COMP 4102 Project Report

Handwritten Digits Recognition using SVD

Course: COMP 4102
Date: Apr 17, 2021
Student Name: Haiyue Sun
Student Number: 100864611

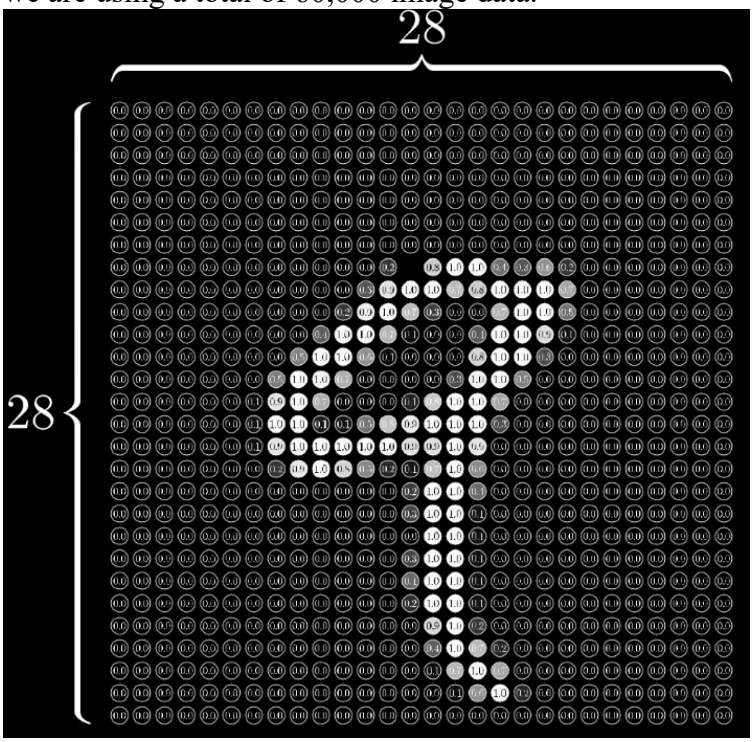
Table of Contents

1. Abstract

Recognizing handwritten digits is a basic topic in the field of neural network. However, this project is going to approach this objective without anything neural network related. But using a kind of matrix decomposition techniques. Among these techniques, Singular Value Decomposition (SVD) has the most generality and greatest ability to save the information of a matrix.

2. Introduction

In this algorithm, we need first to find out what makes a digit that digit. In another word, what are the differences between digits 0 to 9. So, we need a set of labeled data of all digits to train our algorithm. With the term of "labeled data", it means pixel data of an image of handwritten digit corresponding to a correct label of what digit is for this image.
With each of these labeled data, we slice the image line by line and connect each line to a single array. Get the transpose of each array and make it a vertical vector. Then according to the label, join these vectors with group of digits and make a matrix of each digit. Since the images, we are using, are all 28 by 28 pixels. Up to here, we have 10 matrices for 10 digits (0 to 9), each matrix has 784 (28 * 28 pixels) rows and about 6000 columns. Each column represents one image, and we are using a total of 60,000 image data.



Then, calculate SVD for each matrix above. Left singular vectors (U) are all we need. Left singular vector is a 784 by 784 matrix, and we have 10 of them. These 10 matrices are feature matrix of 10 digits. In another word, these matrices indicate differences between 10 digits. As the weight decrease to a very small number after several columns of U, so we can omit some columns of these Us.

We will be able to classify a give image. With a give image, we create the same vertical vector as we did before and call this vector z. Then use the left singular vectors, we calculate how much residual this z has to each digit. Finally, z will be the digit which has the minimum residual. The formula to calculate residual of each digit is given by: $\|(I - UU^T)z\|$, according to Lassiter[1].

3. Background

MNIST is database of handwritten digits. This database contains a set of 28-by-28 pixels training images of 60,000 examples with labels, and 10,000 examples of test image. Files from MNIST are NOT collections or a zipped folder of images. They are simple binary pixel data of images. Please refer to the website to check the detail for the file structure.

4. Approach

Performance of this algorithm depends on how we omit left singular values (U). The application will be so slow if we don't omit or omit too less. In the other hand, if we omit too much, the accuracy will drop. So, we need to find a balance between execute speed and accuracy on the result.

Therefore, I made a test program to find the best omit value k for this algorithm. I have tested accuracy with the value of k from 1 up to 50.

