NML502 HW #3

Elliot Smith, Eugen Hruska, Warum Suriyanarayana

1/31/2018

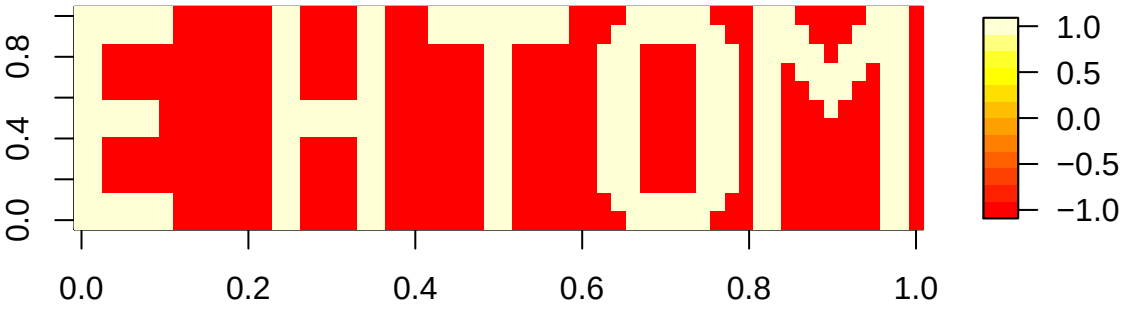
Problem 3

Problem 3a

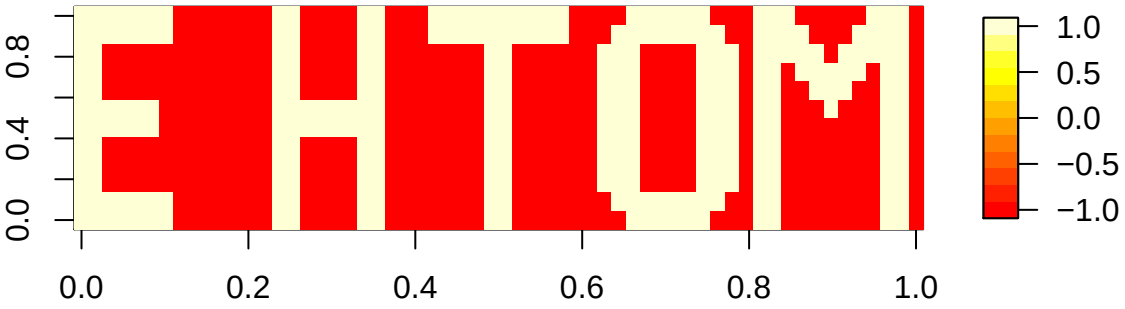
Recall Accuracy

- Learning Rate: 0.001
- Max Number of Learning Steps: 1000
- Error Threshold: 0.0000001
- Error Recall Measure: Percent of thresholded image pixels properly recalled

Original Input and Desired Output



Memory Recall



Summary

Please see above for our parameter selections. Neither our data nor memory matrix in this iteration were corrupted in any way. Our maximum number of learning steps is 1000, however, if we reached our error tolerence of 0.0000001 then we would immediately terminate the learning process and return the memory matrix. Our learning rate acts as a factor by which to influence the change of weights, we felt that 0.001 was

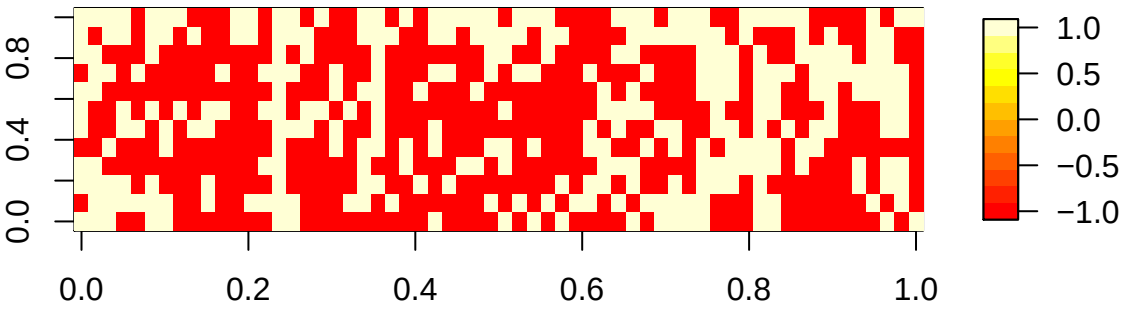
appropriate for us. There was no provided, thresholded, recalled image in this case because our recall was perfect. In this case, our recall error was a perfect 100%!

