Matching Handwriting With Its Author

Ziran Jiang, Aditya Ramgopal Mundada
Stanford University CS229 Final Project, 12/8/2015

Introduction

Each person has a unique handwriting, and this makes handwriting a useful feature to identify individuals. Handwriting matching can be used by banks for check-writing and signature authentication. In forensic science, handwriting matching algorithm can aid handwriting analysis experts make decision. The goal of this project is to manually implement and optimize handwriting matching, including: 1) Data Acquisition, 2) Image Preprocessing, 3) Algorithm Implementation: Naïve Bayes, SVM, and SVM for three authors, and 4) Algorithm Comparison and Optimization.

Data Acquisition

We considered using the MNIST Database for handwriting samples, but since the samples are random and we do not know the corresponding authors, we could not use them as training or test samples. Instead, we initially collected handwriting samples using MS Paint, and later switched to using an app called INKredible, which allows directly writing on a tablet screen using stylus/finger. The INKredible app enables collecting realistic handwriting samples similar to the samples written on paper. Each handwriting sample was scaled to the following four resolutions using MATLAB : 64x64 pixels, 32x32 pixels, 16x16 pixels, and 8x8 pixels.

Fig. 1. Handwriting samples at different resolutions

We collected 100 handwriting samples from each of the three authors (authors A, B, and C), and scaled each sample to the different resolutions mentioned above.

Fig. 2. Handwriting samples of authors A, B, and C

Image Preprocessing

The preprocessing algorithm was implemented in MATLAB, and it does the following:

1. Make the image black and white, so that each pixel is either 1 (for black letter) or 0 (for white background).
2. Normalize the size of handwriting, center the handwriting using the bounding box method, and resize the image to its original resolution.

Fig. 3. Original image (left) and preprocessed image (right)

Naïve Bayes Algorithm

Naïve Bayes algorithm was implemented to distinguish the handwriting samples from author A and author B. We varied the number of training samples (half and half from authors A and B) and used 100 test samples (50 from author A and 50 from author B). At each number of training examples, the average generalization error was obtained by averaging the generalization error collected over 100 runs, where each run used different randomly picked training samples.

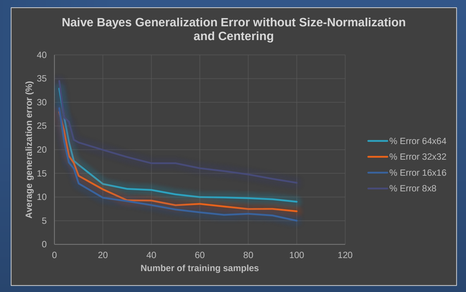

Fig. 4. Naïve Bayes generalization error without size-normalization and centering

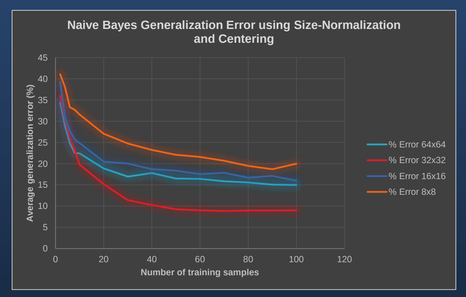

Fig. 5. Naïve Bayes generalization error using size-normalization and centering

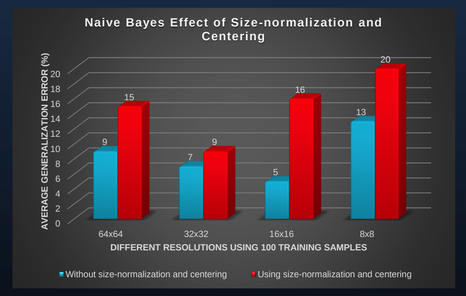

Fig. 6. Naïve Bayes effect of size-normalization and centering

Support Vector Machine (SVM) Algorithm

Support Vector Machine (SVM) algorithm was also implemented to distinguish the handwriting samples from author A and author B. For accurate comparison, we used the same setup as the Naïve Bayes algorithm, including the number of training/test samples, and the method to obtain the average generalization error.

