Optical Character Classification:

A functional examination of feature extraction in artificial neural networks

Christopher J. Rico

Northwestern University

Introduction and Problem Statement

Deep Neural Networks (DNN) are a cutting-edge approach to machine learning and artificial intelligence, consisting of one to many layers of hidden nodes sandwiched between a layer of input and output nodes. As a network is trained on a dataset, hidden nodes perform undirected feature extraction by reacting to particular combinations of input values. Teasing apart the input data features that a network is reacting to remains a difficult task, although one which has great value to those who wish to understand the inner workings of DNNs.

This paper details an experiment wherein 2 DNNs with various topologies are trained on the basic task of classifying letters into 8 gestalt "big shape" classes that have been determined one of two ways: either through k-means clustering, or heuristic examination. Hidden node activation values are examined to gain better insight into how DNNs encode data and extract features.

Dataset

The dataset consists of 26 capital letters A-Z. Each letter is encoded as a Python list of 81 binary integers, wherein values of 1 or 0 are used to create a 9x9 bitmap representation of a letter.

Letters were sorted into 8 big shape classes via either k-means clustering or heuristic examination, resulting in two training datasets with identical character representations but different class assignments. Both datasets and their big class assignments can be seen in Appendix A and B.

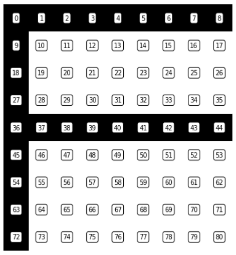


Fig 1 – 9x9 bitmap representation of letter F

Research Design and Methods

This experiment was performed using the provided code and executed in Google Colaboratory (a cloud-based Jupyter notebook) running a Python 3 interpreter.

This experiment endeavors to explore how various numbers of hidden nodes (HN) and varying methods of character clustering affect the way that DNN HN encode information about combinations of input data. To explore the effects of both of these factors, a fully crossed 2x2 experimental design will be employed. 4 different networks will be built and trained with either 6 or 10 hidden nodes, and 8 big shape classes determined either via SciKit-Learn k-means clustering or heuristic examination. Because these two different class assignment methodologies differ in how abstract they are, each big shape classification is quite different from the other. With heuristic clustering, the *relative* positions of pixels (i.e., high-level 'big shapes') are the main discriminating factors. K-means clustering, on the other hand, is more interested in *absolute* positions of pixels, ignoring big shapes. I hypothesize that the differences in class assignment will cause very different combinations of input pixels to be recognized by each DNN's hidden nodes.

Because the dataset is very small – some big shape classes only contain one or two characters – 10% noise will be introduced into the training data to help prevent overfitting. Noise introduction is implemented by "bit-flipping" random pixels in the training dataset at a given frequency, creating subtle variability in the input bitmaps.

Results and Analysis

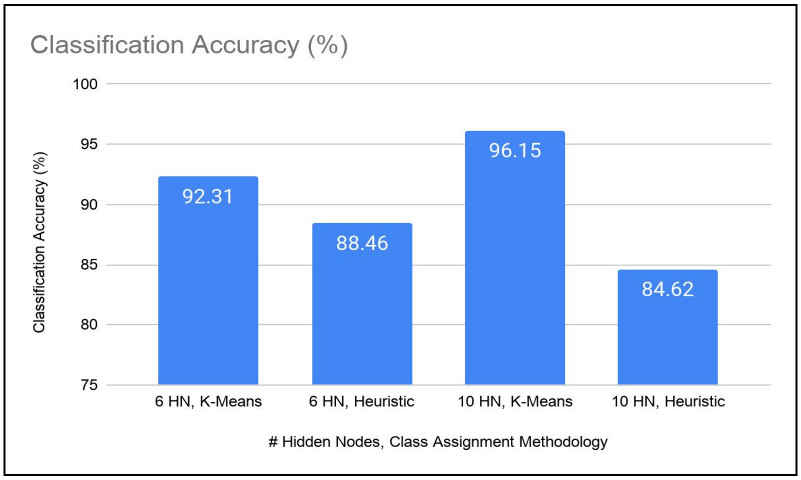


Fig 2 – Classification accuracy of each DNN

Right away, it is clear that both factors have an effect on each network's ability to classify characters. Networks that are trained using the k-means assigned classes have a higher percent accuracy than those trained using the heuristically assigned classes. However, while 10 HN gives higher accuracy over 6 HN paired with the k-means dataset, the opposite is true for 10 HN trained on a heuristic dataset.

