MSDS 458 Deep Learning
Juan Campos
Assignment 1
7/22/2019

1. Executive Summary

This research was conducted to obtain practical experience and an understanding of designing, training, and evaluating a simple neural network. The study's main goal was to understand and interpret how the hidden layers in a simple neural network have learned to represent features within the input data to ultimately train neural networks that correctly generate desired output. The analysis did not focus on developing a model that accurately predicted the alphabet characters on out-of-sample data. Nevertheless, a reasonably accurate model was constructed to obtain sensible output results for the analysis.

This research is particularly important for constructing neural networks for optical character recognition (OCR). Many companies, especially those in the financial service industry, are becoming more and more efficient with the help of document automation software solutions that employ OCR. These advanced software solutions allow the companies to increase productivity and customer satisfaction and ultimately, to provide the companies substantial financial benefits.

2. Data Preparation, Exploration, Visualization

The dataset contains 29-element vectors representing English alphabet letters. Each feature is made up of 81-input nodes that form a 9 x 9 matrix. The letters are formed by pixels that are 'on' or 'off' in the matrix. Figure 1 displays examples of the letters. The letters 'V' and 'Y' were excluded from the analysis. As such, only 24 of the 26 letters of the alphabet were used in the analysis. Five additional variants of letters were added to the input vectors. The variant inputs represent letters that are constructed/drawn using alternative pixels within the matrix that result in letters that slightly differ in appearance and pattern from other input versions. The variant letters that were selected include letters 'G', 'P', 'R', and two variants of the letter 'E'. Each letter has been assigned to a class (numbers) 0 through 8. The nine classes represent 'Big Shape' classes based on the overall shape of the letters. For example, the

letters ‘B’, ‘P’, and ‘R’ share many features and are grouped in the same ‘class’. Each class is labeled with a letter that is representative on the ‘Big Shape’ of the class. For instance, Class 1 is labeled with the letter ‘B’.

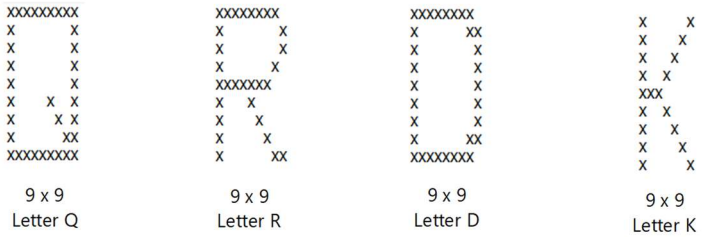


Figure 1: Example of dataset Letters

A Python code template was employed to conduct the analysis. The template was initially set-up to run 16 features/letters and nine classes. The code was modified to activate eight additional letters and to append the five letter variants that were discussed above. Also, the letter ‘Z’ was reassigned to its own separate class resulting to the total number of classes being increased to ten classes. Appendix A lists the classes and their representative letters. Appendix B lists all 29 vectors and their ‘Big Shape’ classification.

3. Implementation and Modeling

The neural network is comprised of one input layer containing 81 inputs, a single fully connected hidden layer with variable nodes, and an output layer containing 10 outputs, one for each class. At the outset, several iterations of the model were ran using various hyperparameter settings. Although the goal of this research was not to develop an accurate classifier, it was decided that the neural network should be fairly accurate in pattern recognition to obtain reasonable results for the analysis. After several tests, a set of hyperparameters that resulted in a reasonably accurate model were settled on: 1) the transfer function with alpha set to 1.25; 2) the learning rate (eta) set to .25; 3) max

iterations equal to 5,000; an epsilon threshold of .01 for the resulting squared summed errors (SSE); and 4) 29 training datasets, one for each of the 29 vectors/letters. Using these parameters, 4 models were constructed using varying numbers of nodes in the hidden layer. The models were constructed using 3,4,5, and 6 hidden nodes, respectively. The models were constructed to classify the feature patterns based on the 'Big Shape' classification of each letter instead of the individual letters.

The model was initiated using randomly set weights to obtain a baseline set of outputs. Following forward-propagation, the weights were passed through the input layer, through the hidden layer, to the output layer. The model performance was evaluated to calculate the loss function (desired output – actual output). To encourage convergence of the neural network, the loss function was defined as the SSE. The neural network was then trained with the 29-vectors for pattern recognition. Another forward-pass was implemented to obtain the actual outputs after training. Backpropagation was then employed to compute the partial derivatives of the loss function with respect to the weights and any bias in the network. Backpropagation performed a back-ward pass to update the weights from the output layer to the hidden layer, and the hidden layer to the input layer with the goal of achieving the actual output be close to the target output. A forward-pass was then performed again to recalculate the SSE scores after back-propagation completed the weight updates. The program generated statistics for the baseline values and the model's performance after training and back-propagation were implemented. Heatmap graphs that display which nodes turned 'on' for each class were constructed.

