OPTICAL CHARACTER RECOGNITION

Yusuf AKSOY

Abstract- ***Real life handwritten characters are different from printed characters because every person has a unique writing style and recognizing them is a challenging problem for computers. Optical Character Recognition (OCR) solves this type of problems. SVM and KNN methods applied to handwritten Optical Character Recognition problem. Before applying these methods, dataset processed with different techniques such as Scaling and Principal Component Analysis. They reduced features while retaining features that have more impact on the shape of letters. Dataset trained and tested with K-Fold Cross Validation method which is an effective way to work with the dataset and helps to reduce bias for training. Test results show that for specified configurations in the project SVM has better accuracy compared to KNN counterpart.***

I. INTRODUCTION

Optical Character Recognition, abbreviated as OCR, is a field that focuses on identifying printed or handwritten characters. It covers many problems encountered in real life such as recognition of license plates, paper documents, pictures or handwritten letters. It is challenging for computers to recognize characters like a human. Over the years in machine learning and computer vision, many methods applied to these problems.

Aim of this project is to classify handwritten letters, the dataset is taken from http://ai.stanford.edu/~btaskar/ocr/. There are 26 distinct lowercase letter type from ‘a’ to ‘z’ (full list is: 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'). Also 52152 characters and their 16x8 resolution grayscale images exists in the dataset. Images are given as binary pixel vectors in the dataset, for example “p_4_2” value represents pixel in row 4, column 2. Therefore a letter has 16x8=128 many features as a vector in dataset. Some of letters are visualized from dataset in **Figure 1**. Also **Figure 2** shows number of each letter in the dataset.

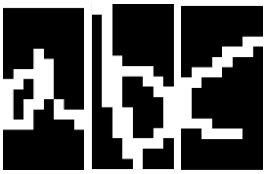

Figure 1: Visualization of ‘a’, ‘b’ and ‘c’ letters from dataset respectively

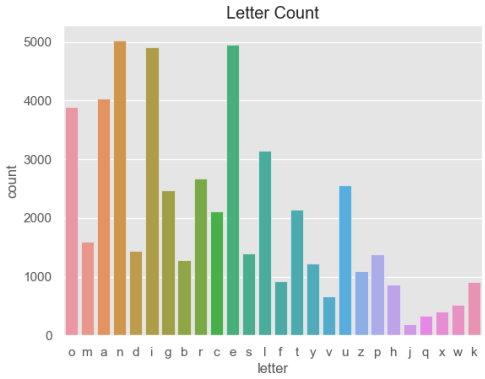

Figure 2: Letter distribution in the dataset

Project has a flow to solve problem. There are five steps in project flow shown in **Figure 3**.

1. Reading and extracting necessary data from dataset. Label dataset columns.
2. Preprocess data (Scaling, PCA).
3. Train model with a selected approach using folds (SVM and KNN fitting).
4. Test data with the trained model.
5. Evaluate data, plot or print results.

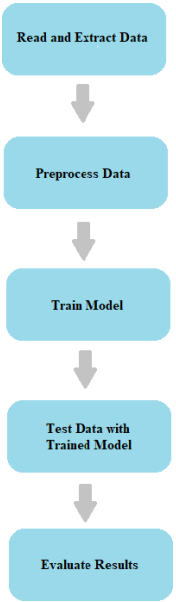

Figure 3: Project Flow

2. Preprocessing

2.1. K- Cross Validation

Cross-validation is a statistical method which reduces bias and gives better intuition for a limited amount of data. Usually, dataset is dividing into different subsets and test/training is evaluating according to them. There are different cross-validation techniques and K-Fold Cross Validation is one of them. In this technique dataset is divided into K many subsets, K is chosen as 10 for this project.
There are 10 many training/test iterations and each time one of the fold has chosen for test. Rest of the folds are trained to build a model and then this model tested on chosen test fold. Test fold is changed every iteration, therefore every fold tested.
Dataset separated according to its "fold" label. **Figure 4** shows the number of letters in each fold.

Fold Number	Number of Letters
0	4617
1	5375
2	5110
3	5353
4	5270
5	5001
6	5583
7	5370
8	5331
9	5142

Figure 4: Dataset divided into 10 folds. Figure shows the number of letters for each fold.

2.2. Scaling and Principal Component Analysis (PCA)

Each letter has 128 pixels and they can be considered features. Not all features have a significant impact on the shape of a letter, therefore features need to be reduced according to their impact. High number of features slow down training time and trained model prone to overfitting. Features need to be reduced but this shouldn't cause loss of data.