Feature Extraction

Looking at the 9x9 weighted heatmaps below, it becomes clear that assigning characters to “big shape” classes causes HN to pick out much larger features than if each letter were its own class. This is not a huge surprise: greater variation within a class tends to force a network to recognize and generalize about broad pixel similarities between class instances instead of smaller, idiosyncratic pixel combinations. This is similar to the effect of introducing noise.

In general, HN in networks trained using the heuristically classed dataset exhibit greater differentiation in features extracted. In other words, each node seems to be responding more clearly to an individual feature, instead of being a muddy mix of features.

6 Hidden Nodes

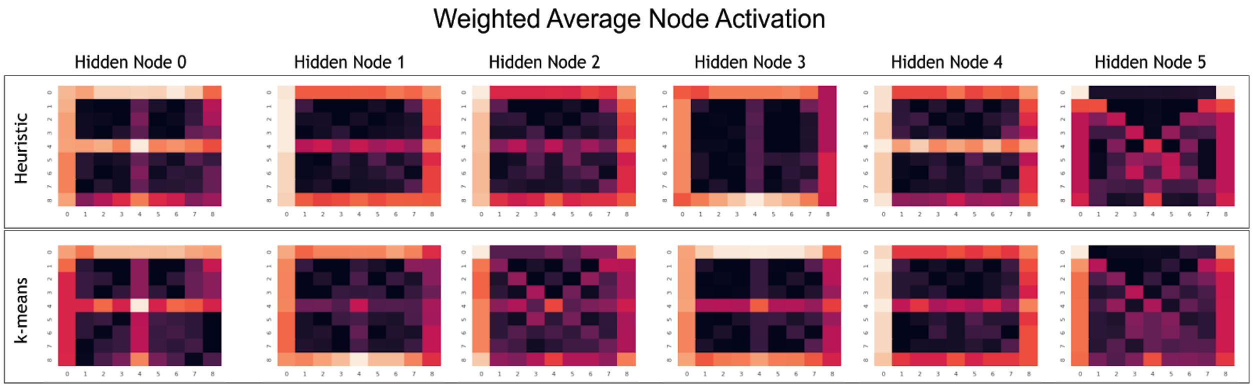


Fig. 3 – Weighted average hidden node activation for 6-HN networks

The difference in hidden node activations between heuristic and k-means clustered letters is striking. One of the most obvious patterns is the very strong activation of HN 0 and 4 of the heuristic network by pixels that form an “E” or “F” shape – top horizontal bar, middle horizontal bar, left vertical bar. Class B (cluster 1) of this dataset contains letters with mostly these features: B, E, F, H, P, R. In the k-means dataset, these letters are mostly spread throughout different classes, so, similar to the aforementioned example, we do not see any HN in the k-means network lighting up particularly strongly for these pixel combinations.

Also of interest is the activation of heuristic network's HN 5. The heuristic dataset's class M (cluster 6) contains a variety of letters with outer-top → center-middle diagonals: M, N, V, W, X, Y, Z. We can see that HN 5 responds strongly to this pattern, as well as both top corner pixels being activated. The k-means dataset, on the other hand, considers H, M, N, and W in the same class, and we can see that HN 5 in this network lights up much less strongly for this combination of pixels in any of its HN.

