14	14	14
15	10	11	4	4	5	5	5	15	15
16	11	12	13	13	14	15	15	16	16
17	12	13	14	14	15	16	16	17	17
18	13	14	15	15	16	17	17	18	18
19	4	4	6	7	17	18	18	19	19
Accuracy	**70%**	**75%**	**80%**	**80%**	**90%**	**95%**	**95%**	**100%**	

We can state the following points about the network's performance for a series of vigilance levels:

- In terms of patterns correctly associated/clustered, we observe that with increasing vigilance level, the categorizations become more distinct.
- The least accuracy achieved is for the lowest vigilance level i.e., 0.3. Five patterns have been incorrectly associated.
- The greatest accuracy achieved is for the highest vigilance level i.e., 0.95. All patterns are correctly recognized.
- The learning/training time is constant for all iterations, thus depicting vigilance levels does not impact it.

ii. Noisy Patterns – 12.5% Noise

We look at a similar configuration for noisy patterns generated with a noise level of 12.5%. These 20 patterns generated are fed to the ART network along with the original 'clean' patterns for comparative evaluation. Table 4. represents the metrics recorded:

Table 4. Noisy Pattern Recall

Input Pattern Number	**Clustering Vigilance Level**								**Desired Resonance Cluster**
	0.3	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**0.95**	
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	1
2	0	0	1	1	2	2	2	2	2
3	1	1	1	1	2	3	3	3	3
4	1	1	1	1	3	2	4	4	4
5	0	0	1	2	2	4	5	5	5
6	1	1	2	2	4	3	6	6	6
7	0	0	0	3	5	4	7	7	7
8	1	2	2	4	4	5	8	8	8
9	2	2	3	5	6	6	9	9	9
10	2	1	3	2	3	7	10	10	10
11	2	1	1	1	7	7	11	11	11
12	0	3	4	3	5	8	12	12	12
13	3	3	4	5	8	9	13	13	13
14	2	4	3	6	6	10	14	14	14
15	3	3	5	6	2	11	5	15	15
16	2	4	5	7	9	10	15	16	16
17	3	1	1	6	3	11	10	17	17
18	3	4	6	7	8	12	16	18	18
19	4	2	2	4	10	6	17	19	19
0'	0	0	5	0	0	13	18	20	0
1'	5	3	4	8	1	1	19	21	1
2'	2	1	3	9	7	14	2	22	2
3'	3	5	6	10	11	15	3	23	3
4'	3	4	6	9	11	15	20	24	4
5'	3	5	7	10	12	16	21	25	5
6'	2	2	2	11	13	17	6	26	6
7'	0	5	4	3	14	9	22	27	7
8'	1	2	2	4	10	5	9	28	8
9'	4	6	7	5	6	18	23	9	9
10'	2	6	8	2	15	19	24	29	10
11'	2	6	8	12	7	19	25	30	11
12'	5	7	8	8	9	8	26	31	12
13'	5	7	9	12	16	20	27	32	13
14'	4	8	9	13	17	21	14	33	14
15'	5	7	7	10	12	22	28	34	15
16'	4	8	9	13	17	10	15	35	16
17'	3	7	6	11	15	22	29	36	17
18'	6	8	10	14	16	12	30	37	18
19'	4	9	10	14	18	23	31	38	19

Noisy Pattern A

Noisy Pattern_T

iii. Noisy Patterns – 25% Noise

Configuration for noisy patterns generated with a noise level of 25%. All 20 patterns are fed to the ART network along with the original 'clean' patterns for comparative evaluation. This is depicted in the table below.

