EECE6036 - Homework 5

Wayne Stegner

December 1, 2020

1 Problem 1

1.1 Problem Summary

The goal of this problem is to create a self-organized feature map (SOFM). The SOFM will have a 12 X 12 neuron output map, and it will learn the features of the MNIST dataset.

1.2 Results

1.2.1 System Description

Table 1 shows the hyper-parameters used in training the SOFM. These hyper-parameters were found empirically with considerations for maximizing the classification accuracy for the second problem of this assignment.

Table 1: Autoencoder Training Hyper-Parameters

Parameter	Value	Description
$epochs_{phase\_1}$	1000	Epochs for phase 1
$epochs_{phase\_2}$	4000	Epochs for phase 2
η_0	0.1	Initial learning rate
τ_L	434.29	Learning decay time constant
η_{floor}	0.01	Learning rate floor for phase 2
σ_0	6.0	Initial neighborhood parameter
τ_N	244.24	Neighborhood decay time constant
σ_{floor}	0.1	Neighborhood floor for phase 2

Weight initialization is done on a uniform distribution on the interval $(0, 1)$.

This SOFM trains using two-phase learning, as described in the slides. In the first phase, the learning rate, $\eta(t)$ starts at η_0, and it decays to η_{floor} by:

$$\eta(t) = \eta_0 * exp(\tfrac{-t}{\tau_L})$$

$$\tau_L = \tfrac{-epochs_{phase\_1}}{ln(\eta_{floor}/\eta_0)}$$

Similarly, the neighborhood parameter, $\sigma(t)$ starts at σ_0, and it decays to σ_{floor} by:

$$\sigma(t) = \sigma_0 * exp(\tfrac{-t}{\tau_N})$$

$$\tau_N = \tfrac{-epochs_{phase\_1}}{ln(\sigma_{floor}/\sigma_0)}$$

During the second phase, the learning rate and neighborhood parameter are held at η_{floor} and σ_{floor} respectively.

1.2.2 Heat Maps

Figure 1 shows activation heat maps for the test set after training. These maps were found by measuring the number of times a given neuron won for each class.

1.2.3 Features

Figure 2 shows the feature map learned by the SOFM after training. Each feature is a representation of the weights for that neuron.

1.3 Discussion and Analysis of Results

Overall, the results look pretty good. The class heat maps in Figure 1 show that the classes each have their own distinct regions of activity, with a few exceptions. These regions of activity correlate with the class appearing on the feature map. For example, the heat map for class 0 is active in the bottom right corner, and the feature map in Figure 2 looks like 0s in the bottom right corner. Classes 4 and 9 are very close in proximity, with class 4 being in the middle of the class 9 region. This is expected because 4 and 9 look very similar, so their regions on the feature map should be spatially close together. The feature map in Figure 2 supports this observation. In the upper right portion of the feature map, there is a mix of 4s and 9s in a similar arrangement to the heat maps for those classes. Also in Figure 1, the heat map for class 5 shows three major regions of activity. The feature map in Figure 2 has 5s appearing in those regions.

Classes which are similar in shape are spatially close to each other in the feature map. For example, the bottom left corner of the feature map in Figure 2 has a combination of 3s and 5s because they are similar in shape. Both 3 and 5 are similar to 8, which is found directly above the 3s and 5s in the feature map. 6 and 0 are also similar, and 6 is found just above the 0 region in the feature map.

There are a few features in Figure 2 which do not clearly resemble any class. For example, the feature three blocks down and three from the left side looks like a smudge or a blur. That same feature exhibits activity in the heat maps in Figure 1 in classes 2, 3, 5, 6, 7, and 8. That feature may be an artifact of the hyper-parameters chosen for training, and more testing should be done to try to eliminate ambiguous features.

1.4 Conclusion

The SOFM is able to effectively learn features from the MNIST dataset and group them by similarity. Further optimizing the hyper-parameters of the network may help to improve the produced feature map further.

