A New Model of Artificial Intelligence: Application to Data II

Charles Davi

April 10, 2019

Abstract

In this article, I'm going to apply the new, polynomial time model of artificial intelligence that I've developed to the MNIST numerical character dataset, as well as two small image datasets made with an ordinary iPhone camera. The MNIST dataset was analyzed on a supervised basis, with a success rate of 95.402%, where success is measured by the percentage of predictions that are consistent with the classification data. The other two datasets were analyzed on an unsupervised basis, with an average success rate of 95.834%. All of the code necessary to run these algorithms, and apply them to the training data, is available on my researchgate homepage.[1]

1 Introduction

In a previous working paper,[2] I introduced a new model of artificial intelligence rooted in information theory that can solve high-dimensional, machine learning problems in polynomial time by making use of data compression and vectorized processes. Specifically, I introduced an image feature recognition algorithm, a categorization algorithm, and a prediction algorithm, each of which has a low-degree polynomial run time, allowing a wide class of problems in artificial intelligence to be solved quickly and accurately on an ordinary consumer device.

In this article, I'm going to apply the categorization algorithm and prediction algorithm to three datasets: the MNIST numerical character dataset, and two

[1] I retain all rights, copyright and otherwise, to all of the algorithms, and other information presented in this paper. In particular, the information contained in this paper may not be used for any commercial purpose whatsoever without my prior written consent. All research notes, algorithms, and other materials referenced in this paper are available on my researchgate homepage, at https://www.researchgate.net/profile/Charles_Davi, under the project heading, *Information Theory*.

[2] *A New Model of Artificial Intelligence*.

small datasets of images made using an ordinary iPhone camera, one consisting of photos of a pair of headphones and a set of speakers, and the other of photos of the handwritten letters A and B.

2 Supervised Image Classification

The MNIST Dataset

The MNIST dataset is courtesy of the Institute of Standards and Technology, though the version used in this paper was converted to jpeg format by a third party.[3] The algorithm begins by reading the jpeg file into a matrix, removes any pixels that are approximately black, and then stores the locations of the remaining pixels in two column vectors that contain the horizontal and vertical index of each pixel. Figure 1 shows the result of pre-processing an image representing the digit 0.

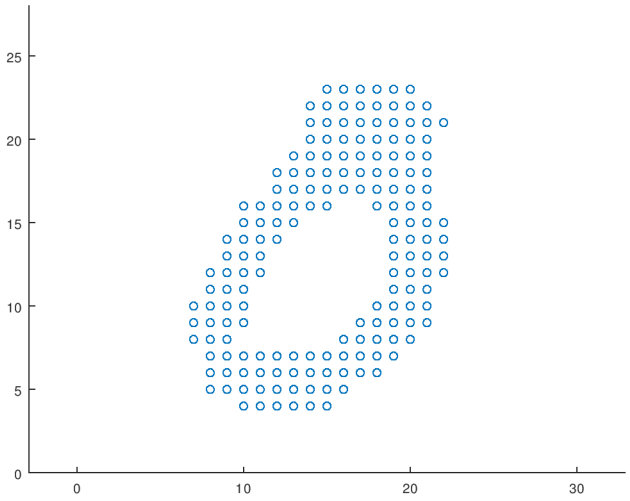

Figure 1: The result of pre-processing an image representing the digit 0.

After this first step, the resulting plot of pixels is then subdivided into 121 equally sized rectangular regions, and the number of pixels in each region is then counted, and stored as an 11×11 matrix. The matrix is then reshaped into a 1×121 vector, but is otherwise unchanged. This vector serves as the input

[3] https://github.com/teavanist/MNIST-JPG

to the categorization and prediction algorithms.[4] Figure 2 shows the matrix generated for the set of pixels shown in Figure 1.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 4 & 5 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 7 & 6 & 9 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 6 & 6 & 0 & 1 & 8 & 6 & 2 & 0 & 0 & 0 \\ 0 & 3 & 8 & 1 & 0 & 4 & 9 & 9 & 0 & 0 & 0 \\ 0 & 0 & 6 & 8 & 6 & 8 & 9 & 9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 6 & 4 & 3 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 2: The matrix generated by counting the number of pixels in each region of the image shown in Figure 1.