There is a method called Principal Component Analysis (PCA) which is applied to reduce features while protecting most of the data which have an impact on the shape of letter. PCA tends to hold variance at maximum while reducing features. **[1]**

Table 1 shows the test results of different percentage of retained variance in the features. It means PCA choose the minimum number of principal component such that specified percentage of variance will be retained. Initially each letter data have 128 features but PCA reduces them. Percentages 75, 85 and 95 have tested by test accuracy and training time aspects. PCA percentage input 85% performs slightly better than others and trained reasonable amount of time. **Table 1** shows that increasing percentage of PCA variance also increases training time.

PCA Variance Percentage	Overall Test Accuracy	Overall Training Time	Number of Retained Features
75%	90.97%	40.77 seconds	46
85%	91.05%	64.34 seconds	64
95%	90.91%	113.32 seconds	93

Table 1: Different PCA conservation of variance percentage results with test accuracy and training time aspect for overall of folds.

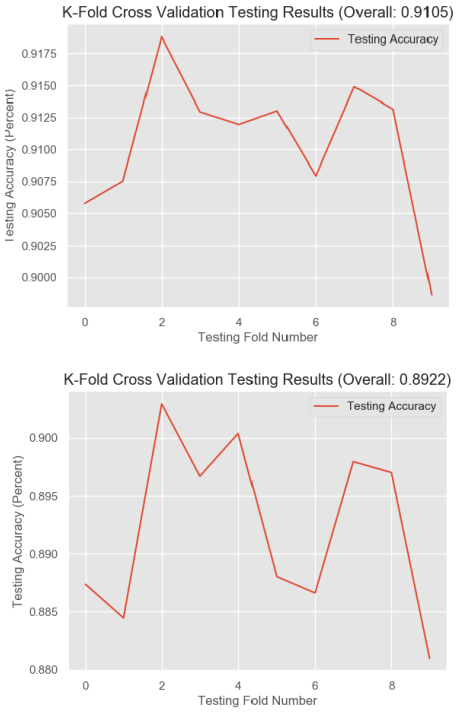

Figure 5: Top graph shows test results with scaling and then applying PCA. Graph on the bottom shows test results with applied PCA only to raw features. For both graphs, same SVM algorithm executed and PCA adjusted to preserve 85% of variance.

Features are scaled before applying PCA because it standardizes features and gives a fair distribution of variance. Scaling applied with subtracting mean value from feature and dividing result by standard deviation.
Figure 5 shows the test results of models with and without scaling features before applying PCA. For both method same SVM algorithm executed and PCA adjusted to conserve 85% of variance. PCA conserved 93 many features with scaling and PCA, on the other hand, it conserved 90 features with only PCA. Overall test accuracy of scaled and PCA applied features' test result is 91.05% while only PCA applied one is

89.22%. Applied SVM classifier has following parameters: γ = 0.01, C = 10 and kernel = "Radial Basis Function".

2.3. Label Encoding

For training and test steps letters are transformed to numerical values representation with Label Encoding. It basically turns character data ('a', 'b', 'c' etc.) to numerical data (0, 1, 2 etc.). It makes easier to represent target with numerical values.

3. Training and Testing Data

3.1. Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised learning algorithm using for classification of letters in the project. It is effective for high dimensional data. It draws decision boundary every iteration and uses kernels. It classifies well optical digits and it can be using in this project as well [**2**].

Steps of SVM training-test are:

- Preprocess data (Scaling, PCA and encode letters)
- Perform Grid Search to find optimal C (regularization) and γ (gamma) values.
- Use optimal parameters to train model with K-Fold Cross Validation.
- Test each fold.

There are tuning parameters regularization parameter (C), γ and kernel type that have a significant effect on SVM decisions. Regularization parameter C defines penalty of misclassified data while γ is non linear hyperplane parameter and tries to fit training data. C value changes decision boundary for every misclassified data and high values of C penalizes decision boundary. On the other hand higher the γ value means high sensitive for individual data points. Kernel is selected as Radial Basis Function for SVM since it performs well on non-linear data **[3][4].**

For parameter tuning Grid Search performed for C and γ values on most successful fold number 2. It executed only on one fold which is fold number 2, reason is that it takes too much time to find optimum parameter. Grid search basically tries different combinations of C and γ parameters and compares results between them. It shows result of most accurate parameter adjustment. For SVM classifier, grid search executed with C = [1, 10, 100], γ = [0.001, 0.01, 0.1] values and Radial Basis Function (RBF) kernel. Also, it cross validated parameters itself by dividing input fold into three folds internally. It found optimum parameters as C = 10 and γ = 0.01. **Figure 6** shows test results of folds with optimum SVM parameters (also PCA adjusted to conserve 85% of features and they are scaled before PCA). It performed with 91% accuracy. Although training time in **Figure 6** cannot be considered accurate and varies each training since it depends on training computer environment, it may give intuition for comparing other figures. **Figure 7** shows training-testing accuracy. It is observed that training model is general enough to avoid overfitting.