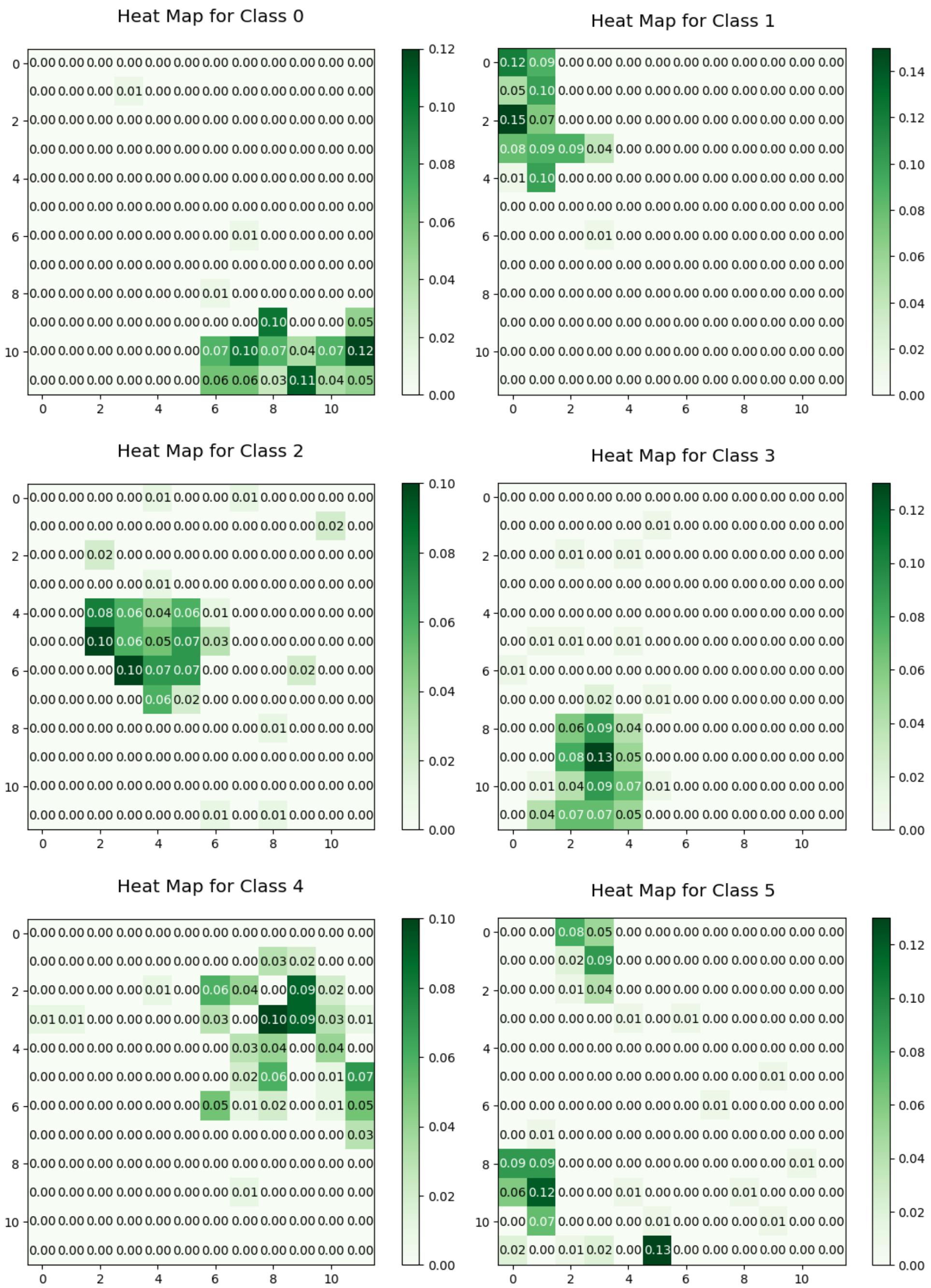

Figure 1: Class heat maps of the SOFM.