Problem 3b

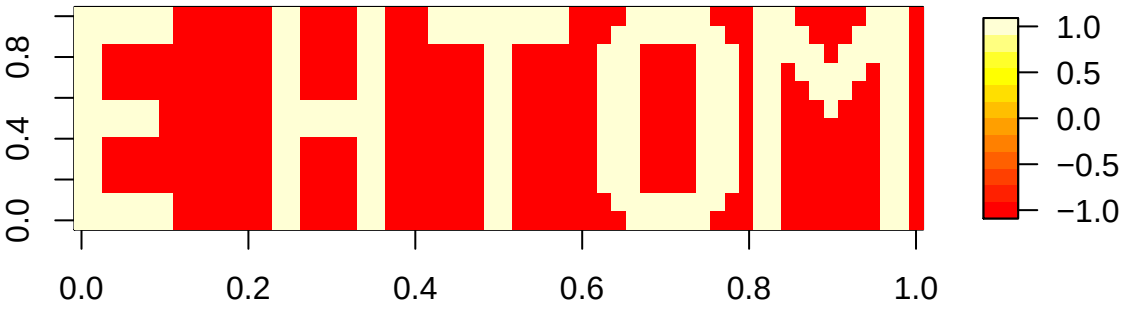
25% Input Corruption Recall Accuracy

- Learning Rate: 0.001
- Max Number of Learning Steps: 1000
- Error Threshold: 0.0000001
- Error Measure: Percent of thresholded image pixels properly recalled

