Question 2

Introduction

I programmatically implemented a Kohonen network that supported 2 to 4 units, each of which was 3-dimensional. I then tested the network with input for which the output was predetermined. Finally, I used the network to cluster a set of data into 2 to 4 clusters.

Program

Description

The entry point to the program is the main() function in the __main__ module. The interaction between the user and the program is command-line-style. The program consists of 3 modules:

- **__main__**. The entry point into the program, responsible for getting user input and passing data points to the ‘kohonen’ module.
- **kohonen**. An implementation of the Kohonen layer of a Kohonen-Grossberg counter-propagation network, responsible for training it and clustering the data points received from the main module. Equation 1 and Equation 2 give the formulae for net and activation (Vella & Vella, 2009).
- **vector**. A class representing an n-dimensional vector, responsible for computing its length, and normalising and performing arithmetic on it.

Equation 1: A unit u's net net_u in terms of each input x and its weight w, out of n weighted inputs.

$$net_u = \sum_{i=1}^{n} w_i x_i$$

Equation 2: A Kohonen unit u's activation a_u in terms of the unit's net net_u and each other unit v's net net_v.

$$a_u = \begin{cases} 1, & \forall\, v \neq u, net_u > net_v \\ 0, & otherwise \end{cases}$$

Development

I developed the program in Python using the platform's IDLE (Interactive Development Learning Environment).

Testing

I tested the program using the Python Shell, in which I input the source of the data to cluster, the number of units to use, and their coordinates. I checked that the data points were normalised correctly by having the program print out the lengths of the normalised data points and checking that the lengths were effectively 1.

The test data were the inputs to an AND gate perceptron and the choice of units were the associated weights for the perceptron, both normalised. I used these as input to the program and compared the output with their working to verify my network's clustering.

Results

Table 1 shows the results for the AND gate data.

Table 1: Results for example data.

Cluster: (-0.58, -0.58, 0.58)	**Cluster: (0.41, 0.41, -0.82)**
(0.00, 0.00, 1.00)	(0.5774, 0.5774, 0.5774)
(0.00, 0.71, 0.71)	
(0.71, 0.00, 0.71)	

Table 2 to Table 4 show the results for the data in Table 1 (specification) with different numbers of clusters.

Table 2: Results for specification data (2 clusters).

Cluster: (0.15, 0.03, 0.99)	**Cluster: (0.73, 0.00, 0.69)**
(-0.82, 0.49, 0.29)	(0.28, 0.96, 0.01)
(-0.77, 0.47, 0.43)	(0.31, 0.95, 0.07)
(-0.73, 0.55, 0.41)	(0.36, 0.91, -0.22)
(-0.71, 0.61, 0.36)	(0.37, 0.93, 0.09)
(-0.69, 0.59, 0.42)	(0.43, 0.83, -0.34)
(-0.68, 0.60, 0.42)	(0.43, 0.90, 0.01)
(-0.66, 0.58, 0.48)	(0.47, -0.10, 0.88)
(-0.61, 0.69, 0.40)	(0.47, -0.07, 0.88)
(0.35, -0.06, 0.94)	(0.48, 0.82, -0.30)
	(0.53, -0.23, 0.82)
	(0.54, -0.21, 0.82)
	(0.58, -0.38, 0.72)
	(0.58, 0.81, 0.00)
	(0.65, -0.04, 0.76)
	(0.74, -0.05, 0.67)

Table 3: Results for specification data (3 clusters).

Cluster: (0.51, -0.02, 0.86)	**Cluster: (0.73, 0.00, 0.69)**	**Cluster: (0.70, -0.22, 0.68)**
(-0.82, 0.49, 0.29)	(0.28, 0.96, 0.01)	(0.53, -0.23, 0.82)
(-0.77, 0.47, 0.43)	(0.31, 0.95, 0.07)	(0.58, -0.38, 0.72)
(-0.73, 0.55, 0.41)	(0.36, 0.91, -0.22)	
(-0.71, 0.61, 0.36)	(0.37, 0.93, 0.09)	
(-0.69, 0.59, 0.42)	(0.43, 0.83, -0.34)	
(-0.68, 0.60, 0.42)	(0.43, 0.90, 0.01)	
(-0.66, 0.58, 0.48)	(0.48, 0.82, -0.30)	
(-0.61, 0.69, 0.40)	(0.58, 0.81, 0.00)	
(0.35, -0.06, 0.94)	(0.65, -0.04, 0.76)	
(0.47, -0.10, 0.88)	(0.74, -0.05, 0.67)	
(0.47, -0.07, 0.88)		
(0.54, -0.21, 0.82)		

Table 4: Results for specification data (4 clusters).

