1. Describe your algorithm. Is your algorithm batch or online?
   1. Algorithm is online because the weights are updated, and different initial weights are used for each input sample.
2. What is the difference between batch and online?
   1. online machine learning is a method of machine learning in which data becomes available in a sequential order and is used to update our best predictor for future data at each step, as opposed to batch learning techniques which generate the best predictor by learning on the entire training data set at once.
   2. The on-line and batch modes are slightly different, although both will perform well for parabolic performance surfaces. One major difference is that the batch algorithm keeps the system weights constant while computing the error associated with each sample in the input. Since the on-line version is constantly updating its weights, its error calculation (and thus gradient estimation) uses different weights for each input sample. This means that the two algorithms visit different sets of points during adaptation. However, they both converge to the same minimum.
   3. Online: Learning based on each pattern as it is observed.

Batch: Learning over groups of patters. Most algorithms are batch.

1. Why is it important to normalise data and weight vectors for competitive learning to work?
   1. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. For machine learning, every dataset does not require normalization. It is required only when features have different ranges.
   2. Its important because it accelerates optimization
2. What could happen if data and weight wasn’t normalised. Give a mathematical argument
   1. If data and weights weren’t normalised, the algorithm will take too long to find appropriate weights for each feature.
      1. <https://datascience.stackexchange.com/questions/31292/what-is-normalization-for>
   2. Mathematical argument?? Use an equation and explain the process, also mentioning why it wouldn’t be ideal without normalisation
      1. 
      2. Allows data to changed to a scale of 0 to 1. If data was not scaled, different features will have different ranges of numbers. For example, age range can be from 0-100, whereas salary range can be from 20,000-500,000. The difference in scaling will make the algorithm take longer and reduce the accuracy of the results.
      3. Not normalising the data also increases the number of dead cells in the prototypes.
3. Describe the techniques you used to optimise the network and what you achieved by using them
   1. Normalised input or initial weights.
   2. Noise addition on the weights.
   3. Decaying learning rate.
   4. Leaky learning: update the weights of the losers as well as winners but with a much smaller learning rate.
   5. Update the winners and neighbouring losers.
4. Provide a measurement of improvement (How many dead units you have on average)
5. Perform statistics over at least 10 different initial conditions
6. Describe your method of identifying dead units without the need of visualising the prototypes
7. Create a figure showing the average weight change as a function over time
8. When has your network sufficiently learned from the data?
9. Produce a figure of the prototypes and comment on what they represent
10. How many prototypes did your network find?
11. Calculate the correlation matrix of the prototypes
12. How can you use the correlation matrix to find similarities between the prototypes?