SL II - Experiment B2

Aim: Write a python program to illustrate ART neural network.

Objective: To be able to implement ART Neural Network

Theory:

Adaptive Resonance Theory (ART) Adaptive resonance theory is a type of neural network technique developed by Stephen Grossberg and Gail Carpenter in 1987. The basic ART uses unsupervised learning technique. The term **"adaptive"** and **"resonance"** used in this suggests that they are open to new learning(i.e. adaptive) without discarding the previous or the old information(i.e. resonance). The ART networks are known to solve the stability-plasticity dilemma i.e., stability refers to their nature of memorizing the learning and plasticity refers to the fact that they are flexible to gain new information. Due to this the nature of ART they are always able to learn new input patterns without forgetting the past. ART networks implement a clustering algorithm. 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.

Types of Adaptive Resonance Theory(ART) Carpenter and Grossberg developed different ART architectures as a result of 20 years of research. The ARTs can be classified as follows:

- **ART1 –** It is the simplest and the basic ART architecture. It is capable of clustering binary input values.
- **ART2 –** It is extension of ART1 that is capable of clustering continuous-valued input data.
- **Fuzzy ART –** It is the augmentation of fuzzy logic and ART.
- **ARTMAP –** It is a supervised form of ART learning where one ART learns based on the previous ART module. It is also known as predictive ART.
- **FARTMAP –** This is a supervised ART architecture with Fuzzy logic included.

Basic of Adaptive Resonance Theory (ART) Architecture The adaptive resonant theory is a type of neural network that is self-organizing and competitive. It can be of both types, the unsupervised ones(ART1, ART2, ART3, etc) or the supervised ones(ARTMAP). Generally, the supervised algorithms are named with the suffix "MAP". But the basic ART model is unsupervised in nature and consists of :

- F1 layer or the comparison field(where the inputs are processed)
- F2 layer or the recognition field (which consists of the clustering units)
- The Reset Module (that acts as a control mechanism)

The **F1 layer** accepts the inputs and performs some processing and transfers it to the F2 layer that best matches with the classification factor. There exist **two sets of weighted interconnection** for controlling the degree of similarity between the units in the F1 and the F2 layer. The **F2 layer** is a competitive layer. The cluster unit with the large net input becomes the candidate to learn the input pattern first and the rest F2 units are ignored. The **reset unit** makes the decision whether or not 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. Thus we can say that the **vigilance parameter** helps to incorporate new memories or new information. Higher vigilance produces more detailed memories, lower vigilance produces more general memories.

Generally **two types of learning** exists,slow learning and fast learning. In fast learning, weight update during resonance occurs rapidly. It is used in ART1.In slow learning, the weight change occurs slowly relative to the duration of the learning trial. It is used in ART2.

Advantage of Adaptive Resonance Theory (ART)

- It exhibits stability and is not disturbed by a wide variety of inputs provided to its network.
- It can be integrated and used with various other techniques to give more good results.
- It can be used for various fields such as mobile robot control, face recognition, land

cover classification, target recognition, medical diagnosis, signature verification, clustering web users, etc.

- It has got advantages over competitive learning (like bpnn etc). The competitive learning lacks the capability to add new clusters when deemed necessary.
- It does not guarantee stability in forming clusters.

Limitations of Adaptive Resonance Theory Some ART networks are inconsistent (like the Fuzzy ART and ART1) as they depend upon the order of training data, or upon the learning rate

Program :

```
import numpy as np

def initialize_weights(input_dim, category):
   weights = np.random.uniform(size=(input_dim,))
   weights /= np.sum(weights)
   return weights

def calculate_similarity(input_pattern, weights):
   return np.minimum(input_pattern, weights).sum()

def update_weights(input_pattern, weights, vigilance):
   while True:
      activation = calculate_similarity(input_pattern, weights)
      if activation >= vigilance:
         return weights
      else:
         weights[np.argmax(input_pattern)] += 1
         weights /= np.sum(weights)

def ART_neural_network(input_patterns, vigilance):
   num_patterns, input_dim = input_patterns.shape
   categories = []
   for pattern in input_patterns:
      matched_category = None
      for category in categories:
         if calculate_similarity(pattern, category["weights"]) >= vigilance:
            matched_category = category
            break

      if matched_category is None:
         weights = initialize_weights(input_dim, len(categories))
         matched_category = {"weights": weights, "patterns": []}
         categories.append(matched_category)
      matched_category["patterns"].append(pattern)
      matched_category["weights"] = update_weights (pattern, matched_category ["weights"],
vigilance)
```

```
    return categories

# Example usage
input_patterns = np.array([[1, 0, 1, 0], [0, 1, 0, 1], [1, 1, 1, 0]])
vigilance = 0.5

categories = ART_neural_network(input_patterns, vigilance)

# Print the learned categories
for i, category in enumerate(categories):
    print(f"Category {i+1}:")
    print("Patterns:")
    [print(pattern) for pattern in category["patterns"]]
    print("Weights:")
    print(category["weights"])
    print()
```

