CS587 - Homework 1
Letter Recognition in a Hebb Network

Ronald Broberg
University of Colorado at Colorado Springs

***Abstract*—This paper examines character recognition in a limited training set in a single layer Hebb network. The training set consists of 3 examplars each of 7 different alphabetical characters from different fonts. The test set consists of the same set as the training set but with the addition of randomized noise.**

I. Introduction

This project is driven from examples in Fausett's *Fundamentals of Neural Networks* [1]. In particular, Example 2.8 "*A Hebb net to classify two-dimensional input patters (representing letters)*" and Example 2.14 "*A Perceptron to classify letters from differnt fonts: one output class*."

II. Data

Data was prepared by the instructor and made available for download from a class website. Two files were presented for the training and testing data respectively. The both files consisted of 21 letters arranged as 7 columns, 9 rows character arrays with '.' representing a null character and a '#' character representing a positive character. The 21 7x9 character arrays were presented sequentially by row with no breaks. The testing file was similar but included '@' and 'o' to represent noise effects that flipped null and positive bits within the chracter arrays presented in the training set.

III. Model

A single layer neural network with 63 nodes was created in the Julia language. Weights unique to each letter were tracked in a 7x63 array with one layer for each letter. Similarly, a 7 element bias vector was tracked with one element for each letter. The weights were randomly generated from a gaussian distribution and were scaled so that the initial weights were distrbuted around a mean of zero within a range of +/- 0.5 while the bias nodes were initialized with a value of one. Figure 1 depicts a schematic of the model. A Hebb training rule adjusted weights and biases after each input of a letter.

Each letter was trained separately. A single pass for each letter through the training set was sufficient for the network to distinguish characters within the training set. While the network was 100% accurate when "tested" against the original training letters, even with a single pass through the data set, the noisy test set triggered multiple false positive classifications when using a simple threshold for classifications. A post-neural-network filter was added which selected a single classification from the highest ("most confident") output values from the network. This resulted in two misclassifications out of 21 as shown in table 1 with the columns showing the prediction and the rows showing the known classification. When the network was retrained with two epochs, the result was still 2 misclassifications out of 21, but in this case it was two 'E's being misclassified as opposed to the two 'B's in the single epoch training model. Training the model with three epochs on the training data tuned the model sufficiently to perfectly classify the noisy test data.

R. Broberg is with the Department of Mathematics, University of Colorado at Colorado Springs, Colorado Springs, CO, 80909.

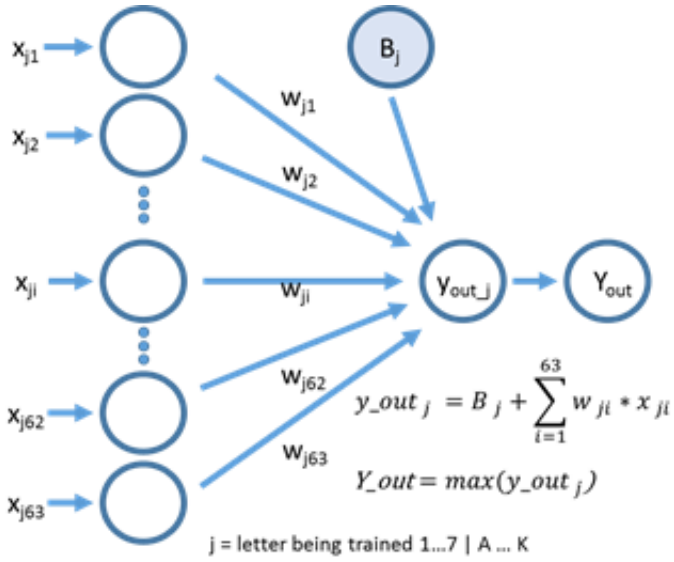

Figure 1: Single layer Hebb network with 63 nodes for 7 letters.

	A	B	C	D	E	J	K
A	3	-	-	-	-	-	-
B	-	1	-	1	1	-	-
C	-	-	3	-	-	-	-
D	-	-	-	3	-	-	-
E	-	-	-	-	3	-	-
J	-	-	-	-	-	3	-
K	-	-	-	-	-	-	3

Table I: Confusion table after one epoch.

IV. Results

A single layer neural network was created with 63 input nodes and a single output node. Seven sets of weights and biases were trained in parallel using a Hebb correction rule. A post-processing selection filter to select the ”most confident” classification was added to disambiguate multiple positive classifications of a single letter. This model perfectly classified the test data when trained on three epochs of the training data. The small sizes of both the training set and the test set warns against generalizing the results. Julia code is provided in the Supplemental Information.

References

[1] L. Fausett,*Fundamentals of Neural Networks: Architectures, Algorithms And Applications*, Pearson, 1993