This process was applied to 500 images from each of the 10 categories of digits in the MNIST training set, for a total of $5,000$ images. The resultant vectors for each category were then separately provided as the input to the categorization algorithm, which in turn generated 10 sets of categories, one for each class of digit. The predictions were generated by then providing 10 new images from each category of digit (i.e., images that were not included in the initial $5,000$ images), for a total of 100 predictions. The algorithm produced 83 correct classifications, 4 incorrect classifications, and 13 rejected classifications, implying an accuracy of either $\frac{83}{100} = 83\%$, or $\frac{83}{87} = 95.402\%$, depending upon whether we do, or do not, include rejections in the denominator.[5]

3 Unsupervised Image Classification

The Speakers and Headphones Dataset

This dataset consists of 20 photos of a pair of speakers, and another 20 photos of a pair of headphones, for a total of 40 photos, each taken on an ordinary iPhone camera. The photos were taken on an off-white background, with the position, and to a lesser extent, the orientation, of the objects being somewhat idiosyncratic to each photo. An example of a speaker photo and a headphone photo is shown in Figure 4.

[4] For an explanation as to how the categorization and prediction algorithms work, see the previous working papers referenced in the footnotes above.

[5] The prediction algorithm rejects any data it believes to be beyond the scope of the original training data, rather than make a potentially erroneous prediction. See the previous articles for an explanation as to how this process works.

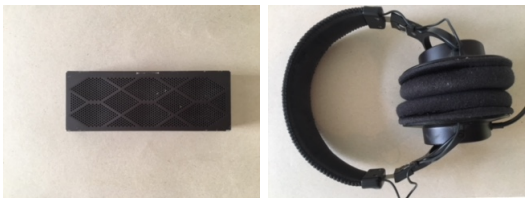

Figure 3: An example of a speaker photo and a headphone photo from the dataset.

The algorithm begins by applying an edge detection algorithm I introduced in a previous research paper.[6] This algorithm extracts shape information from the image, in effect removing the background of the image, and extracting a set of points that outline the contours of the image. The results of this process, as applied to the two images in Figure 3 above, are shown in Figure 4 below. We then apply the same algorithm described above, that counts the number of points in each region of the resultant shape, producing a matrix, and in turn a vector that serves as the input to the categorization algorithm.

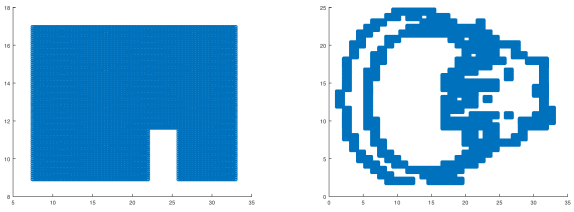

Figure 4: The shapes extracted from the photos in Figure 3 above.

In this case, both classes of images are combined into a single data array and fed to the categorization algorithm on an unsupervised basis. We measure the success rate of this process by counting the number of categories that consist of only a single class of images. If a category contains a single image that is of a different class than the other images in that category, then we treat that category as an error. The success rate is then the number of error-free categories divided by the total number of categories. In this case, the success rate was 100%. That is, each of the categories generated consisted of only headphones or speakers, and never both.

The Handwritten Character Dataset

This dataset consists of 20 photos of a handwritten A, and another 20 photos of a handwritten B, for a total of 40 photos, each taken on an ordinary iPhone camera. The characters were drawn on a ruled page, and as a result, there is

[6] *Unsupervised 3D Feature Extraction and Edge Detection Algorithm.*

an idiosyncratic amount of underhang and overhang in each photo. The photos are also deliberately sized differently, and a bit off-center. Examples of an A and a B from the dataset are each shown in Figure 5.

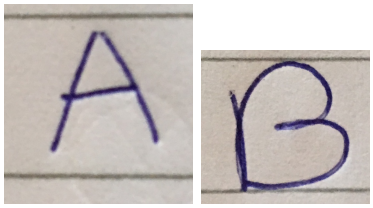

Figure 5: An example of an A and a B from the dataset.

The same process applied to the "Speakers and Headphones" dataset described above was applied to this dataset. The results of the edge detection algorithm, as applied to the images in Figure 5, are shown in Figure 6.

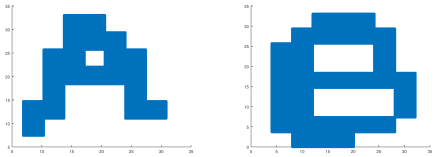

Figure 6: The shapes extracted from the photos in Figure 5 above.

Both classes of images were again combined into a single data array and fed to the categorization algorithm on an unsupervised basis. In this case, the success rate was 91.667%.