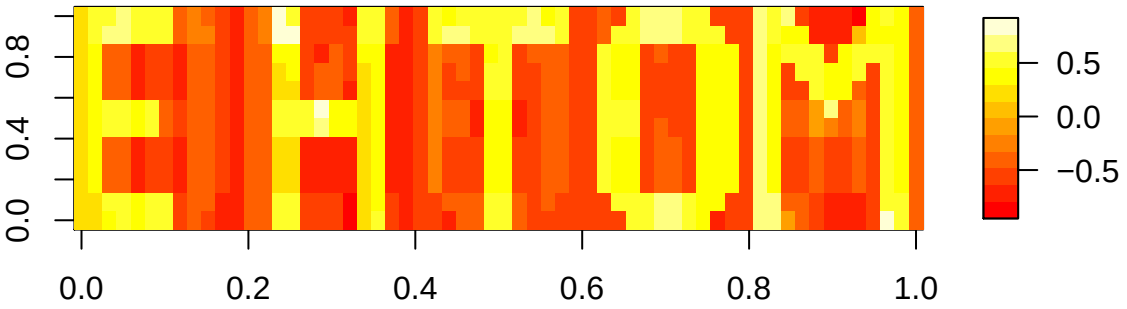
Original Input



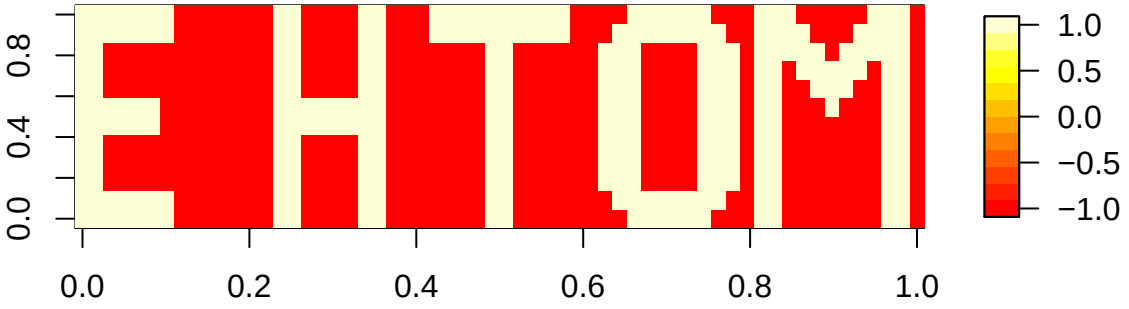
Desired Output



Memory Recall



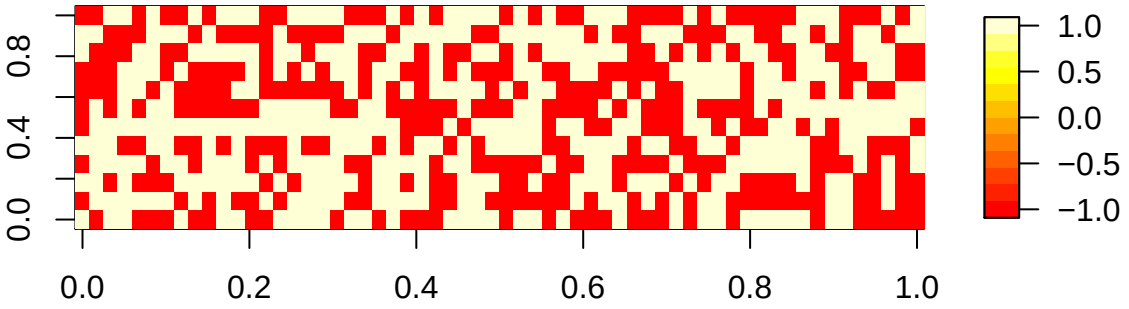
Thresholded Memory Recall



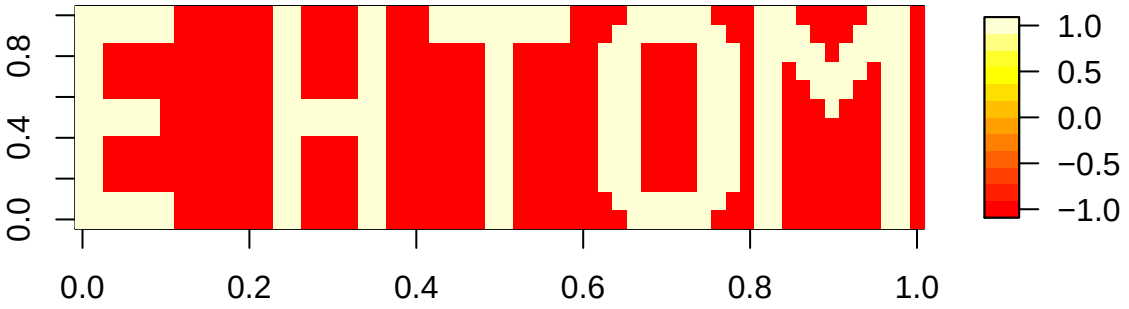
50% Input Corruption Recall Accuracy

- Learning Rate: 0.001
- Max Number of Learning Steps: 1000
- Error Threshold: 0.0000001
- Error Measure: Percent of thresholded image pixels properly recalled