10 Hidden Nodes

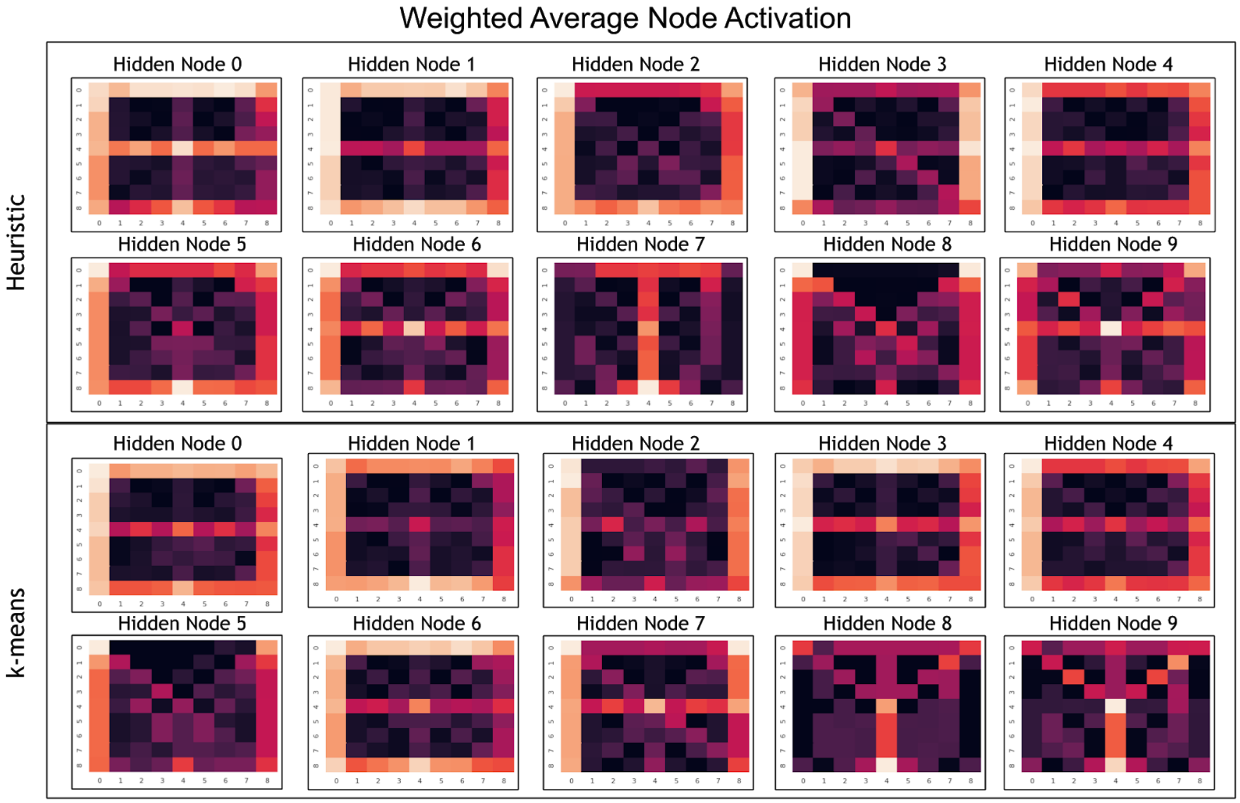


Fig. 3 – Weighted average hidden node activation for 6-HN network

In DNN with 10 hidden nodes, we can see that the greater number of HN allows for some nodes to extract similar features: fully half of the HN show a fairly strong activation for pixels forming a vertical bar on the left side of the grid.