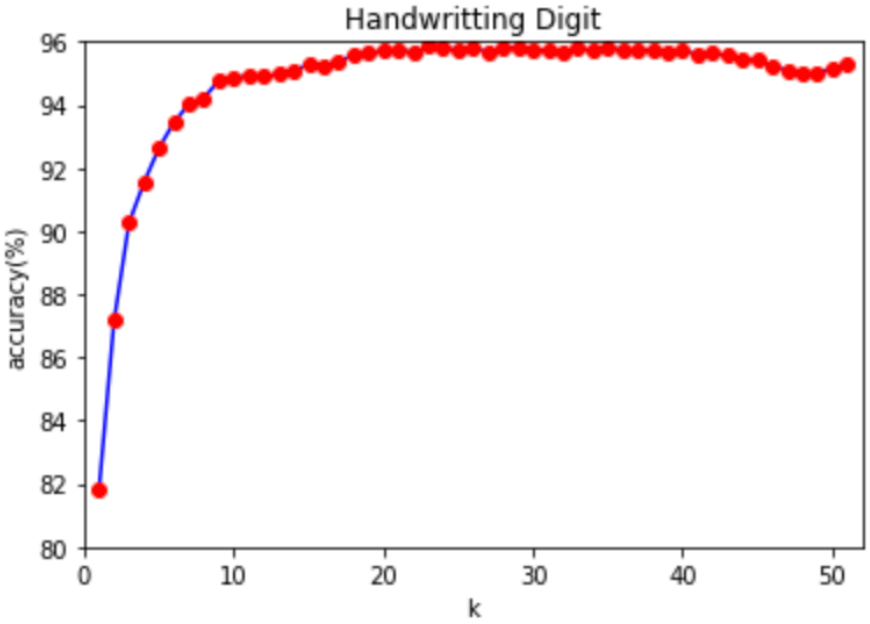


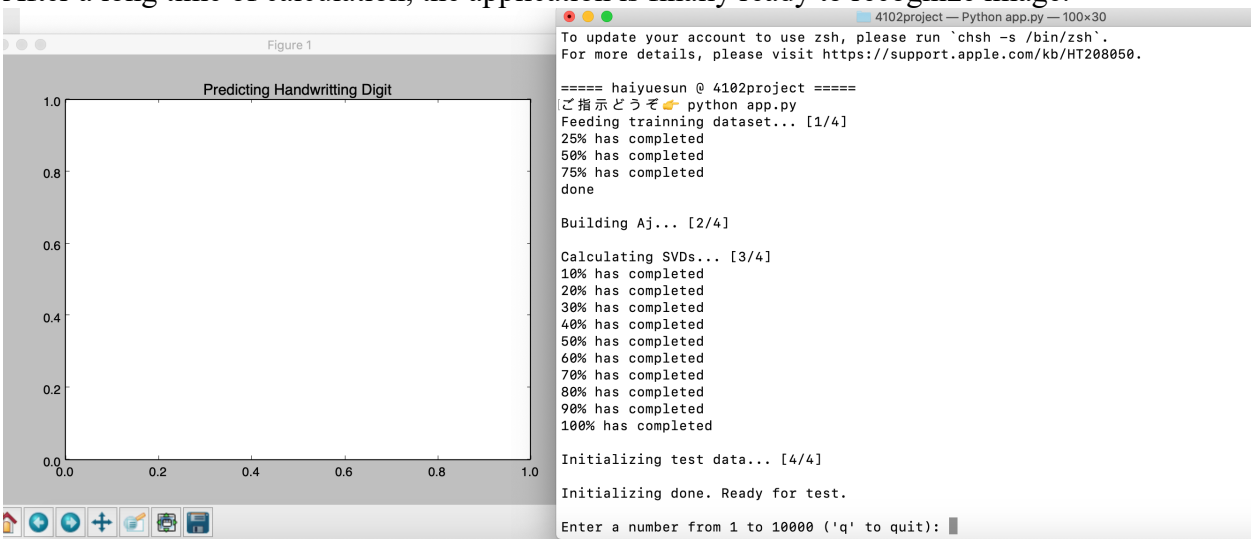
Figure 1 : Accuracy with different k values

[1] Andy L. (2013). Handwritten Digit Classification and Reconstruction of Marred Images Using Singular Value Decomposition. https://intranet.math.vt.edu/ugresearch/Lassiter_2012_2013.pdf

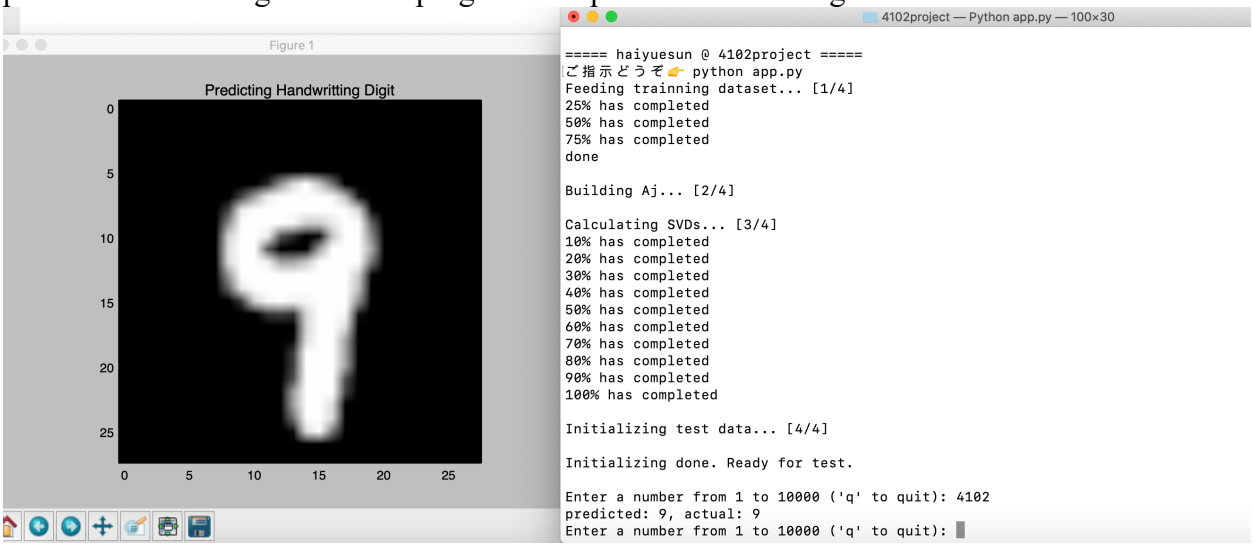
As the result above, 28 will be the most reasonable value for k. Then I take only 28 columns for U in my application.

