UNSUPERVISED LEARNING

Assignment guidelines:

The exercises should be done in Matlab. You must provide a written report, as a Microsoft Word or PDF file that answers each section. Each section (e.g. 1A, 1B, etc) is worth 1 point, for a total of 10 maximum points for the entire assignment. The text of the report should not be more than eight pages, ideally much less, including figures. Font size should be 12, and margins should be 1”. You should illustrate key results such as graphs or plots with figures, which should each have a short figure legend. Describe in the text what you found, and discuss what it means. It is <u>not</u> sufficient to simply show Matlab figures, with little or no explanatory text — you must show me that you understand.

Please provide any Matlab code <u>that you make and that you think is relevant</u> in a form that I can run. Do not send me code that I provided; I already have it. The Matlab code you made must be provided in addition to the above written report. User-friendliness and comments in the M-files are very welcome.

All of this should be packaged in a single archive file, ideally ZIP. <u>Please include your last name, as part of the file name, e.g. Smith_603_assign.zip.</u>

Part 1 — Machine Learning: k-Means Clustering (3 points)

Run the k-Means clustering code using

kMeans;

This code makes three two-dimensional distributions and clusters them with k=3.

A. Make two distributions but cluster with k=3

Investigate how the k-Means algorithm behaves when k is erroneously set too high. Report and describe.

B. Make three distributions but cluster with k=2

Investigate how the k-Means algorithm behaves when k is erroneously set too low. Report and describe.

C. Make three distributions, let your code determine k

Use e.g. the silhouette measure to determine the ideal k, trying k = [2, 3, 4, 5]. For a starting point, see:

http://www.mathworks.com/help/stats/k-means-clustering.html

Your may want to use the ‘replicates’ option for this section.

Part 2 —Learning in the Brain: STDP (3 points)

Simulate Fig 4 in Song et al 2000 and Fig 1 in Song et al 2001 using

Song2000_F4;

Song2001_F1;

respectively.

A. Song et al say STDP reduces latencies and sharpens responses

Our **Song2000_F4** simulation code shows that STDP reduces latencies but it does not actually sharpen the response. This is because the starting setting of the synaptic conductance g is too low, so synapses are initially too weak to properly drive the postsynaptic neuron. Find this setting for initial g in **simSTDPlatencies**, double it, and explain what is different.

B. STDP provides a degree of stability

Keep increasing the starting value of the synaptic conductance g to something unreasonably large, as indicated by the postsynaptic neuron saturating (reaching "depolarization block"). Run the **Song2000_F4** simulation and compare the before/after activation pattern. Show the resulting graphs. Explain how STDP can provide a degree of stability to a neuron.

C. Without instruction, STDP can pick up on correlations in inputs

The simulation code **Song2001_F1** recapitulates the findings in Fig 1 of Song et al 2001. Run the code. Explain how this is unsupervised learning and classification. Note that the outcome in the bottom right panel is random and so you may get quite different results if you run it a few times.

Part 3 — An Associative Memory: the Hopfield Network (4 points)

Run the network using e.g.

hopfield_net(100,'mem_ABC.txt',10,1);

This sets the network size to **100**. The code will pick the memories from the specified text file **mem_ABC.txt**, from which it will create the attractor states. The number **10** specifies the amount of noise it will use to corrupt the memory states with. The number **1** at the end is a Boolean flag that tells the code to rescale the image bit values in the text file from [0,255] to [-1,+1], so that you can import images from e.g. ImageJ, or other image processing software packages, which do not normally operate with negative pixel values.

A. Experiment with noise levels to find local minima and ghost states

Alter the noise levels until the network no longer correctly recalls the stored memories. What does the network converge to instead? Try several times with different noise levels until you obtain other attractor states, document these states, and explain why they happen, and what they are.

B. Alter one memory

Alter the text file so that e.g. the letter 'C' is no longer one of the attractor states, but store some other state, e.g. another character. Next try altering this memory so that is similar to one of the other two stored attractor states. What happens? Explain why.

C. The information storage limit of a Hopfield network

Add new well-separated memories to the Hopfield network, so that you have nine in total. Analytically calculate the theoretical storage limit of the network.

D. Calculate numerically the network energy as it converges

Use the Lyapunov function to show numerically how the network is attracted to a lower energy state as it converges. Explain how the Hopfield network can be used as a classifier to cluster data into memorized categories.