Batuhan Erden

batuhan.erden@ozu.edu.tr

CS 423/523 – Introduction to Computer Vision

Assignment 2

10.12.2017

Sudoku Puzzle Recognition

Abstract

In this assignment, I have used Python and OpenCV to detect the digits in the Sudoku image. To do so, I used MNIST dataset in my training and trained my model using PCA (Principal Component Analysis). As a classification, I made use of the Euclidean Distance to see the results of my algorithm.

1 Introduction

At first, I have downloaded MNIST data and start working with it. The second thing I did was implementing PCA. After the implementation of the PCA, I have researched couple of classification methods. Then, I implemented my classification method using the Euclidean Distance. When these steps are completed, I have tested PCA on MNIST dataset using this classification technique. As a next step, I have tried to integrate PCA into Sudoku dataset without touching anything in the base code and divided the Sudoku puzzle into 81 pieces such that there were 9 columns for 9 rows. Then I have tried to check the results but the results were so discouraging. After doing some research, I realized that the cropped image should have only contained the digit. However, when I divide the puzzle into 81 pieces, the lines of the puzzles were also being cropped with the digits and these were blocking classification method from doing its job properly. So, I decided to use my assignment 1. In assignment 1, the code was finding most of the squares in the image. So, I used that squares to extract digits from Sudoku. In conclusion, the results were encouraging and I continued with the implementation of the confusion matrix.

2 PCA (Principal Component Analysis)

PCA, Principal Component Analysis, is a Dimensionality Reduction technique. It emphasizes variation and highlights the strong patterns in a dataset. PCA simply gets a set of observation points and deconstructs these sets' covariance matrix into eigenvectors and eigenvalues. Eigenvector is used to determine the direction while eigenvalue tells you the variance in the data in that direction. They exist in pairs. It can easily be said that the eigenvector with the highest eigenvalue is the principal component. An example is shown in the Figure 1that is taken from lecture slides.

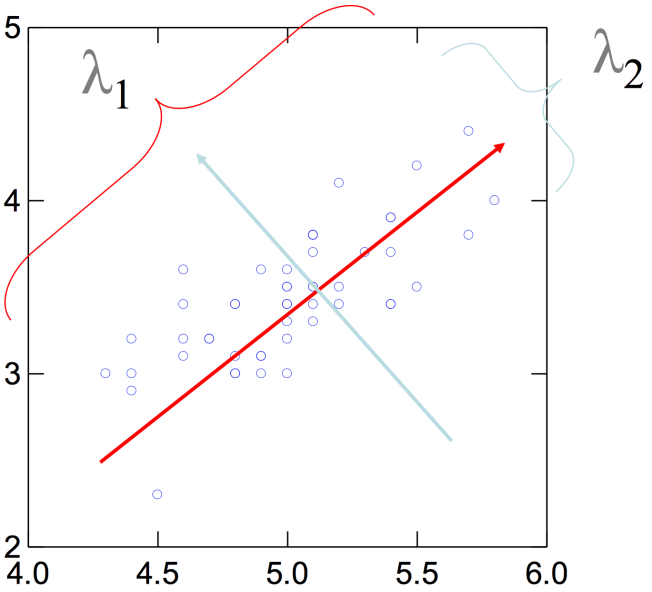

Figure 1: Eigenvectors and eigenvalues

The way I applied PCA in my project is that, I take train images and test images in to PCA and processed them. I first subtracted their means from them. Because subtracting the mean makes variance and covariance calculation easier by simplifying the equations. Besides, the values of both variance and covariance remain unchanged because they are not affected by the mean value. After that, I calculate the covariance matrix of the train images. This is the dot product of train image and transpose of train images. Then using the covariance matrix, I obtain eigenvalues and eigenvectors. After the successful calculation of eigenvalues and eigenvectors, I normalize eigenvalues to do my calculations easier. Next step was sorting eigenvalues from highest to lowest. Finishing the sort operations transfers the algorithm into one of the most important steps. This step was about iterating through each eigenvalue and eigenvector from highest to lowest and summing each eigenvalue up. I collected each eigenvector until the sum is greater than or equal to **e=0.8**. These eigenvectors were the ones that were going to be used. The last step was to multiply these eigenvectors with train images and test images and assign the resulting matrix to train images and test images respectively.

3 Classification Method: The Euclidean Distance

After the researched I made, I came up with a method that makes use of the Euclidean distance to find the nearest index. I chose this method because it was fast. Instead of checking each possible neighbor, it simple subtracts train data from the given sample. Then it squares the resulting matrix and sums each row. At last, it uses the resulting matrix obtained after the summation operation, and finds the smallest number in that matrix and returns the index of that smallest number as a nearest neighbor. In conclusion, when it came to using that nearest neighbor to test my data, it was not that hard. I picked the train label at that index and compared it with my sample.