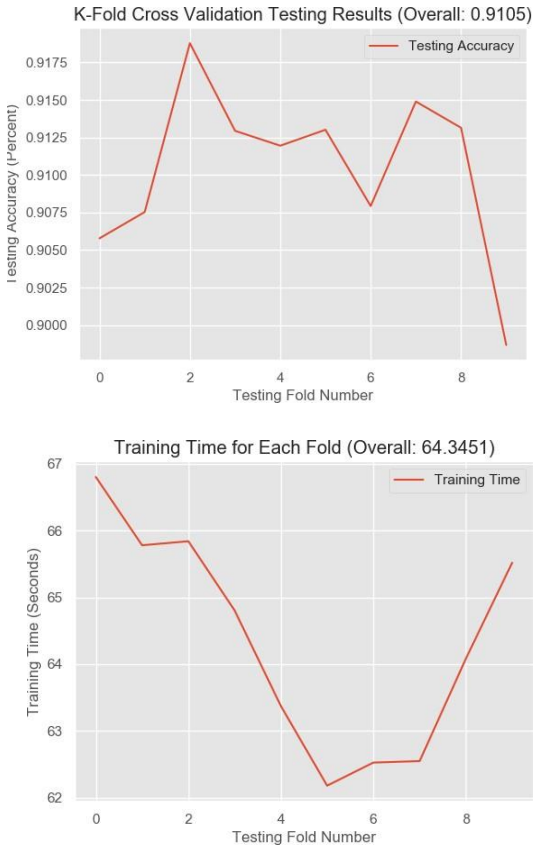

Figure 6: Test results and training time with parameters: PCA Variance Percent: 85%, SVM Parameters: C = 10, γ = 0.01, Kernel = RBF

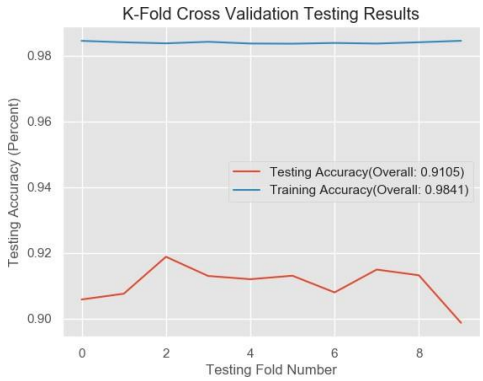

Figure 7: Accuracy of training and test of SVM

3.2. K- Nearest Neighbors (K-NN)

K-Nearest Neighbors method is a supervised learning algorithm and using for classification. It takes new unclassified data and calculates the distance between unclassified point and its closest classified "k" many neighbors. After that, it compares calculations and checks the classes that had smaller distances. Unclassified data becomes classified according to most appeared close neighbors. It is using for classification problems in OCR area such as handwritten digit recognition [**5**].

Steps of K-NN training-test are:

- Preprocess data (Scaling, PCA and encode letters)
- Train models with different numbers of k values to find optimal parameters using KD Tree. Then cross validate to observe k values for each fold.
- Use optimal parameters to train model with K-Fold Cross Validation.
- Test each fold.

Number of nearest neighbors parameter 'k' is varies between 1 to 5 and trained for each fold. To measure distance between neighbors KD Tree algorithm preferred. Training with different number of neighbors shows optimum neighbor number parameter is 3. **Figure 9** shows different neighbor numbers for some folds.