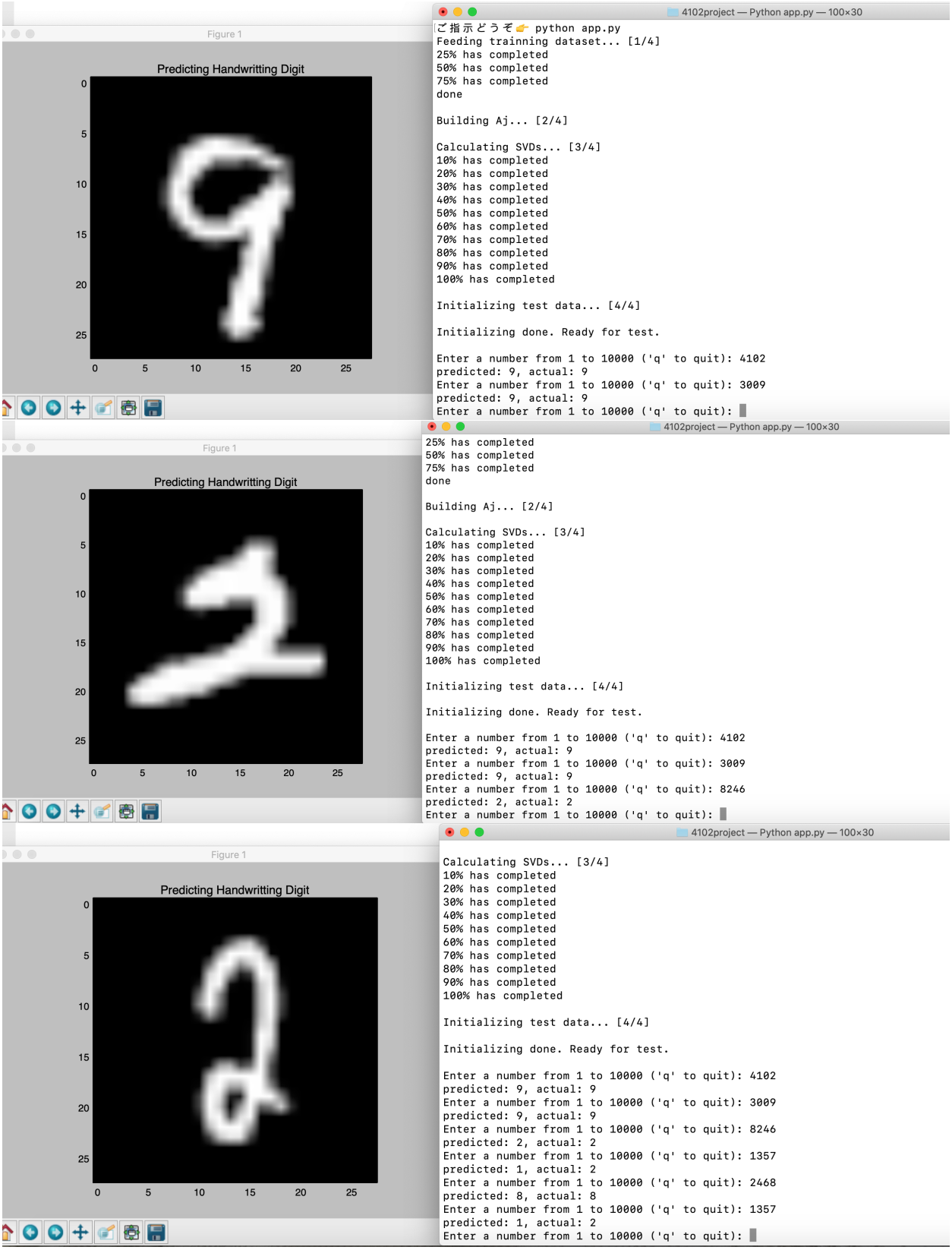
5. Result

After a long time of calculation, the application is finally ready to recognize image.



As we have been given 10,000 examples for testing, and they are not in form of images. We just pick an index among them. The program will predict what the digit is.





As this program has a 95.8% of accuracy, therefore not all images can be predicted perfectly. In the case above, digit 2 was written too thin, so the program predicted it as a 1.

6. GitHub Page

https://github.com/DishySun/4102project

For compiling and executing instruction, please refer to README in GitHub page