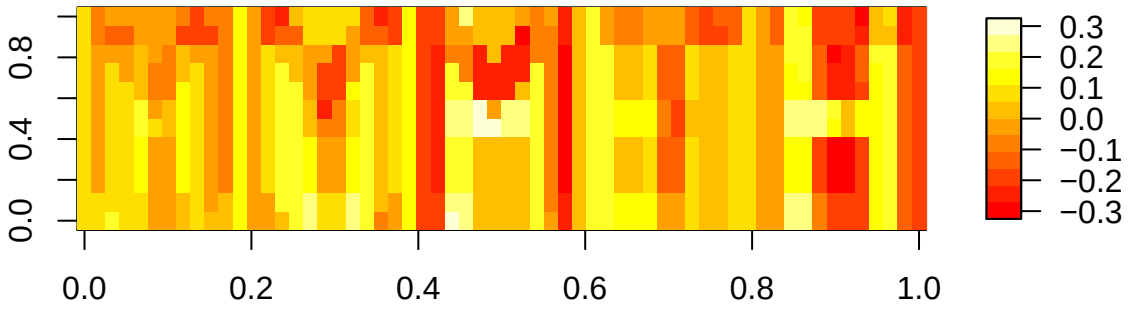
Original Input



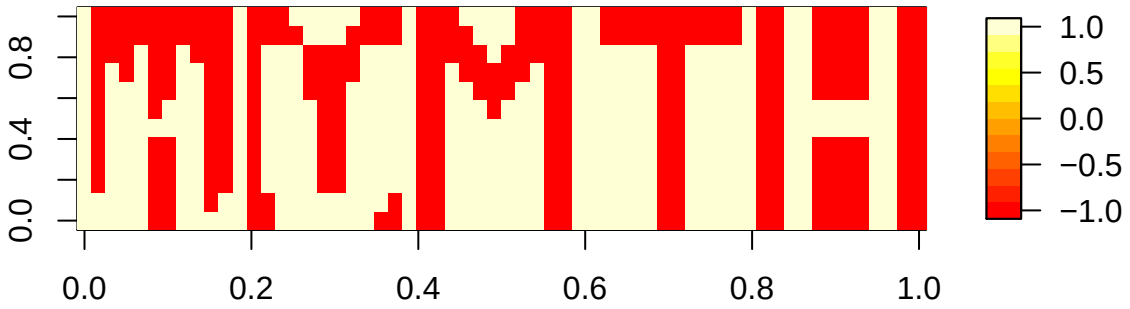
Desired Output



Memory Recall



Thresholded Memory Recall



Summary

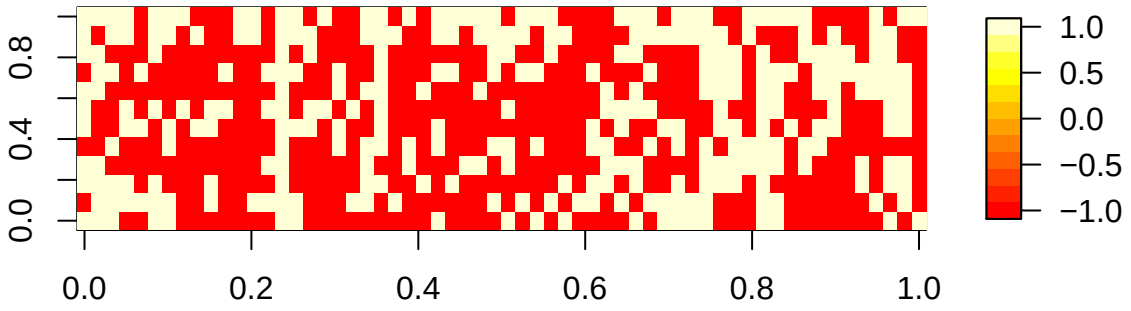
Please see above for our parameter selections. In our first iteration in this section, we randomly select 25% of the input values for each pattern and corrupted them by switching their signs; in the second iteration we increased that number to 50%. Our maximum number of learning steps is 1000, however, if we reached our error tolerence of 0.0000001 then we would immediately terminate the learning process and return the memory matrix. Our learning rate acts as a factor by which to influence the change of weights, we felt that 0.001 was appropriate for us. Our threshold methodology was to set the value to 1 if a value was greater or equal to 0 and to change it to -1 in all other cases. As can be seen by our recalled image and recalled thresholded image of the 25% corruption, we had great results overall and the image can still be interpreted very clearly. However, our 50% corruption did not have great results trying to reproduce the desired output, we can see some what appears to be some of our letters, but it is quite vague. For our 25% corrupted inputs our error recall was again a perfect 100%! However, for our 50% corrupted input, our error recall was only 49.72%, much lower than for the 25% corrupted inputs.

Problem 3c

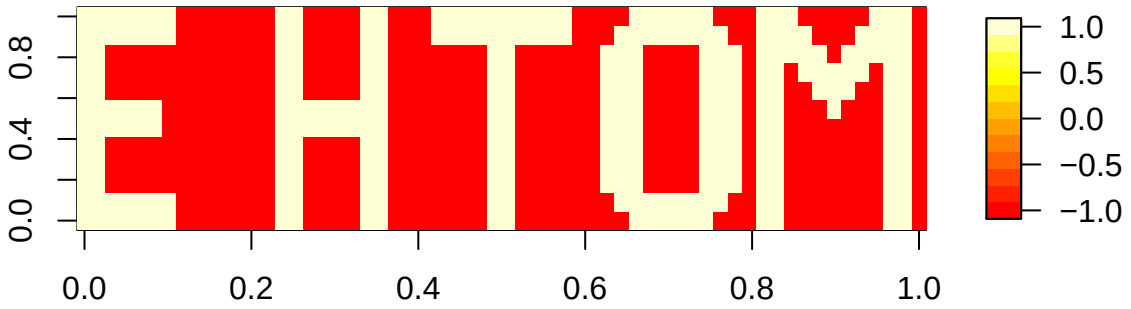
Corrupted Memory Recall Accuracy

- Learning Rate: 0.001
- Max Number of Learning Steps: 1000
- Error Threshold: 0.0000001
- Error Measure: Percent of thresholded image pixels properly recalled