4. Results

As was discussed above, in testing several hyperparameter settings to achieve a fairly accurate model, a set of hyperparameter settings were selected. The selected parameters resulted in substantially better performance of the neural network. For example, when the model with 6 hidden-nodes was ran, the baseline total SSE score of 115.03

(avg. 3.97) was generated. After training and backpropagation were implemented, the neural network generated a total SSE score of 6.09 (avg. 0.21). While there is still room for improvement, the parameter settings reduced the SSE values by 108.94 or 95% in this example. All four of the constructed models achieved similar performance results after training. The resulting SSE scores of the trained models were much closer to the value of zero than scores of the baseline outputs. However, there is still room to improve the performance of the neural network to reduce the SSE values even further.

Using the same hyperparameter values in constructing all four models, it was hypothesized that a neural network with fewer hidden-nodes in the hidden layer would perform better than the models with larger quantities of hidden-nodes. In this example, the model with the least hidden-nodes, 3 hidden-nodes, performed the worst. This model achieved a total SSE score of 14.27. With a total SSE score of 5.01, the neural network with four hidden-nodes performed the best. The five hidden-node, and six hidden node neural networks generated total SSE scores of 6.24 and 6.09, respectively. It is curious as to why the six hidden-node neural net performed better than the five hidden-node neural network. At the very least, the sex hidden-node neural network was able to more accurately classify patterns to the classes 1-C and 9-Z. Appendix C lists the SSE scores of the four neural networks by class. It appears that none of the nodes 'died' during the testing and all completed their tasks.

Given that the four hidden-node neural network achieved the lowest total SSE score among the four constructed models, this model was used for the analysis of how neural network has learned to represent features within the input data. Table 1 lists a summary of the classes that activated the four nodes of the neural network.

Node	Activated by Big Shape Class (Number-Letter)
H0	1-B; 2-C; 3-O; 5-I; 9-Z
H1	2-C; 3-O; 4-E; 5-I; 7-L; 9-Z
H2	0-A; 1-B; 2-C; 4-E; 5-I; 6-K; 7-L; 9-Z
H3	0-A; 1-B; 2-C; 3-O, 4-E; 8-M; 9-Z

Table 1: Hidden-node activations by Big Shape Class

H0: The first hidden node responds to the Big Shape classes represented by the letters B, C, O, I and Z. The class letters B, C, and O share the pixels that form the outer top, bottom, and left borders of the grid. The class letters Z and I were also selected. The only features that these letters share with the other three letters is are the pixels found at the top and bottom borders of the matrix. Because the letter classes A and E did not activate the node, both of which have pixels at the top border of the grid, it is estimated that it was the bottom border of the grid that activated the node. It is likely that pixels in the top border deactivates the node.

H1: The second hidden node responds to the classes represented by the letters C, O, E, I, L, and Z. These class letters are similar to the classes that H0 responded to. The letter class B was replaced by the letter class E. The letter E shares pixels with the letter B in the upper, bottom and the left borders of the grid. It also shares the pixels in the middle of the grid that for the middle line of the letters. The letter class L that was also selected shares the left and bottom border pixels with the letter classes C, O, and E. Because the letter classes A and B did not activate the node, it is likely that the pixels that form the center line do not activate the node. The letter class for M also did not activate the node. This is also curious because the class also shares pixels on the outer left and right borders as with some of the letters that activated this node. It is believed that pixels found at the top and the bottom borders of the grid are activating the node.

H2: The third node responded to the classes characterized by the letters A, B, C, E, I, K, L, and Z. Based on the pixels in the center of the grid that these letters share, it is believed that it is these areas that activated the node. It appears that the letters that have pixels on the right border of the grid may deactivate the node.

H3: The final node was activated by the classes characterized by the letters A, B, C, O, M and Z. The letter classes I, K, and L did not activate the node. Based on this it is estimated that the pixels found at the center-top border and the center right border triggered the node. It is likely that the node also responded to the pixels that form the upper and lower portions of the diagonal lines of the letters.

Based on the above, it appears that the neural network may group different features to be represented if the neural network has several outputs and comparatively fewer hidden nodes. Appendix D displays a heatmap of the nodes and the classes that they responded to. Appendix E displays a table that displays the hidden node output activation values grouped by the Big Shape Class.

Appendix D displays a heatmap of the nodes and the classes that they responded to. Appendix E displays a table that displays the hidden node output activation values grouped by the Big Shape Class.

