University of Dayton

***School of Engineering***

***Department of Electrical and Computer Engineering***

**ECE 477/595**: Artificial Neural Networks

# **Project 5:** ART based Image Compression

**Instructor:** Prof. K. Asari Vijayan

**Student Name**: Serge Alhalbi **Student ID:** 101682971

**Date:** 10/30/2022

## Methodology

### Loading Steps

Three Images from MATLAB have been imported and used. The “rgb2gray” function was used to convert rgb images to gray images.

## Adaptive Resonance Theory Algorithm

1. Input $X^s; s = 1,2,\dots,P$.

2. Choose a vigilance parameter, a learning rate, and a choice parameter:
   $0 < \rho < 1; 0 < \beta \leq 1$.
   $\alpha \approx \varepsilon$ is a choice parameter to avoid dividing by 0.

3. Complement coding:
   $X^s := [X^s, 1 - X^s]; s = 1,2,\dots,P$.
   This solves the problem of weights dropping after being updated.

4. Initialize weights:
   $w_{ji};\ i = 1,2,\dots,N; j = 1$
   The weights could be initialized with values equal to one, or with zero values like in the implementation.

5. Loop over the training dataset: $X^s; s = 1,2,\dots,P$

   a. Compute the choice function for each node:
      $$CF_j = \frac{\sum_{i=1}^{N} w_{ji} \wedge X_i^s}{\alpha + \sum_{i=1}^{N} w_{ji}}; j = 1,2,\dots,M.$$

   b. Obtain the winner node:
      $$J = \max_j CF_j$$

   c. Compute the category choice:
      $$CC_J = \frac{\sum_{i=1}^{N} w_{ji} \wedge X_i^s}{\alpha + \sum_{i=1}^{N} X_i^s}$$

   d. Test the vigilance:
      i. If $CC_J \geq \rho$
         $w_{Ji}^{New} = \beta \cdot w_{Ji}^{Old} \wedge X_i^s + (1 - \beta) \cdot w_{Ji}^{Old}; i = 1,2,\dots,N.$

      ii. If $CC_J < \rho$
         Reset node $J$, then find the next largest $CF_j$ and name its winner node as $J$.
         Repeat step 8 until $CC_J \geq \rho$ if possible.

   e. If all existing $M$ nodes fail:
      Create a new node $M = M + 1$.
      The weights of new class will take the values of the pattern $X^s$.

    f. Update the labels for the class chosen:
       The class will have labels recorded from the training dataset.
       Continue looping over the training dataset until it’s done.

6. Record and save all the output labels:
   Each class will have labels recorded from the training dataset.

7. Repeat steps 5 to 6 until the new output labels are the same as the old ones: (Convergence)
   This way, the number of classes is fixed, and the patterns remain fixed in their classes. This loop can be skipped because it will take so long to converge even for a small image.

## Compression Algorithm

1. Input image with size $(M \times N)$.

2. Choose the block size $(m \times n)$. $(m = n)$ if square block.

3. Create the empty compression image matrix of size $\left(m \times n, \frac{M}{m} \times \frac{N}{n}\right)$.

4. Apply the ART algorithm to get the blocks (patterns) in the categories.

5. Take the first pattern from each category and save them to create the code book.

6. Create the block code that will be used to reconstruct the image by replacing all the blocks that belong to a category by its corresponding pattern from the code book.