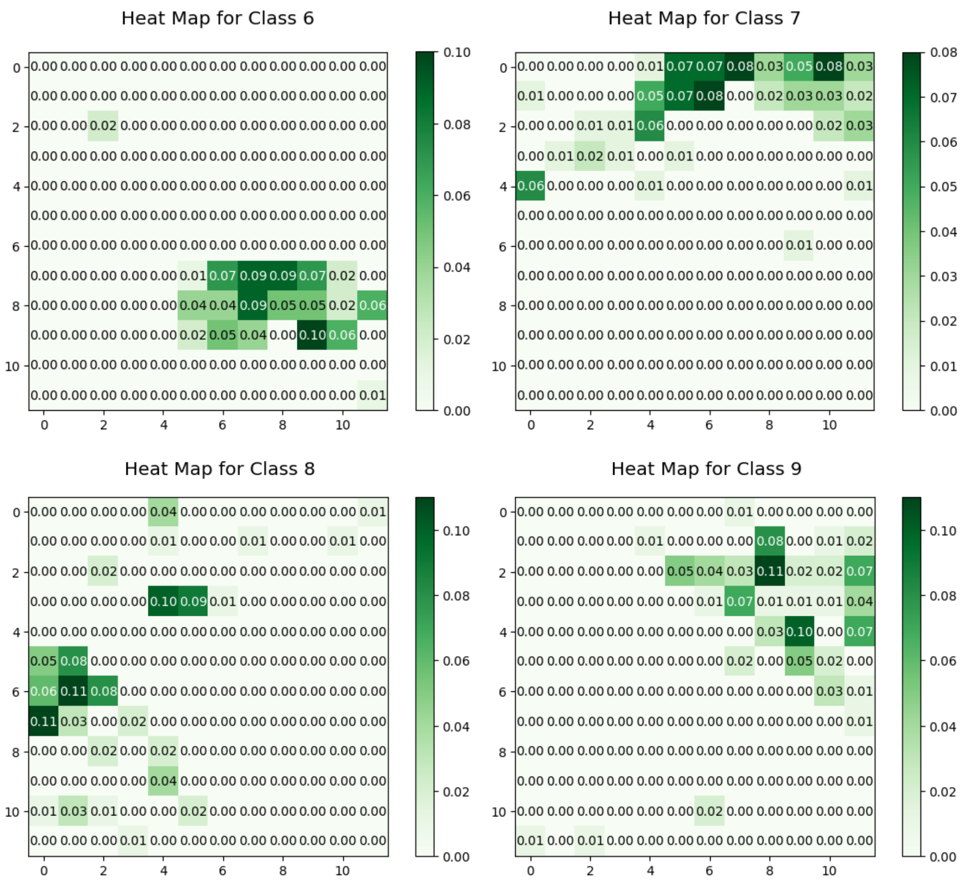

Figure 1: Class heat maps of the SOFM, continued.

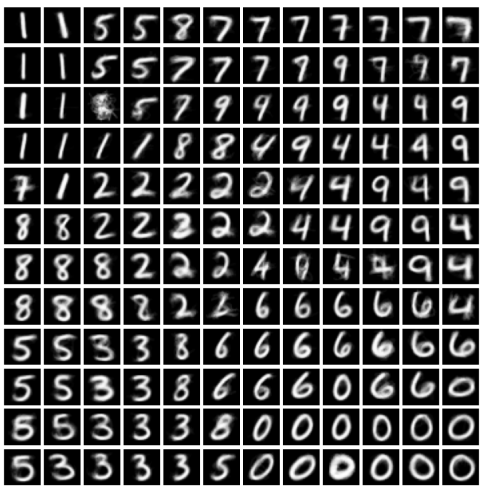

Figure 2: Feature map produced by the SOFM algorithm.

2 Problem 2

2.1 Problem Summary

The goal of this problem is to create a classifier using the features learned from the SOFM trained in Problem 1. This is done by passing the image through the SOFM using winner-take-all to generate a one-hot encoded feature vector to pass into the MLP classifier.