Table 5. Noisy Pattern Recall

Input Pattern Number	**Clustering Vigilance Level**								**Desired Resonance Cluster**
	0.3	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**0.95**	
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	1
2	0	0	1	1	2	2	2	2	2
3	1	1	1	1	2	3	3	3	3
4	1	1	1	1	3	2	4	4	4
5	0	0	1	2	2	4	5	5	5
6	1	1	2	2	4	3	6	6	6
7	0	0	0	3	5	4	7	7	7
8	1	2	2	4	4	5	8	8	8
9	2	2	3	5	6	6	9	9	9
10	2	1	3	2	3	7	10	10	10
11	2	1	1	1	7	7	11	11	11
12	0	3	4	3	5	8	12	12	12
13	3	3	4	5	8	9	13	13	13
14	2	4	3	6	6	10	14	14	14
15	3	3	5	6	2	11	5	15	15
16	2	4	5	7	9	10	15	16	16
17	3	1	1	6	3	11	10	17	17
18	3	4	6	7	8	12	16	18	18
19	4	2	2	4	10	6	17	19	19
0'	0	2	3	8	7	13	18	20	0
1'	2	4	6	9	9	14	19	21	1
2'	4	5	7	10	11	15	20	22	2
3'	5	5	7	11	12	16	3	23	3
4'	3	6	5	12	13	17	21	24	4
5'	5	6	8	10	14	18	22	25	5
6'	4	6	8	12	15	19	23	26	6
7'	6	7	4	13	16	20	24	27	7
8'	6	7	2	4	10	21	25	28	8
9'	5	8	9	14	6	22	26	29	9
10'	7	8	9	14	14	23	27	30	10
11'	7	9	10	15	17	24	28	31	11
12'	7	8	10	16	18	9	29	32	12
13'	8	9	11	16	19	25	30	33	13
14'	8	10	11	17	20	26	31	34	14
15'	9	10	12	18	21	27	32	35	15
16'	8	11	12	17	22	28	33	36	16
17'	3	3	13	18	21	29	34	37	17
18'	9	12	13	19	23	12	35	38	18
19'	10	11	14	20	22	30	36	39	19

The following observations are deduced:

- Using a ρ < 0.8, the network is unable to categorize the original 20 patterns clearly.
- For lower values of vigilance, both tables revealed that the network was able to associate some of the noisy patterns with the original ones.
- However, it also showed that the network did poorly to separate the original patterns from one another.
- For higher values of vigilance, the performance in terms of mapping each pattern to a specific cluster was seen.
- At the same time, the model mis-categorized the noisy patterns as being entirely distinct.
- We also observed the impact of top-down LTM traces or the connection weights between the F2 layer and F1 layer, on the network's performance.
- Results revealed that for values near 1, the ART network was continuously overfitting, regardless of the vigilance levels. This can be attributed to the network weights being rapidly polarized, causing the learning process to stagnate after a few iterations.
- For values less than 0.5, the network was more responsive and was able to associate similar patterns to the same clusters, but it also meant incorrect associations of distinct patterns.
- Thus, the importance of T-D LTM traces initialization values is significant in terms of the network's performance.
- I also studied the impact of shuffling the dataset on the network's ability to correctly recognize patterns. There is a large deviation of results obtained, when compared with the original unshuffled dataset.
- In terms of the training time required, there was no significant variability noted in all runs of the network.
- Finally, the network was able to successfully learn all patterns for high vigilance values whilst ensuring previous clusters were not impacted. This reaffirms our initial hypothesis about the ability of ART network in resolving the stability-plasticity dilemma.

Conclusion

To conclude, the ART 1 network is a complex self-organizing network that can handle many patterns for successfully storing in the network as determined by this assignment. Furthermore, it is somewhat robust to noise and can distinguish between clean and noise-induced patterns given the right parameter values. The role of the vigilance parameter in terms of fine-tuning the degree to which the network can recognize correctly, is linear. However, there exists a trade-off between accurate recognition and overfitting – since the model does not generalize well with outliers. The answer to this question lies with the problem at hand and the dynamics associated with it.

Ultimately, ART 1 can learn using binary-input patterns with limited training time required. It is an efficient algorithm that can overcome problems other neural networks fail to handle i.e., stability-plasticity issue. The results achieved through this assignment reinforce the proposed analysis.