5. Conclusion and Recommendations

One of the challenges of constructing the neural network classifier is identifying the optimal number of hidden nodes to use in the model. Too many nodes could result in the hidden-nodes not generalizing well and becoming specialized in representing features; or the nodes may fail to train. Noise and Bias may also adversely impact the performance of the hidden-nodes if there are too many. Too few hidden-nodes also presents challenges. By constructing several models using, the same hyperparameters using varying numbers of hidden-nodes, it was possible to observe how the number of hidden-nodes can impact the accuracy of the neural network.

As regards how the neural network learns to represent features of the input values, a clear pattern was not confidently observed in the experiment. While some of the nodes responded to several letter classes that share many pixels, in some cases the nodes also responded to letter classes that appeared to share very few pixels with the other letter classes. It is predicted that the pixels turned 'off' (the spaces in the letters such as the space inside the letter 'O') may play a significant role on how the hidden-nodes are learning to represent the features. But such a comparison is not easily completed by visually examining the letters. A speciated graph would be needed to compare the 'off' and 'on' pixels that are shared by the letter classes. It is also believed that the neural network may group different features to be represented if the neural network has several outputs and comparatively fewer hidden nodes.

Based on the above, it is recommended that efforts to improve the accuracy of the neural network, by lowering the after-training SSE score, should continue by testing various hyperparameters. It is also recommended that different hidden-node quantities be tested with the continued testing of hyperparameter values to find the optimal number for the tested settings. It is recommended that heatmap graphs that highlight which pixels of the letter classes turn ‘on’ and ‘off’ when a node responds to the input.

Appendix A

‘Big Shape’ Classes

Class Code	'Big Shape' Class Letter
0	A
1	B
2	C
3	O
4	E
5	I
6	K
7	L
8	M
9	Z

Appendix B

Alphabet Letter by 'Big Shape' Class

Letter	Class Code	'Big Shape' Class Letter
A	0	A
B	1	B
C	2	C
D	3	O
E	4	E
F	4	E
G	2	C
H	0	A
I	5	I
J	5	I
K	6	K
l	7	L
M	8	M
N	8	M
O	3	O
P	1	B
Q	3	O
R	1	B
S	4	E
T	5	I
U	7	L
W	8	M
X	8	M
Z	9	Z
E v1	4	E
G v1	2	C
P v1	1	B
R v1	1	B
E v2	4	E

Appendix C

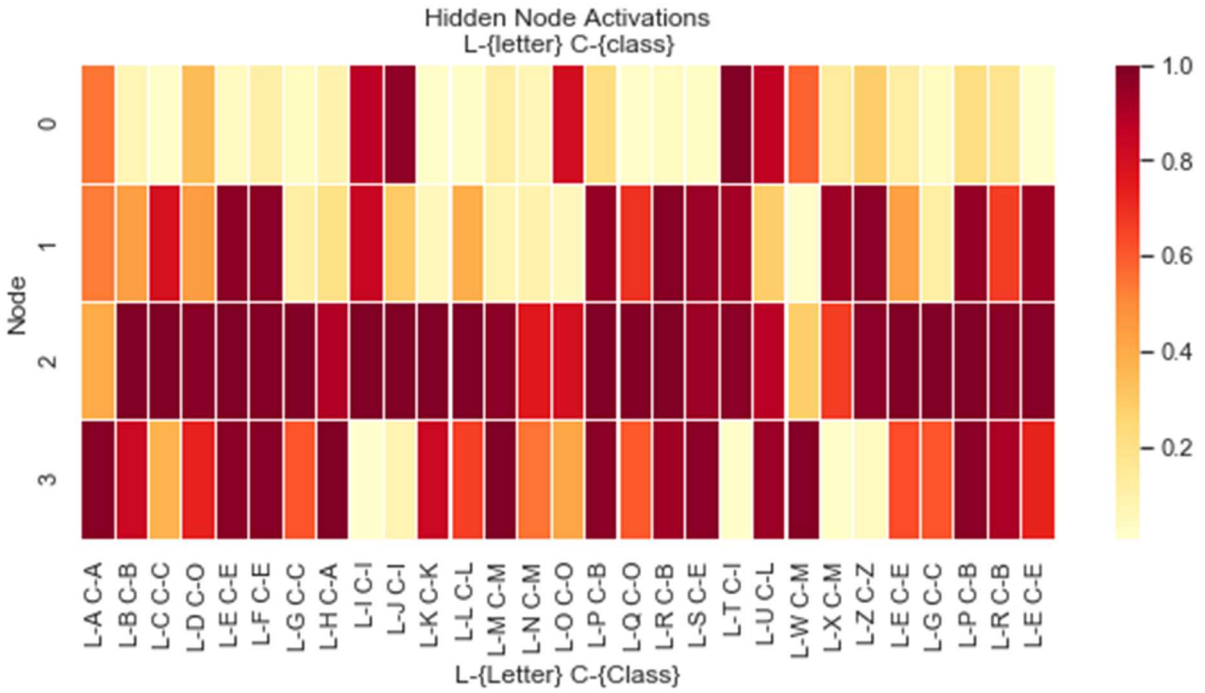
SSE Score after Training by Class and Neural Network