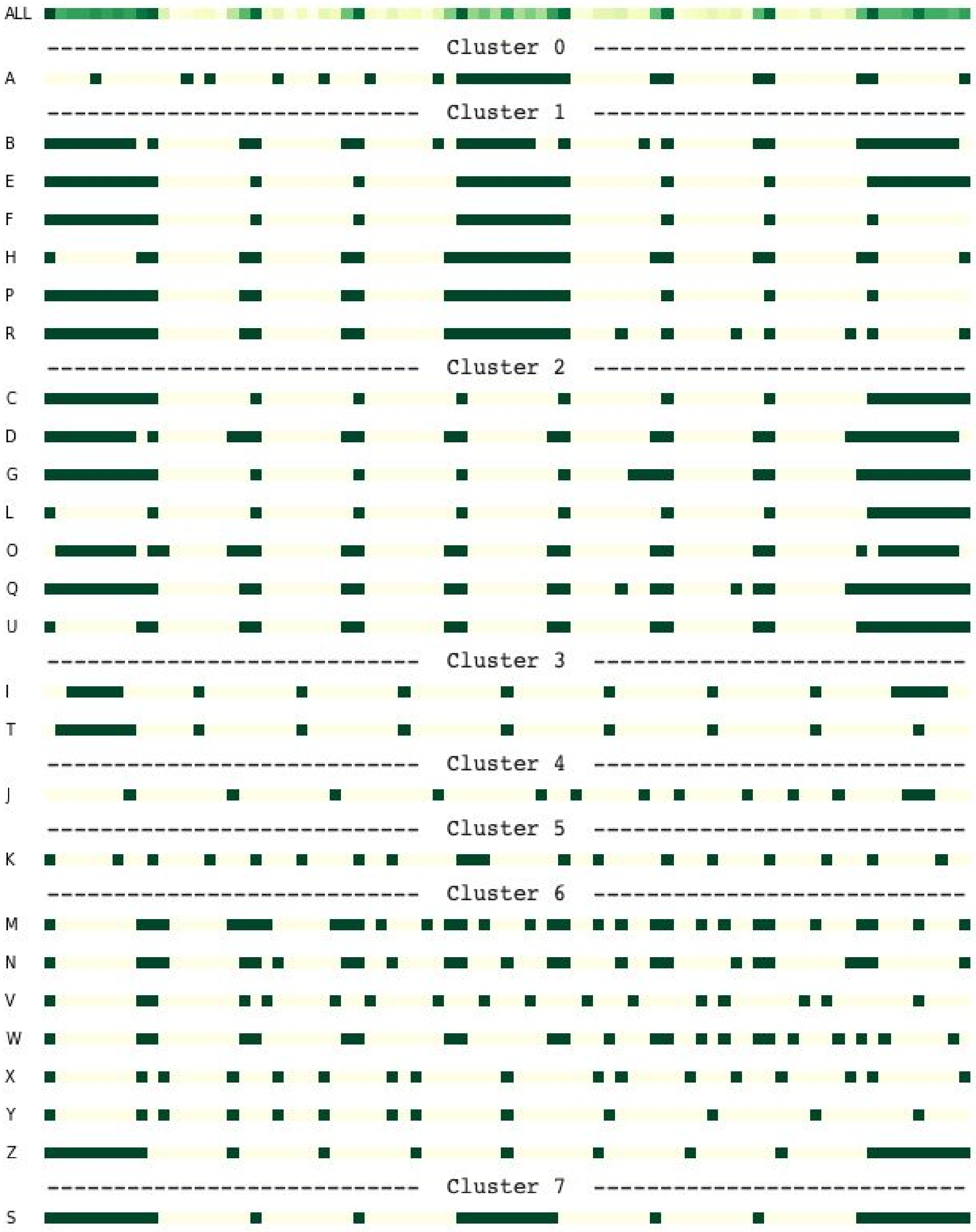
On the other hand, the difference in activation values between HN 7 in the heuristic network and HN 8, 9 in the k-means network are fairly confounding: in both datasets, letters I and T are alone in a class. HN 7 in the heuristic network is clearly reacting to the pixel combinations in these letters. We would expect to see a similar reaction in one of the HN of the k-means network, but instead we see HN 8, 9 reacting to a center vertical bar and two diagonals coming from the top corners to the middle, like a Y shape. The difference between these is that in the heuristic dataset, Y is classed with N, V, W, X, Z -- none of which have the center vertical bar. However, in the k-means dataset, Y is classes with X and J. Apparently the inclusion of J in the same class as Y is enough to cause HN 8, 9 in the k-means network to react strongly to a center bar *and* diagonals from the top corners to middle.