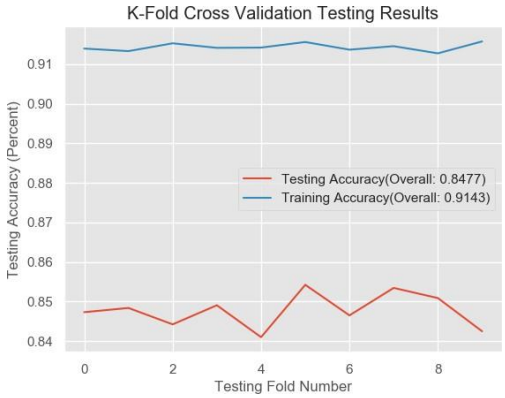

Figure 8: Overall test and training result with using KNN

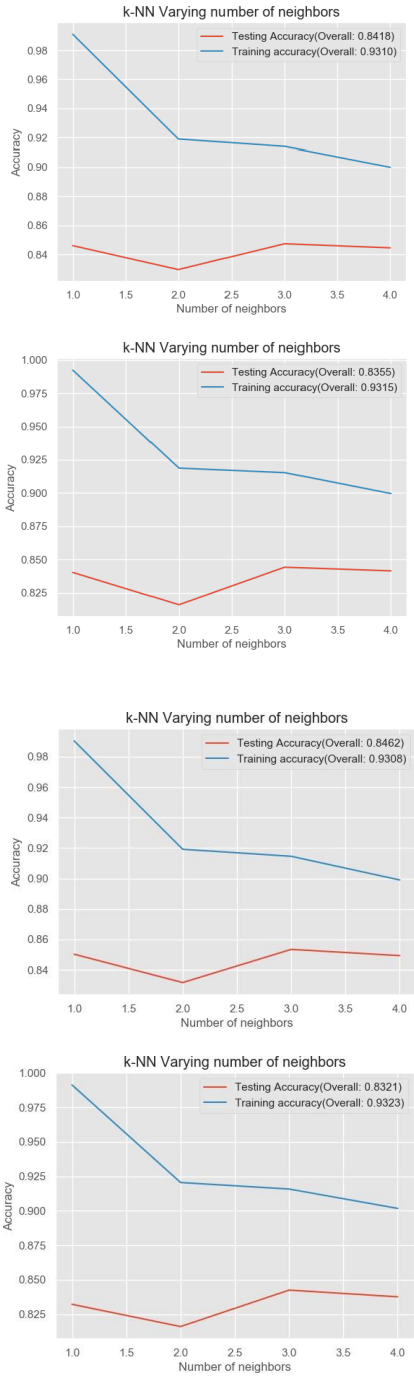

Figure 9: Varying number of nearest neighbors from 1 to 5, test and training results for some folds
(upper left: fold 0, upper right: fold 2, lower left: fold 7, lower right: fold 9)

After experimenting on number of neighbors **Figure 9** shows that three nearest neighbor parameter is more accurate than others. **Figure 8** shows test and training accuracy with k=3 and KD Tree distance measurement algorithm. Overall test score is 84.77% and it is lower than SVM counterpart.

4. Conclusion

In this project, several methods have experimented on OCR dataset and compared each other. Two major methods, SVM and KNN, applied to train data after same preprocessing processes. K Fold Cross Validation method is preferred for separating train/test dataset because it is an efficient way of processing dataset and also reduces bias.

Number of features in the dataset needs to be reduced because large of them slow down training time and not effective. For preprocessing method, features are scaled and then PCA applied to them. Results show that the preprocessing method succeeded and number of features reduced while most of the information in the features retained. Scaling before PCA approach slightly improved test accuracy. For future work to improve results, different preprocessing methods such as Kernel PCA may be applied since it may result better for non-linear data [**1**].

SVM method is optimized with Grid Search parameter tuning method and optimum parameters are found. Overall accuracy is 91.05% for optimum parameters. For KNN method the optimum number of neighbors parameter is found as 3 and KNN overall accuracy is 84.77% which is lower than SVM.

Training time is important factor as well as accuracy and considered along with accuracy of the project [**5**]. Training time also projected to figures and all methods except Grid Search method (it took few hours to complete on same computation environment with others) completed quickly.

REFERENCES

[1] Bernhard Schölkopf, Alexander Smola, Klaus-Robert Müller, Kernel Principal Component Analysis. Advances In Kernel Methods – Support Vector Learning, MIT Press, 1999

[2] Corinna Cortes, Vladimir Vapnik, Support-Vector Networks, Machine Learning, 20, 273-297, 1995

[3] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, A Practical Guide to Support Vector Classification, National Taiwan University, Taipei 106, Taiwan, 2016

[4] Christopher J.C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 2, 121–167(1998)

[5] Yann LeCun, L.D. Jackel, Leon Bottou, Corinna Cortes, John S. Denker, Harris Drucker, Isabelle Guyon, Urs A. Müller, Eduard Sackinger, Patrice Simard, Vladimir Vapnik, Learning Algorithms For Classification: A Comparison On Handwritten Digit Recognition, AT&T Bell Laboratories, 2000