Classification of Hand-Written Numeric Digits

Nyssa Aragon, William Lane, Fan Zhang

December 12, 2013

1 Objective

The specific hand-written recognition application that this project is emphasizing is reading numeric digits written on a tablet or mobile device. We were interested in two goals. First of all, we wanted to build a model that can classify a hand-written digit on a tablet with high accuracy. Second of all, we were interested in how training on a specific set of users will improve the predictions of digits written by those users.

2 Data

The primary dataset used is the Pen-Based Recognition of Handwritten Digits Data Set from the UCI Machine Learning Repository. [1] This dataset is produced by collecting roughly 250 digit samples each from 44 writers, written on a pressure sensitive tablet that sent the location of the pen at fixed time intervals of 100 milliseconds. [2] Each digit is written inside a 500 x 500 pixel box, and then was scaled to an integer value between 0 and 100 to create consistent scaling between each observation. Spatial resampling is used to obtain 8 regularly spaced points on an arc trajectory.

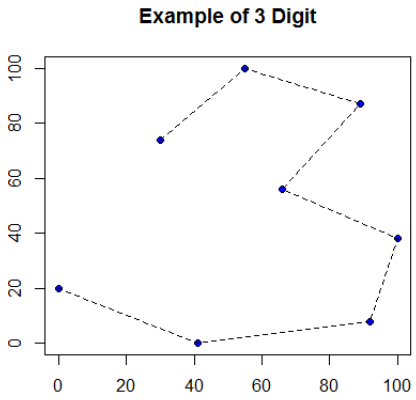
The data can be visualized by plotting the 8 sampled points based on their (x, y) coordinates, along with lines from point to point. Because the tablet sensed the location of the pen throughout time, the order of the points gives the direction that the pen was moving.

The data set contains the x and y coordinates of each of the 8 points, for a total of 16 integer variables ranging from 0 to 100. There are 7494 test data points and 3498 training data points.

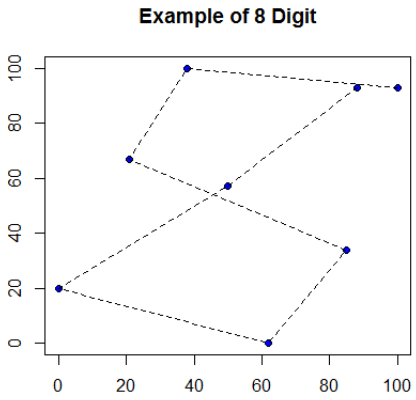
Table 1: Number of Digits Used

	'0'	'1'	'2'	'3'	'4'	'5'	'6'	'7'	'8'	'9'	Total
Training	780	779	780	719	780	720	720	778	718	719	7493
Test	363	364	364	336	364	335	336	364	335	336	3497

The training set consists of samples written by 30 writers and the testing set consists of samples written by 14 writers. The models were evaluated based on the performance on the testing dataset after being trained on the training dataset. We also explore the difference between errors when the training and



(a) Sample Points w/ Ordered Line Segments ('3')



(b) Sample Points w/ Ordered Line Segments ('8')

Figure 1

testing sets are sourced from separate writers and when the training and testing sets have some writers in common.

3 Model

After using various supervised learning techniques, the optimized error for each approach is shown in the table.

Table 2: Error Rates Over Diverse Classifiers

	SVM	KNN	Random Forest	Softmax	K-Means
Test Error	0.0172	0.0215	0.0349	0.18	0.25

The Support Vector Machine and K-Nearest Neighbor models perform the best. The data, which consists of 8 (x, y) coordinates is perhaps not surprisingly most responsive to Euclidean-distance based classifiers.

The SVM model is implemented using the *'e1071'* package in R [3]. The 'one-against-one' approach is used to extend binary classifying SVMs to multiclass predictors, meaning a series of SVMs for each pair group of the digits are created. The class probabilities for each test observation are computed using quadratic optimization. The digit with the highest probability is chosen as the prediction. Following the idea that a prediction made with low probability may be a weaker prediction, the data is separated into two groups: one group has a predicted digit with probability above the threshold of 0.525, another

with a predicted digit with a probability under the threshold.

Table 3: Error Rates Grouped By Probability Threshold

	SVM Test Error	KNN Test Error
SVM Pr < 0.525	0.4722	0.2777
SVM Pr ≥ 0.525	0.0124	0.0194

3.1 Primary Model

The main model capitalizes on this relationship. Predictions based on the SVM model is used where the probability of the prediction is above the threshold, while predictions based on the KNN model is used where the probability of the SVM prediction is below the threshold. The value of the threshold and the cost parameter for the SVM regularization is chosen based on which values give the lowest error, with the threshold at 0.525 and the cost parameter at 3.75.

The model combining SVM and KNN models gives a final test error rate of 0.0152. Error rates in range of 1-2 % are considered competitive for this dataset [5].

3.2 Sampling From the Whole Dataset

The previous models were evaluated based on the error classifying the test set of 14 writers, after being trained on the training set of 30 independent writers. Because an individual may exhibit unique patterns when writing a digit, It is supposed that the correlation between two samples of a specific digit written by one individual is higher than the correlation between two samples of a specific digit written by two separate individuals. One of our motivations was to learn how much continuous learning on a tablet would improve the predictions. In order to gain insight into this problem, we remove samples from the test data and augment them into the training set, then find the error on classifying the remaining test data. In order to train on a representative sample of the test writer's digits, we randomly selected a portion of the test data to migrate to the training set, increasing this proportion by 5% each time until we are training on half of the test data and predicting on the other half. This simulates a situation where specific users gradually feed the learning system a number of their own inputs with the correct labels. The results show that as the model receives more and more input from the set of test users, the model performs better and better with a minimum error rate of just under 1%. On the whole, SVM still outperforms KNN.

Additionally, we compared the error based on training and testing data come from separate sources to the error based on a 3-fold cross-validation. Using the primary model found in section 4, the error is found using a 3-fold cross-validation where the three sets are randomly sampled from the combined test and training datasets. A 3-fold cross-validation gives test and training sets of approximately the same size as the original sets. When the training and testing sets do not come from separate writing sources, the error reduces from 1.52% to 0.56%.

Figure 2: Error (Percent Misclassified) by Test Users Input

Using the same model that produced the optimal error of the isolated test dataset, the cross-validation error is 0.0056. When the training and testing sets do not come from separate writing sources, the error reduces from 1.52% to 0.56%.

4 Conclusion

The supervised learning techniques that performed best with this data were Support Vector Machines and K-Nearest Neighbor. Our final model was one that used SVM predictions when the SVM was confident and KNN predictions when the SVM was less confident. The models performed better when it was trained on samples from the same writers who wrote the test samples, indicating that a model that continues to learn to a specific tablet-user will better predict that user's writing.

References

[1] Bache, K. & Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[2] I. S. Dhillon, D. S. Modha, and W. S. Spangler, "Class Visualization of High-Dimensional Data with Applications," August 1999, pg. 18.

[3] David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel and Friedrich Leisch (2012). e1071: Misc Functions of the Department of Statistics (e1071), TU Wien. R package version 1.6-1. http://CRAN.R-project.org/package=e1071

[4] Venables, W. N. & Ripley, B. D. (2002) Modern Applied Statistics with S. Fourth Edition. Springer, New York. ISBN 0-387-95457-0

[5] Keysers et al. "Deformation Models for Image Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 29, No. 8. August 2007, pg. 1430.