# International Journal of Engineering Sciences &Research Technology

(A Peer Reviewed Online Journal)
Impact Factor: 5.164

IJESRT

RESEARCHERID
THOMSON REUTERS

[Patil, *et al.*, 8(9): September, 2019]
IC™ Value: 3.00

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

# HANDWRITTEN CHARACTER RECOGNITION USING MACHINE LEARNING

**Ms. Rupali Patil[*1] & Mrs. Jyoti Sangogi[2]**
[*1&2]RMD Sinhgad School of Engineering, Warje

**DOI**: 10.5281/zenodo.3465599

## ABSTRACT

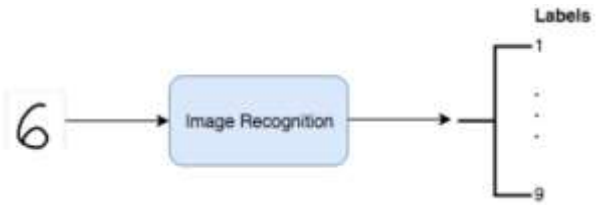
The aim of this work is to review existing methods for the handwritten digit recognition problem using machine learning algorithms and implement one of them for a user-friendly Android application. The main tasks the application provides a solution for are handwriting recognition based on touch input, handwriting recognition from live camera frames or a picture file, learning new characters, and learning interactively based on user's feedback. The recognition model we have chosen is a Support Vector Machines (SVMs) and Nearest Neighbor (NN) techniques especially because of its high performance on nonlinearly separable problems.

**KEYWORDS**: Character recognition, accuracy, SVM, NN, Image processing.

## 1. INTRODUCTION

Handwritten character recognition is a field of research in artificial intelligence, computer vision, and pattern recognition. A computer performing handwriting recognition is said to be able to acquire and detect characters in paper documents, pictures, touch-screen devices and other sources and convert them into machine-encoded form. Its application is found in optical character recognition, transcription of handwritten documents into digital documents and more advanced intelligent character recognition systems.

Handwritten character recognition can be thought of as a subset of the image recognition problem.




***The general flow of an image recognition algorithm***

Basically, the algorithm takes an image (image of a handwritten digit) as an input and outputs the likelihood that the image belongs to different classes (the machine-encoded digits, 1–9). In this blog post, I will elaborate on my approach to solving this problem with a combination of machine learning techniques.

**Problem Statement**

The problem with this project is to classify handwritten digits. The goal is to take an image of a handwritten digit and determine what that digit is. The digits range from one (1) through nine (9).
For this we will consider Support Vector Machines (SVMs) and Nearest Neighbor (NN) techniques to solve the problem. The tasks involved are the following:

1. Download the MNIST dataset
2. Preprocess the MNIST dataset
3. Train a classifier that can categorize the handwritten digits
4. Apply the model on the test set and report its accuracy

The dataset for this problem will be downloaded from kaggle, which was taken from the famous MNIST (Modified National Institute of Standards and Technology) dataset.

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

**Metrics**

We will be using the **accuracy score** to quantify the performance of our model. What percentage of test data classified correctly will be getting from accuracy. The accuracy is a good metric choice because it will be easy to compare our model's performance to that of the benchmark as it uses the same metric. Also, our dataset is balanced (equal number of training examples for each label) which makes the accuracy appropriate for this problem.

**Data Exploration**

Some initial data exploration reveals that our training set contains 42,000 samples in total and 784 features. Each sample in the dataset represent an image that is 28 pixels in height and 28 pixels in width, hence the total of 784 pixels. Each image is labelled with their corresponding category that is the actual digit from 0 to 9 for a total of 10 different labels.

***Some examples of the dataset***

## 2. ALGORITHMS AND TECHNIQUES

It has been shown that Support Vector Machines (SVMs) can be applied to image and hand-written character recognition [4]. SVMs are effective in high dimensional spaces, hence it makes sense to use SVMs for this study given the high dimensionality of our input space, i.e. 784 features. However, SVMs don't perform well in large datasets as the training time becomes cubic in the size of the dataset. This could be an issue as our dataset containing 42,000 samples which is quite large. To deal with this issue, we will adopt a technique proposed by a study conducted at the University of California, Berkeley, which is to train a support vector machine on the collection of nearest neighbours in a solution they called "SVM-KNN" [2]. Training an SVM on the entire data set is slow and the extension of SVM to multiple classes is not as natural as Nearest Neighbor (NN). However, in the neighbourhood of a small number of examples and a small number of classes, SVMs often perform better than other classification methods.

We use NN as an initial pruning stage and perform SVM on the smaller but more relevant set of examples that require careful discrimination. This approach reflects the way humans perform coarse categorization: when presented with an image, human observers can answer coarse queries such as presence or absence of an animal in as little as 150ms, and of course, can tell what animal it is given enough time [6]. This process of a quick categorization, followed by successive finer but slower discrimination was the inspiration behind the "SVM-KNN" technique.