4 Extracting Digits from Sudoku

Finding the Sudoku and the rectangles within was the subject of the 1st assignment. In this assignment, I made use of the algorithms I wrote in the 1st assignment to extract the digits from Sudoku. The only hard work was sorting the rectangles with respect to their locations in Sudoku. After accomplishing that, I cropped the Sudoku from the image, and then I cropped each rectangle within the Sudoku from Sudoku and processed them one by one. In order for me to get rid of the unwanted lines in the cropped image, I cropped it 12.5% percent from the frame. Then I applied some filters to make the image black-white so that MNIST could recognize it. An example can be shown in the Figure 2.

Figure 2: After thresholding the image

After obtaining a black-white image with a digit inside, I resized it to 28x28 and divided each element of the matrix with 255 to be recognized by MNIST data. On the other hand, I simply found the empty cells by counting the non-zero element in the cropped image. If the count is less than or equal to 50, I classified it as an empty cell and made its value 0. To sump up, this is a semi-abstract definition of how I extracted digits from Sudoku. There were some complex algorithms in the code to fill the place of the rectangles that could not be found in the image.

5 MNIST Dataset Test and Results

At first, I must say that the results of MNIST surprised me. It was 97.33%, which was very good in this context. I have tested it using the 10,000 test images from MNIST dataset. The confusion matrix, false positive/negatives can be seen below:

Confusion matrix where rows are actual values and columns are predicted values.

Actual/ Predicted	0	1	2	3	4	5	6	7	8	9
0	**205**	1	1	0	0	2	1	2	0	0
1	0	**105**	3	1	0	0	1	0	0	1
2	4	1	**240**	2	1	0	3	12	1	0
3	1	0	2	**200**	1	17	0	7	12	2
4	0	1	0	0	**184**	1	2	3	0	23
5	1	1	0	11	1	**92**	7	1	6	4
6	4	4	1	0	3	2	**175**	0	1	0
7	0	11	5	3	3	0	0	**227**	0	11
8	2	0	3	9	2	8	6	4	**169**	3
9	2	3	1	6	15	3	1	7	3	**200**

MNIST dataset performance: 97.33%

6 Sudoku Dataset Test and Results

Compared to MNIST dataset results, the results of Sudoku dataset was much worse as expected. Since MNIST is a hand-written digit dataset, we could not expect a high accuracy on computer-written digits. Yet, the results were not that bad. The average accuracy was 58.45%, which is very acceptable because the qualities of some images in our test dataset were not sufficient enough. I tested it with 203 pictures in our images dataset and compared their results with the actual values that were obtained from the **.dat** files. The confusion matrix, false positive/negatives can be seen below. We can easily say that the prediction for 0 is the worst prediction because when the actual is 0, the prediction on the other digits are higher than the 0 itself.

Confusion matrix where rows are actual values and columns are predicted values.

Actual/ Predicted	0	1	2	3	4	5	6	7	8	9
0	**27**	85	194	159	202	177	147	66	118	103
1	86	**188**	26	10	18	6	7	4	36	3
2	64	22	**211**	10	10	13	14	1	35	9
3	83	27	90	**136**	17	14	13	2	25	7
4	79	17	25	17	**178**	13	10	5	24	20
5	101	14	44	9	12	**114**	40	3	39	10
6	87	18	27	13	14	51	**63**	4	42	6
7	100	134	49	15	15	9	10	**24**	45	8
8	172	27	28	26	30	19	14	5	**108**	34
9	121	33	53	13	19	8	12	6	74	**67**

Sudoku dataset performance: 58.4525119179%

7 Conclusion

In conclusion, the results on the MNIST dataset was higher than expected (97.33%) while the results on the Sudoku dataset was lower as expected (58.45%). I realized that Principle Component Analysis is a very efficient way of solving Digit Recognition problem. Apart from PCA, the most problematic part for me was extracting the digits from the Sudoku. The reason why I had faced so many challenged in doing so was because it was vague how to crop the image. In some of the cases, the digits were also cropped which resulted a bad accuracy on the recognition part. In my opinion, as the 1st assignment, this assignment has also contributed to my Computer Vision skills well. I have learned how my Machine Learning background was related to some specific Computer Vision tasks.

References

- Lecture Slides - https://lms.ozyegin.edu.tr/
- Visual Explanation of PCA - http://setosa.io/ev/principal-component-analysis/