2.2 Results

2.2.1 System Description

Table 2 shows the hyper-parameters used in training the classifier. These hyper-parameters were found empirically.

Table 2: Classifier Training Hyper-Parameters

Parameter	Value	Description
η	0.05	Learning rate
α	0.8	Momentum
$epochs_{max}$	1000	Maximum training epochs
L	0.25	Lower activation threshold
H	0.75	Upper activation threshold
$patience$	5	Patience before early stopping

This network uses the same training algorithm as the previous problems, including momentum, thresholding, early stopping, and weight decay.

2.2.2 Network Results

Throughout the duration of training, the loss of the training and validation sets was tracked every 5 epochs, shown in Figure 3. The vertical lines designate the point where the validation error is minimized. The classifier achieves a minimal test loss of 0.134 at epoch 10.

Figure 4 show the confusion matrices for the train and test sets of the classifier.

2.3 Discussion and Analysis of Results

Overall, the classifier performs pretty well. The network only took 50 epochs to train, and was able to train significantly faster than the previous classifiers. The confusion matrices in Figure 4 show a dark diagonal line but with some visible confusion, especially between classes 4 and 9. The significant points of confusion are between classes which are adjecent, such as 4/9 and 3/8.

The classification performance was not as good as the classifiers from Homework 3 (loss = 0.070) and Homework 4 (losses of 0.103 and 0.121). It makes

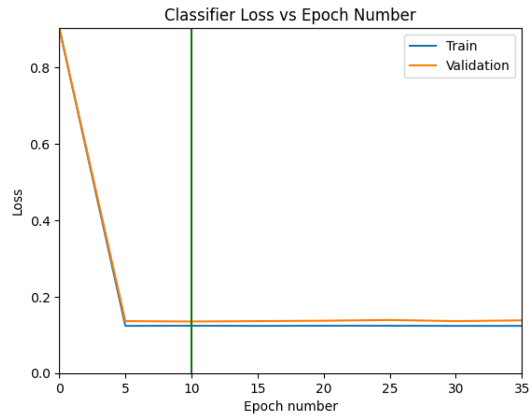

Figure 3: Training and validation loss of the classifier.

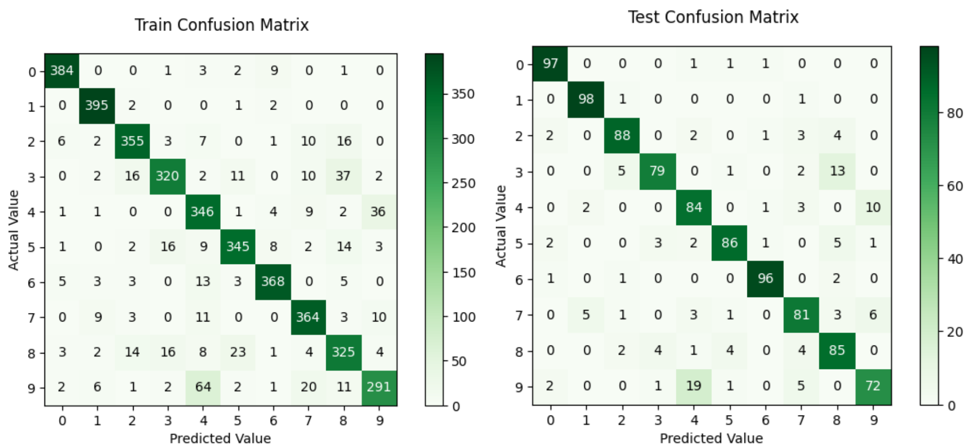

Figure 4: Confusion matrices of the classifier.

sense that this classifier is not as performant, because the one-hot vectors from the SOFM do not carry as much information as the analog layers from the MLPs.

2.4 Conclusion

Using the features learned by the SOFM, the classifier is able to effectively classify the MNIST dataset. While its performance is not as good as the previously studied classifiers, it effectively demonstrates how the SOFM features are viable for distinguishing between different digits.