Conclusion

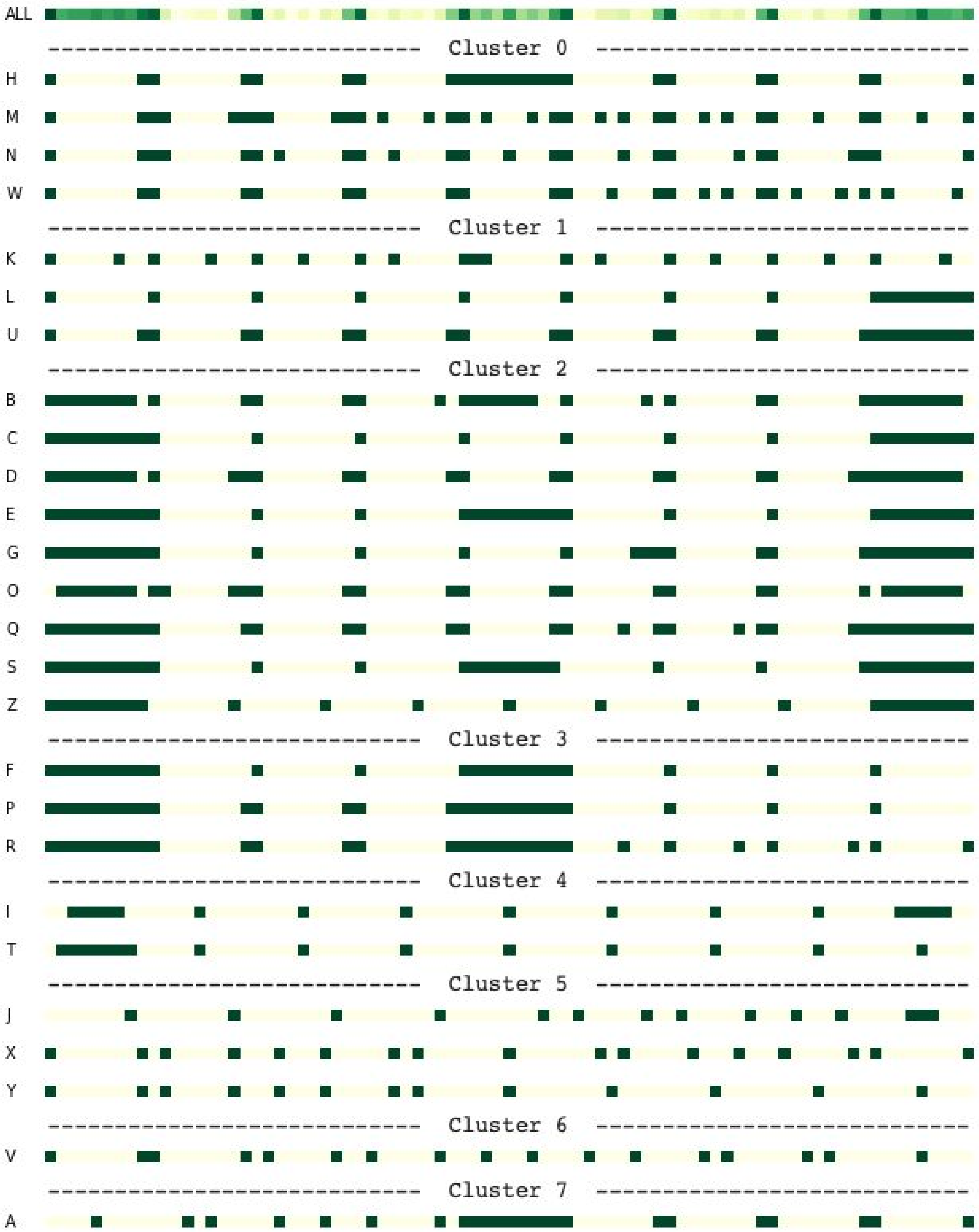
By training and analyzing the hidden node activation values of these four networks, I was able to observe how DNN hidden nodes encode training data. Furthermore, I was able to explore how differentiated class assignment in the training dataset affects which features the hidden nodes opt to extract.

In the future, I think it would be valuable to use some of the built-in visualization tools that Keras and Tensorflow offer in order to view how a more complex CNN might perform feature extraction on this dataset – or a more complicated one like MNIST.

Appendix A. Heuristic Big Shape Class Assignments



Appendix B. K-Means Big Shape Class Assignments


ALL
Cluster 0
H
M
N
W
Cluster 1
K
L
U
Cluster 2
B
C
D
E
G
O
Q
S
Z
Cluster 3
F
P
R
Cluster 4
I
T
Cluster 5
J
X
Y
Cluster 6
V
Cluster 7
A