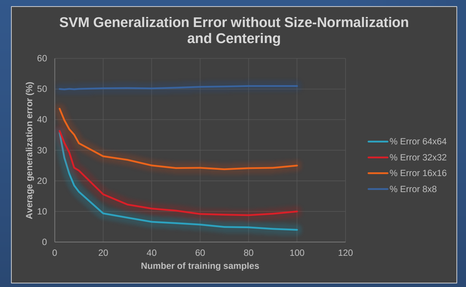

Fig. 7. SVM generalization error without size-normalization and centering

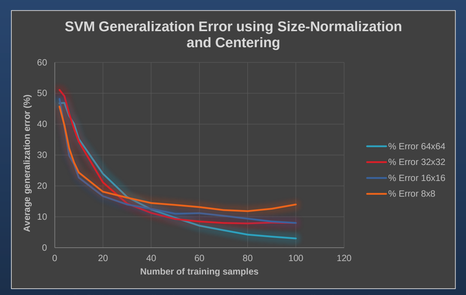

Fig. 8. SVM generalization error using size-normalization and centering

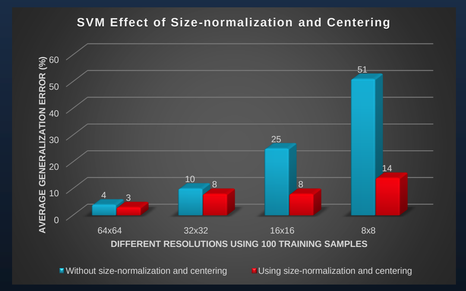

Fig. 9. SVM effect of size-normalization and centering

SVM for Three Authors

The SVM algorithm was extended to distinguish handwriting samples from all three authors A, B, and C. 50 training samples were used from each author to make it a total of 150 training samples, and also 50 test samples were used from each author to make 150 test samples.

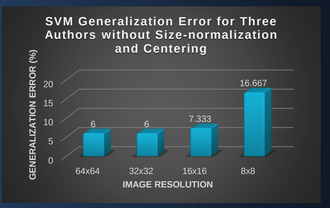

Fig. 10. SVM generalization error for three authors without size-normalization and centering

Comparison and Discussion

# of Training Samples	64x64	32x32	16x16	8x8
2				
4				
6				
8				
10				
20				
30				
40				
50				
60				
70				
80				
90				
100				

Legend
- Naïve Bayes, no size-normalization and centering
- Naïve Bayes, with size-normalization and centering
- SVM, no size-normalization and centering
- SVM, with size-normalization and centering

Fig. 11. Comparison of which algorithm performs the best at different training samples and image resolutions

Naïve Bayes without size-normalization and centering
The 16x16 images resulted in the lowest generalization error of 5%, followed by 32x32 (7%), 64x64 (9%), 8x8 (13%) images. Using 100 training samples had the lowest generalization error for all image resolutions. Refer to Fig. 4.

Naïve Bayes using size-normalization and centering
The results were worse than without using size-normalization and centering for all image resolutions and all training samples. Size-normalization and centering did not improve the performance of Naïve Bayes. Refer to Fig. 5. and Fig. 6.

SVM without size-normalization and centering
The 64x64 images had the lowest generalization error of 4%, followed by 32x32 (10%), 16x16 (25%), 8x8 (51%). The 64x64 images had the lowest generalization error at 100 training samples, but the 32x32 and 16x16 images had results flattening between 60 to 100 training samples. For the 16x16 and 8x8 images, the generalization error was too high and it was basically useless. Refer to Fig. 7.

SVM using size-normalization and centering
The performance was improved, especially for the low resolution images (16x16 and 8x8). The 64x64 and 32x32 images showed slight improvement as well. Refer to Fig. 8. and Fig. 9.

Overall
To get the best performance, for 64x64 images with very few training samples (< 10), or for 32x32, 16x16 or 8x8 images with any number of training samples, use Naïve Bayes without size-normalization and centering. For 64x64 images with 10 or more training samples, use SVM.