**Neuroscience 299: Computing with High-Dimensional Vectors**
**Assignment 10: Relations to neural networks**
**Due November 10, 1pm**

Reminder: Please do *either* the writing assignment *or* the programming assignment. Expected length for the writing assignment is approximately 250-500 words, but there is no strict minimum or maximum.

***Writing assignment:***

Pick either option A or option B (you need not and should not answer both).

*Option A: Hybrid VSA / Neural Network architectures.* At a first glance, HD computing/VSA may seem to operate from different principles compared to neural networks. Nevertheless, as we saw in the lecture there are some striking commonalities. Give an example of a scenario where combining HD computing/VSA and neural networks would be a good idea, and provide evidence as to why this approach should succeed. Examples of evidence can include: related implementations in papers, or complementary advantages of HD computing/VSA (neural networks) that the other framework can help with.

*Option B: Differentiable HRRs*. Read the paper "Learning with Holographic Reduced Representations" by Ganesan et al. (see extended reading list). Explain (to someone familiar with HRR but who has not read the paper) the paper's main contributions, and suggest one research direction in which to extend the results.

***Programming assignment:***

In this assignment, we will consider one concrete case of drawing similarity between HD computing/VSA and randomly connected neural networks called random vector functional link (RVFL) networks aka extreme learning machines as described in "Density Encoding Enables Resource-Efficient Randomly Connected Neural Networks". Your task will be to finish the implementation of the RVFL variant from the paper. The setup is very similar to Part 2 of the programming assignment for Module 9.

We will use the same dataset as in the previous assignment – "Pen-Based Recognition of Handwritten Digits"; it is from the UCI machine learning repository.
As a reminder, here is the link to ".mat" file with the dataset:
https://www.dropbox.com/s/jilmya8a9f8cywh/pendigits.mat?dl=0

You are also provided with a Jupyter notebook (see course website) that includes the code to support loading data and the general logic of the process. The provided Jupyter notebook is missing a few steps, which you will implement (search for ***#ToDo*** comments in the notebook). In particular, you will need to implement the following steps:

1. Implement the generation of thermometer codes (see lecture slides for Module 8) so that with the function "encodings" you can assign a code for a given feature level.

2. Form a compositional HD vector representing the set of key-value pairs, where the key is unique feature ID (random HD vector) and the value is the thermometer code of the feature's value.
3. Apply the clipping function with parameter kappa (see equation (5) in the paper) to values of HD vectors resulting from the previous step.
4. Complete the function "rls", which forms a regularized least squares (RLS) classifier.
5. Form a matrix with one-hot encodings of the ground truth labels of the training data.

Next, please perform the following experiments and answer the following questions:

1. Wrap the code around a "for loop" so that you can perform simulations for random initializations of "keyHVs". Compute the average accuracy over several random initializations for both the centroids-based classifier and the RLS classifier.
2. How do the accuracies of the centroids-based classifier and the RLS classifier compare to each other?
3. How does the choice of dimensionality of HD vectors affect the accuracy of both classifiers?
4. How does the choice of kappa used in forming compositional HD vectors affect the accuracy of both classifiers?
5. What is the role of "lambda" in the RLS classifier? How the choice of "lambda" affects the accuracy?