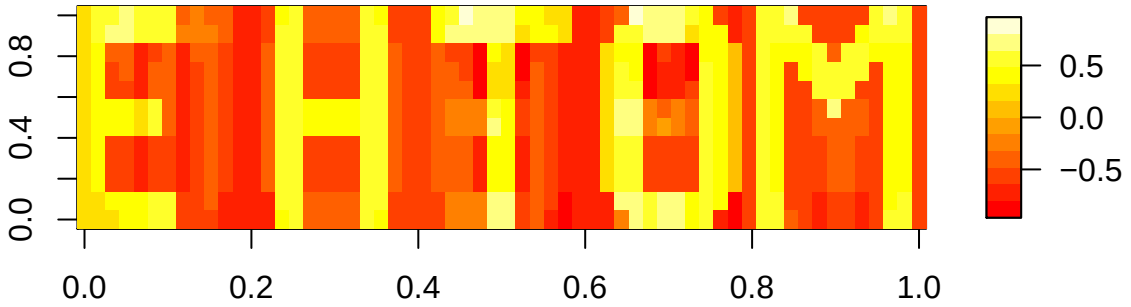
Original Training Input



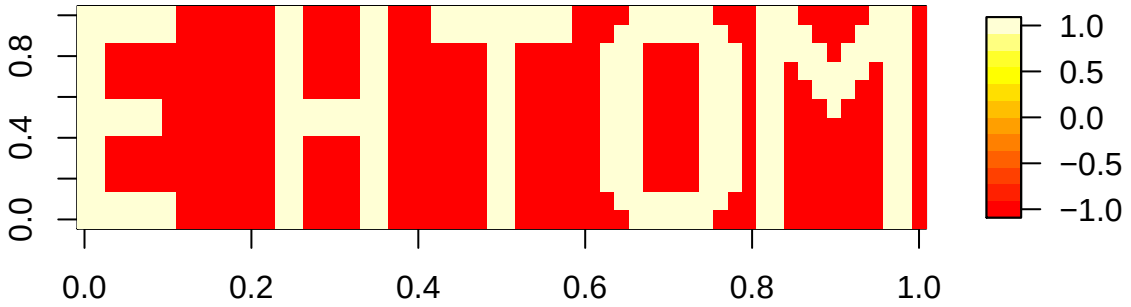
Desired Output



Memory Recall



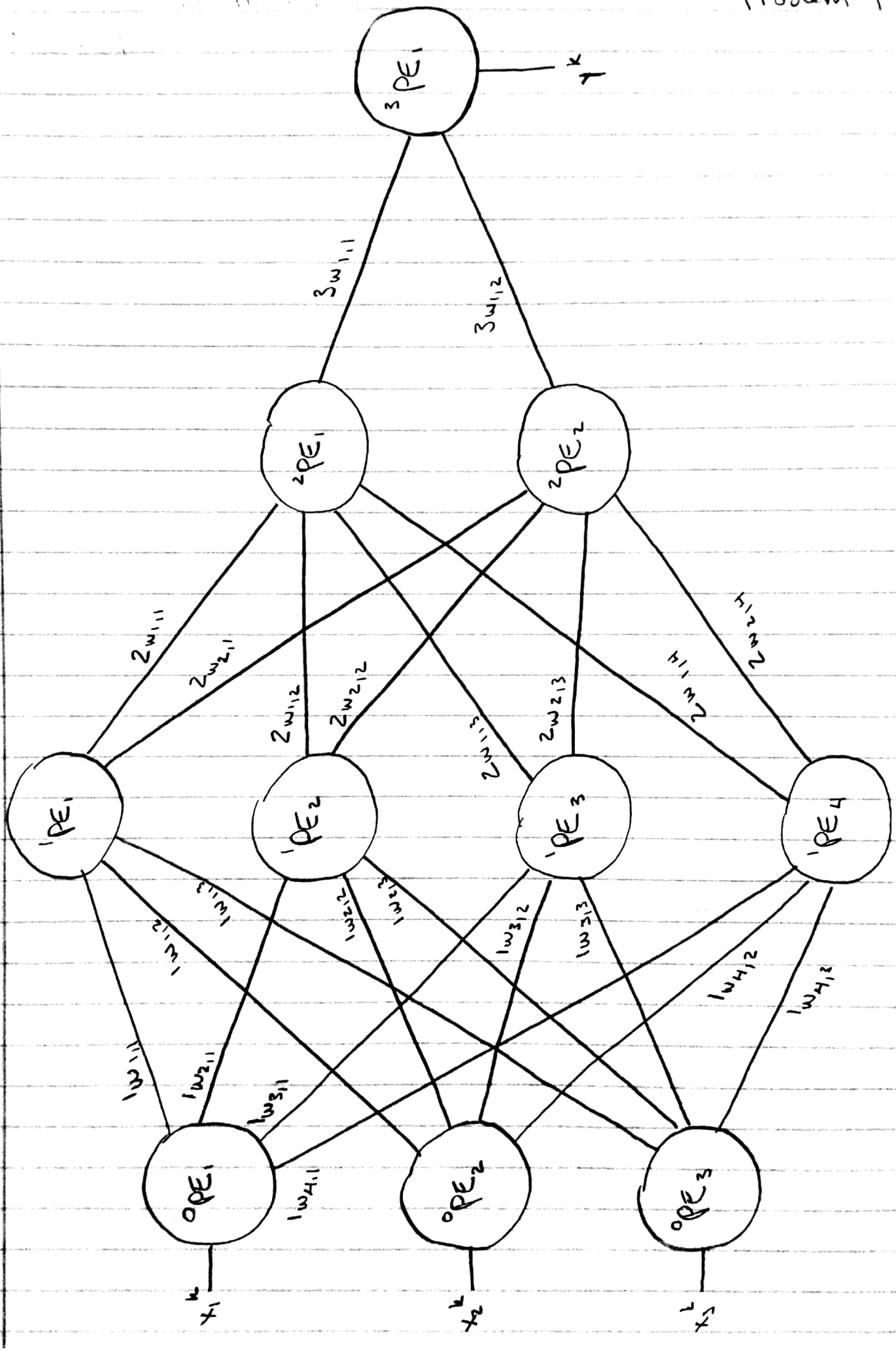
Thresholded Memory Recall



Summary

Please see above for our parameter selections. In this section, instead of corrupting the inputs to our network, we corrupted the inputs used to train our network; we corrupted 25% of the values from each input pattern. We omit our image of the input as it is the same from Problem 3a. Our maximum number of learning steps is 1000, however, if we reached our error tolerence of 0.0000001 then we would immediately terminate the learning process and return the memory matrix. Our learning rate acts as a factor by which to influence the change of weights, we felt that 0.001 was appropriate for us. Our threshold methodology was to set the value to 1 if a value was greater or equal to 0 and to change it to -1 in all other cases. As can be seen by our recalled image and recalled thresholded image in this case, we had quite good results. In the original recalled image, we can clearly see the EHTOM pattern what we expect, and at the same time, our thresholded image looks to be perfectly recalled. In this case, our recall error was a perfect 100%!

Problem 1



Problem 2

a. We will illustrate our solution to this problem on a three-layered perceptron, though these principles apply broadly to multi-layered perceptrons.