			SSE Score			
Letter	Class Code	Class Letter	3 Hidden - Node	4 Hidden - Node	5 Hidden - Node	6 Hidden - Node
A	0	A	0.0741	0.1519	0.0407	0.0745
B	1	B	0.3749	0.0208	0.0531	0.0412
C	2	C	0.9200	0.7967	0.8394	0.4603
D	3	O	0.7832	0.0272	0.0676	0.0401
E	4	E	0.6774	0.0389	0.0338	0.0283
F	4	E	0.3987	0.0587	0.0457	0.0422
G	2	C	0.9600	0.8826	1.1903	0.4884
H	0	A	0.2381	0.1512	0.0498	0.3183
I	5	I	0.0440	0.0778	0.0496	0.0156
J	5	I	0.0461	0.0844	0.0520	0.0323
K	6	K	1.0128	0.2012	0.9403	1.0285
l	7	L	0.9518	0.2453	0.3727	0.6475
M	8	M	0.1169	0.0593	0.0377	0.0568
N	8	M	0.1180	0.0462	0.0386	0.0464
O	3	O	0.7778	0.0260	0.0669	0.0363
P	1	B	0.3978	0.0270	0.0590	0.0232
Q	3	O	0.8178	0.0295	0.0678	0.0447
R	1	B	0.3588	0.0253	0.0552	0.0156
S	4	E	0.3588	0.0384	0.0351	0.0314
T	5	I	0.0429	0.0822	0.0443	0.0210
U	7	L	0.8658	0.1230	0.3413	0.7006
W	8	M	0.1207	0.0449	0.3469	0.2167
X	8	M	0.1303	0.0487	0.1105	0.0439
Z	9	Z	1.0162	1.0055	1.0909	0.3736
E v1	4	E	0.6952	0.0477	0.0390	0.0315
G v1	2	C	0.5460	0.5841	0.0214	1.1535
P v1	1	B	0.3978	0.0270	0.0590	0.0232
R v1	1	B	0.3827	0.0209	0.0527	0.0224
E v2	4	E	0.6495	0.0419	0.0387	0.0319
		Total SSE	14.2742	5.0143	6.2398	6.0898
		Average SSE	0.4922	0.1729	0.2152	0.2100

Appendix D

Hidden Node Activations Heat Map for

4 Hidden-node Neural Network



Appendix E

Hidden Node Activation by Class after Training

			Hidden-Node			
Letter	Class Code	Big Shape Class Letter	H0	H1	H2	H3
A	0	A	0.005	0.010	0.994	0.995
H	0	A	0.008	0.000	0.986	0.999
B	1	B	0.995	0.152	1.000	0.914
P	1	B	0.947	0.166	1.000	0.999
R	1	B	0.941	0.013	0.998	1.000
P v1	1	B	0.947	0.166	1.000	0.999
R v1	1	B	0.975	0.005	0.999	0.994
C	2	C	0.671	0.999	0.997	0.438
G	2	C	0.946	0.879	0.957	0.802
G v1	2	C	0.946	0.879	0.957	0.802
D	3	O	1.000	0.994	0.027	0.918
O	3	O	1.000	0.990	0.004	0.847
Q	3	O	0.996	0.982	0.032	0.989
E	4	E	0.035	0.999	1.000	0.992
F	4	E	0.104	0.931	1.000	0.999
S	4	E	0.037	0.999	0.998	0.998
E v1	4	E	0.088	0.998	0.991	0.965
E v2	4	E	0.067	0.995	0.996	0.990
I	5	I	0.989	0.999	0.995	0.002
J	5	I	0.925	0.984	0.999	0.001
T	5	I	1.000	0.996	0.982	0.029
K	6	K	0.000	0.008	0.998	0.034
I	7	L	0.004	0.953	0.995	0.049
U	7	L	0.028	0.950	0.292	0.085
M	8	M	0.001	0.001	0.015	1.000
N	8	M	0.010	0.000	0.003	0.993
W	8	M	0.136	0.006	0.002	0.986
X	8	M	0.001	0.070	0.018	0.842
Z	9	Z	0.714	1.000	0.797	0.595