7. Form a convenient low pass filter.

8. Compute the mean squared error:
   $$MSE = \frac{1}{M \cdot N} \sum_{i=1}^{M} \sum_{j=1}^{N} (Image - Filter)^2$$

9. Compute the peak signal to noise ratio:
   $$PSNR = 10 \cdot log_{10}\left(\frac{1}{MSE}\right)$$

10. Compute the compression ratio:
    $$CR = \frac{M \cdot N}{(m^2 + 1)(number\ of\ categories)}$$

11. Plot the MSE, PSNR, and CR versus the vigilance parameter.

## Results

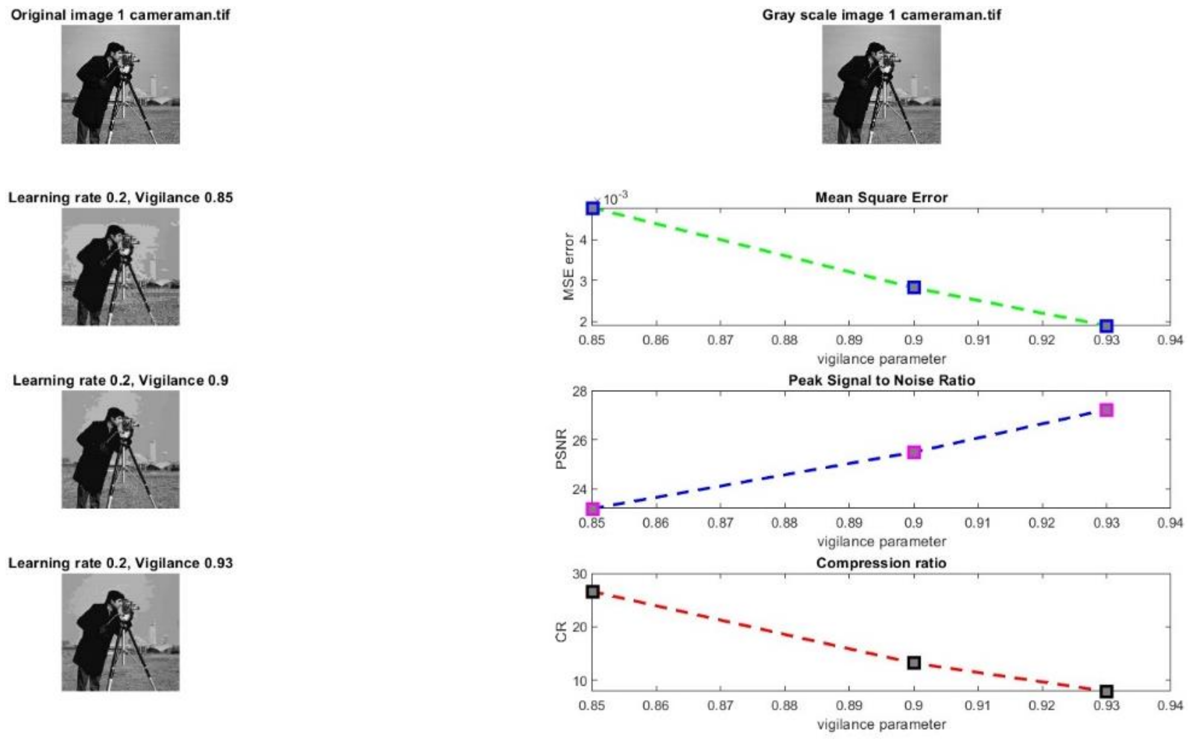

*Figure 1: Cameraman Image Compression and Metrics*

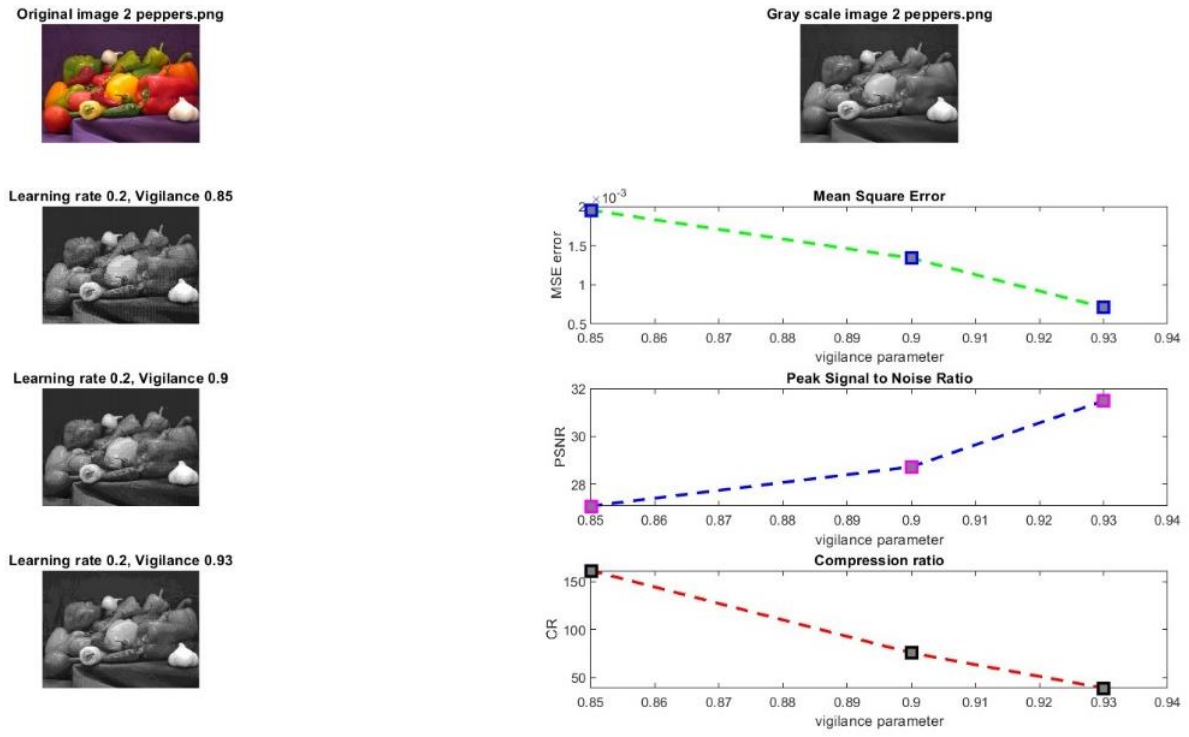

*Figure 2: Peppers Image Compression and Metrics*

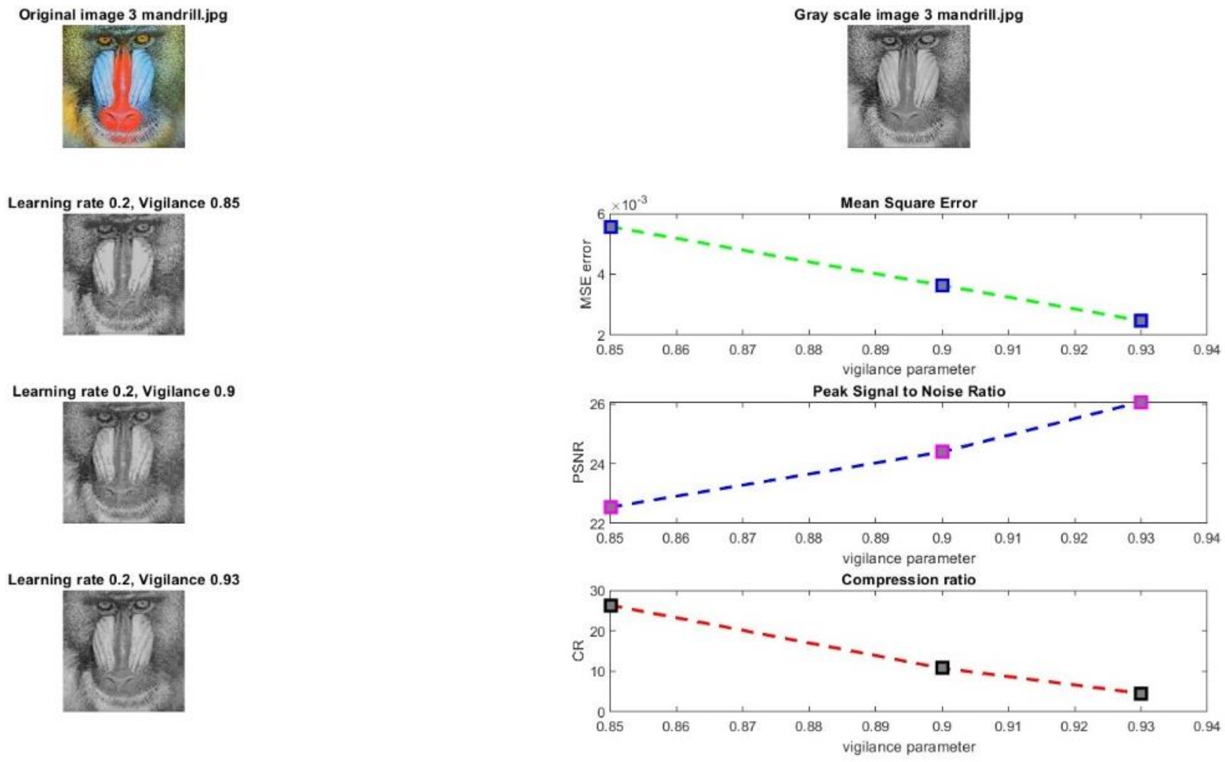

*Figure 3: Mandrill Image Compression and Metrics*

## Conclusion/Comments

The goal of this assignment was met, and an ART based image compression algorithm was developed. Some points may be discussed below:

- The final images are not identical to the originals. In fact, they are less clear. This was to be expected.
- In all the results, the MSE is inversely proportional to the vigilance parameter. The PSNR grows as the vigilance parameter increases. The CR decreases as the vigilance parameter rises.
- Less error occur and less information will be lost the greater the vigilance parameter is. This is logical since there will be more categories. As well, this explains why CR is inversely proportional to the vigilance parameter.
- It would take a lot of time to stabilize the system and let it converge. This step may be skipped here since the algorithm is not used in a supervised way and we are not in need of controlling the number of categories.
- Instead of taking the first pattern from each group to create the code book, we could try taking the mean of all patterns in a category to even generalize more the outcome.
- It's possible to notice that many patterns in the beginning belong to the first category. This is true for all the images examined. As a result, the classification may be biased, and there may be a way to address this issue.
- The algorithm works well with any image type, size, rgb or gray scale. However, the larger the image, the longer it will take to be compressed.
- The results could be seen better in MATLAB through zooming in and out.