Cluster: (0.71, 0.04, 0.70)	**Cluster: (0.68, -0.04, 0.73)**	**Cluster: (0.72, -0.14, 0.68)**	**Cluster: (0.66, 0.15, 0.73)**
(-0.82, 0.49, 0.29)	(-0.77, 0.47, 0.43)	(0.53, -0.23, 0.82)	(0.74, -0.05, 0.67)
(-0.73, 0.55, 0.41)	(0.35, -0.06, 0.94)	(0.58, -0.38, 0.72)	
(-0.71, 0.61, 0.36)	(0.47, -0.10, 0.88)		
(-0.69, 0.59, 0.42)	(0.47, -0.07, 0.88)		
(-0.68, 0.60, 0.42)	(0.54, -0.21, 0.82)		
(-0.66, 0.58, 0.48)	(0.65, -0.04, 0.76)		
(-0.61, 0.69, 0.40)			
(0.28, 0.96, 0.10)			
(0.31, 0.95, 0.07)			
(0.36, 0.91, -0.22)			
(0.37, 0.93, 0.09)			
(0.43, 0.83, -0.34)			
(0.43, 0.90, 0.01)			
(0.48, 0.82, -0.30)			
(0.58, 0.81, 0.00)			

Analysis

Figure 1 to Figure 4 show that over time, the total weight change per iteration and epoch sometimes increases but mostly decreases. The more clusters there are, the higher the frequency and lower the amplitude of these oscillations.

Figure 1: Sum of weight changes by iteration for specification data (2 clusters).

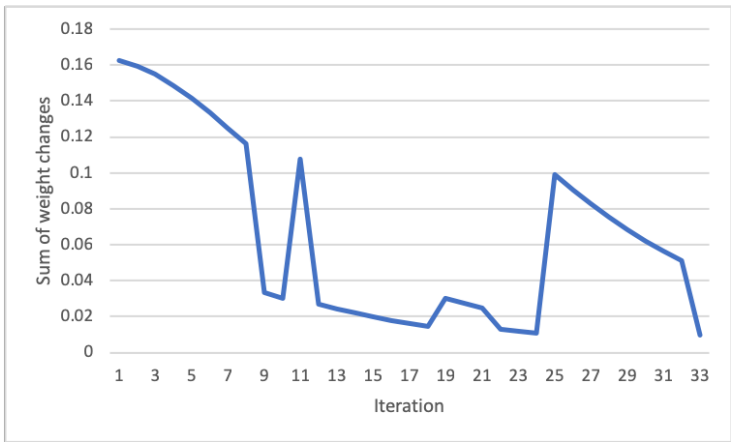

Figure 2: Sum of weight changes by iteration for specification data (3 clusters).

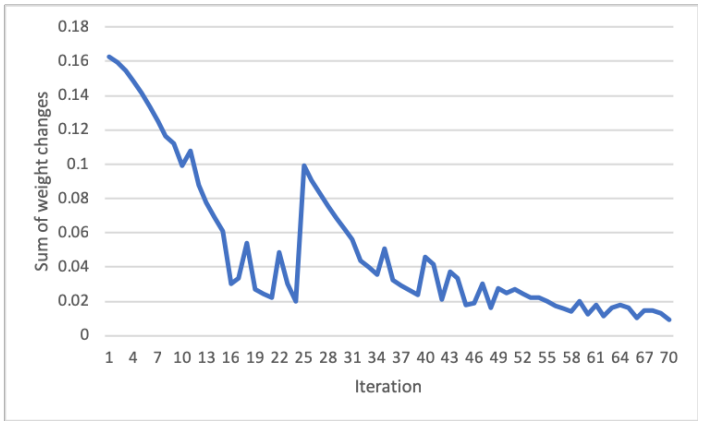

Figure 3: Sum of weight changes by iteration for specification data (4 clusters).