**Benchmark**

To benchmark our model, we will use kaggle's benchmark model which uses a simple random forest model to make predictions on the test set. The benchmark model accuracy score is **0.93**, in other words, it is able to correctly label **93%** of the test data. We will evaluate how this model compares to our final solution. We will aim to have our final solution have an accuracy higher than the benchmark model.

**Data Preprocessing**

Since the original dimension is quite large (784 input features), the dimensionality reduction becomes necessary. First, we extract the principal components from the original data. We do this by fitting a Principle Component

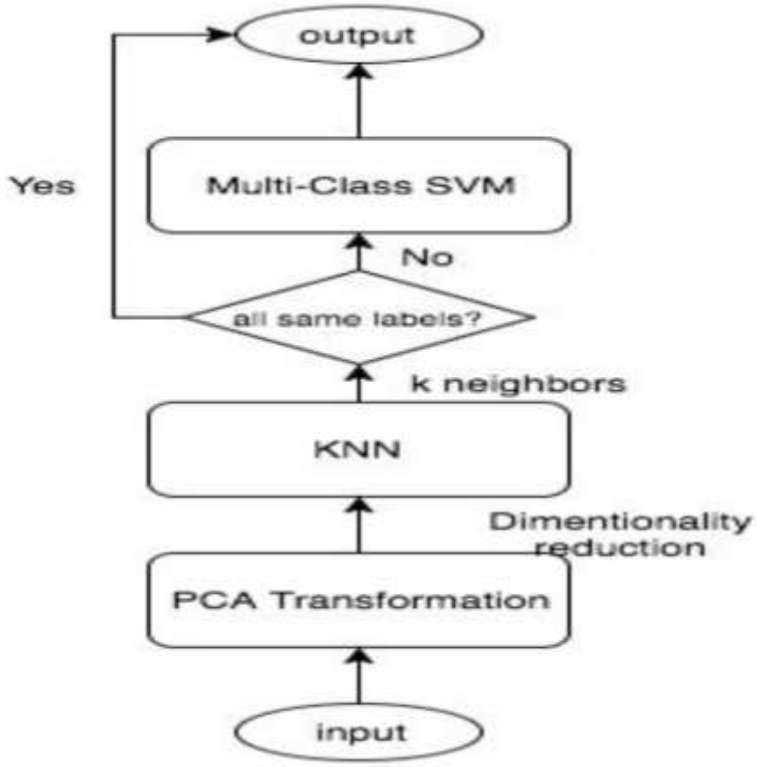
ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

Analysis (PCA) on the training set, then transforming the data using the PCA fit. We used the PCA module of the scikit-learn Python library with **n_components** set to **60** to transform the dataset. We thus choose the first 60

principal components as the extracted features.We also applied cross-validation to split the dataset into training and testing sets, retaining 40% of the data for testing. We use the StratifiedShuffleSplit module of the scikit-learn Python library passing in the label values as one of the parameters. StratifiedShuffleSplit returns train and test indices to split the data into train and test sets while preserving the percentage of the sample of each class.

**Implementation**

Our simple implementation of SVM-KNN goes as follows: for a query, we compute the Euclidean distances of the query to all training examples and pick the K nearest neighbours. If the K neighbours have all the same labels, the query is labelled and exit. Else, we compute the pairwise distances between the K neighbours, convert the distance matrix to a kernel matrix and apply multiclass SVM. We finally use the resulting classifier to label the query.




All the algorithms used in our implementation came from the scikit-learn Python library, version 0.17.1. Namely, we used PCA for dimensionality reduction, Stratified Shuffle Split for cross-validation, K Neighbors Classifier to find the k nearest neighbours and SVC to train our multi-class SVM.

## 3. CONCLUSION

The classification accuracy of **0.9714** is better than that of the benchmark (0.93514). Therefore we can conclude that our model is adequate for solving the problem of classifying handwritten characters in the MNIST dataset as it is able to accurately categorize well with an accuracy quite close to humans. However, our model is useful in a limited domain. Some changes would have to be made to solve a bigger problem of recognizing multiple digits in an image or recognizing arbitrary multi-digit text in unconstrained natural images.

## REFERENCES

[1] http://yann.lecun.com/exdb/publis/pdf/matan-90.pdf
[2] http://www.vision.caltech.edu/Image_Datasets/Caltech101/nhz_cvpr06.pdf

ISSN: 2277-9655
Impact Factor: 5.164
CODEN: IJESS7

[3] http://www.johnwinn.org/Publications/papers/WinnCriminisi_cvpr2006_video.pdf
[4] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.441.6897&rep=rep1&type=pdf
[5] https://en.wikipedia.org/wiki/Support_vector_machine#Applications
[6] https://www.ncbi.nlm.nih.gov/pubmed/8632824
[7] https://github.com/chefarov/ocr_mnist/blob/master/papers/knn_MNIST.pdf
[8] R. Plamondon et al., On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, 2000