Output:

```
Category 1:
Patterns:
[1 0 1 0]
[1 1 1 0]
Weights:
[0.34996657 0.19578498 0.23542308 0.21882537]

Category 2:
Patterns:
[0 1 0 1]
Weights:
[0.19342943 0.21428627 0.18156258 0.41072172]
```

Installation

art-python requires Python (2.7, 3.4, or 3.5) along with several Python package dependencies. Information on installing and using Python can be found at https://www.python.org/. Python distributions, such as Anaconda, are recommended to manage the Python interface.

art-python can be installed using pip, git, or a downloaded zip file. Note that the pip installation will NOT include the examples folder referenced in this manual.

pip: To install art-python using pip:

pip install art-python

Required Python package dependencies include:

- Pandas [Mcki13]: used to analyze and store time series data, http://pandas.pydata.org/
- Numpy [VaCV11]: used to support large, multi-dimensional arrays and matrices, http://www.numpy.org/

Assignment No. B-5

Title: Python program to implement CNN object detection. Discuss numerous performance evaluation metrics for evaluating the object detecting algorithms' performance.

Aim: Write Python program to implement CNN object detection. Discuss numerous performance evaluation metrics for evaluating the object detecting algorithms' performance.

Theory:

Object detection is a computer vision technique for locating instances of objects in images or videos. Object detection algorithms typically leverage machine learning or deep learning to produce meaningful results. When humans look at images or video, we can recognize and locate objects of interest within a matter of moments. The goal of object detection is to replicate this intelligence using a computer.

Average Precision (AP) and mean Average Precision (mAP) are the most popular metrics used to evaluate object detection models, such as Faster RLCNN, Mask R-CNN, and YOLO, among others.

Definition of terms:

- True Positive (TP) — Correct detection made by the model.
- False Positive (FP) — Incorrect detection made by the detector.
- False Negative (FN) — A Ground-truth missed (not detected) by the object detector.
- True Negative (TN) —This is the background region correctly not detected by the model. This metric is not used in object detection because such regions are not explicitly annotated when preparing the annotations.

Object Detection metrics:

Intersection over Union (loU):

loU metric in object detection evaluates the degree of overlap between the ground(gt) truth and prediction(pd). The ground truth and the prediction can be of any shape-rectangular box, circle, or irregular shape). It is calculated as follows:

$$IoU = area(gt \cap pd) / area(gt \cup pd)$$

Diagrammatically, loU is defined as follows (the area of the intersection divided by the area of union between ground-truth and predicted box.

$$IOU = \text{area of overlap} / \text{area of union}$$

loU ranges between 0 and 1, where 0 shows no overlap, and 1 means perfect overlap between gt and pd. IoU metric is useful through thresholding; that is, we need a threshold (a, say) to determine whether detection is correct.

Precision and Recall

Precision is the degree of exactness of the model in identifying only relevant objects. It is the ratio of TPs over all detections made by the model.

Recall measures the ability of the model to detect all ground truths— proposition of TPs among all ground truths.

$$P = TP/(TP+FP) = TP/ \text{ (All detections)}$$

$$R = TP/ (TP+FN) = TP/ \text{ (All ground-truths)}$$

A model is said to be good if it has high precision and high recall. A perfect model has zero FNs and zero FPs (precision=1 and recall=1). Often, attaining a perfect model is not feasible.

Average Precision

AP@a is Area Under the Precision-Recall Curve(AUC-PR) evaluated at a IoU threshold. Formally, it is defined as follows.

$$AP@a = \int_0^1 p(r)\, dr$$

Notation: AP@a or APa means that AP precision is evaluated at o IoU threshold. If you see metrics like AP50 and A75, they mean AP calculated at loU=0.5 and loU=0.75, respectively.

Mean Average Precision (mAP)

Remark (AP and the number of classes): AP is calculated individually for each class. This means that there are as many AP values as the number of classes (loosely). These AP values are averaged to obtain the mean Average Precision (mAP) metric.

Definition: The mean Average Precision (mAP) is the average of AP values over all classes.

$$\text{mAP@}\alpha = \frac{1}{n}\sum_{i=1}^{n} \text{AP}_i \quad \text{for n classes.}$$

For more details, students may refer: https://pyimagesearch.com/2020/07/13/r-cnn-object-detection-with-keras-tensorflow-and-deep-learning/

Conclusion:

We have successfully implemented CNN object detection. Discuss numerous performance evaluation metrics for evaluating the object detecting algorithms' performance.