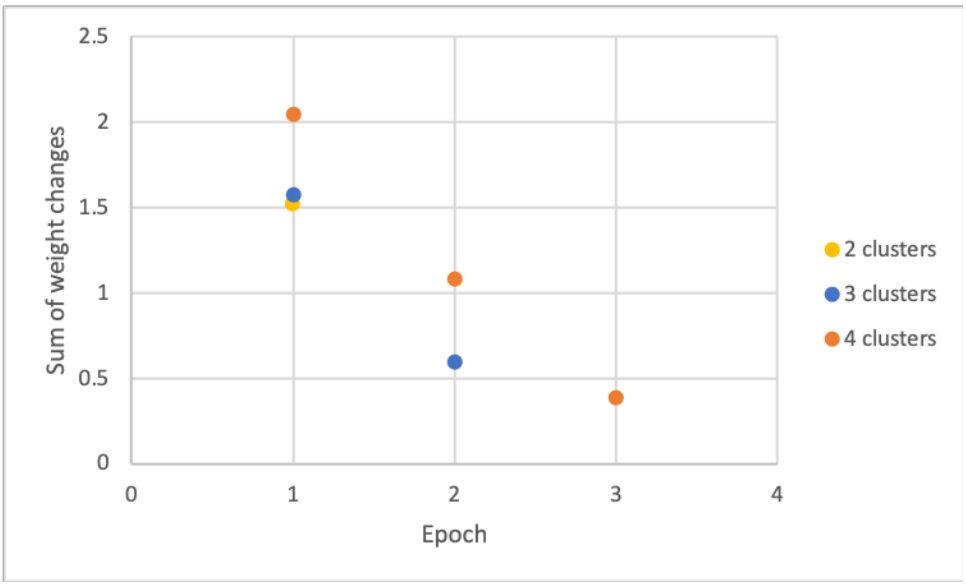

Figure 4: Sum of weight changes by epoch for specification data.

Equation 3 gives the formula for weight update (Vella & Vella, 2009).

Equation 3: Change in selected weight class w_c in terms of itself, a learning rate η, and a training example x.

$$\eta(x - w_c)$$

Observations

The lengths of the normalised data points did not always appear to be 1 but sometimes a decimal point followed by a recurring 9. Also, clustering modified the original data points by a small amount. Finally, the more clusters there were, the smaller the distances between them.

Conclusion

Kohonen networks are useful for clustering data and extracting data from clusters. However, the accuracy of computations may impact the effectiveness of the Kohonen layer's training.

References

BBC, 2014. *UK to allow driverless cars on public roads in January.* [Online]

Available at: https://www.bbc.co.uk/news/technology-28551069

[Accessed 13 February 2019].

Carnegie Mellon University, n.d. *Navlab: The Carnegie Mellon University Navigation Laboratory.* [Online]

Available at: http://www.cs.cmu.edu/afs/cs/project/alv/www/index.html

[Accessed 6 February 2019].

Doude, M., Goodin, C. & Carruth, D., 2018. *Driving autonomous cars off the beaten path.* [Online]

Available at: https://phys.org/news/2018-11-autonomous-cars-beaten-path.html

[Accessed 6 February 2019].

Eindhoven University of Technology, 2018. *New AI method increases the power of artificial neural networks.* [Online]

Available at: https://phys.org/news/2018-06-ai-method-power-artificial-neural.html

[Accessed 13 February 2019].

Flinders University, 2018. *Super-fast flying machines defy body logic.* [Online]

Available at: https://phys.org/news/2018-11-super-fast-machines-defy-body-logic.html

[Accessed 6 February 2019].

Vella, A. & Vella, C., 2009. *Neural networks.* London: University of London.

Martineau, K., 2018. *Aleksander Madry on building trustworthy artificial intelligence.* [Online]

Available at: https://phys.org/news/2018-12-aleksander-madry-trustworthy-artificial-