$$y_m^1 = \sum_{i=1}^{n} {}^1w_{i,m} x_i$$

$$y_m^2 = \sum_{i=1}^{n} {}^2w_{i,m} \left[\sum_{j=1}^{n} {}^1w_{j,i} x_j\right] = \sum_{j=1}^{n}\sum_{i=1}^{n} {}^2w_{i,m} \, {}^1w_{j,i} x_j$$

$$y_m^3 = \sum_{i=1}^{n} {}^3w_{i,m} \left[\sum_{j=1}^{n} {}^2w_{j,i} \left[\sum_{k=1}^{n} {}^1w_{k,j} x_k\right]\right] = \sum_{k=1}^{n}\sum_{j=1}^{n}\sum_{i=1}^{n} {}^3w_{i,m} \, {}^2w_{j,i} \, {}^1w_{k,j} x_k$$

$$\begin{bmatrix} y_1^3 \\ \vdots \\ y_n^3 \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{n}\sum_{i=1}^{n} {}^1w_{i,1} \, {}^2w_{j,i} \, {}^3w_{1,j} & \cdots & \sum_{j=1}^{n}\sum_{i=1}^{n} {}^1w_{i,1} \, {}^2w_{j,i} \, {}^3w_{n,j} \\ \vdots & \ddots & \vdots \\ \sum_{j=1}^{n}\sum_{i=1}^{n} {}^1w_{i,n} \, {}^2w_{j,i} \, {}^3w_{1,j} & \cdots & \sum_{j=1}^{n}\sum_{i=1}^{n} {}^1w_{i,n} \, {}^2w_{j,i} \, {}^3w_{n,j} \end{bmatrix} \begin{bmatrix} x_1^3 \\ \vdots \\ x_n^3 \end{bmatrix}$$

b. The ability of a neural network to learn and recall a certain pattern is not just dependent on its structural complexity, but also on the properties of the transfer function. As illustrated in Problem 2a, increasing the complexity of the ANN by adding more layers has no impact on a linear transfer function's ability to learn and recall patterns any differently than it it were a single-layer perceptron.