

## **INSTITUTE OF ENGINEERING AND MANAGEMENT,KOLKATA**

**Artificial Intelligence Project (CS793C)**

**On**

**HANDWRITING ANALYSIS**

**SUBMITTED BY:**

**(CSE 4th Year , Section C)**

**PRANJAL CHOWDHURY****(UNI. ROLL.:10400116145)**

**PARTHO PROTIM SARKAR(UNI. ROLL.:** **10400116150)**

**ABSTRACT**

The aim of this work is to review existing methods for the handwritten character recognition problem using machine learning algorithms. The main tasks of the application provides a solution for handwriting recognition based on touch input, handwriting recognition from live camera frames or a picture file, learning new characters, and learning interactively basedon user's feedback.

Handwriting recognition has been one of the most fascinating and challenging research areas in field of image processing and pattern recognition in the recent years. It contributes immensely to the advancement of automation process and improves the interface between man and machine in numerous applications. The development of handwriting recognition systems began in the 1950s when there were human operators whose job was to convert data from various documents into electronic format, making the process quite long and often affected by errors.

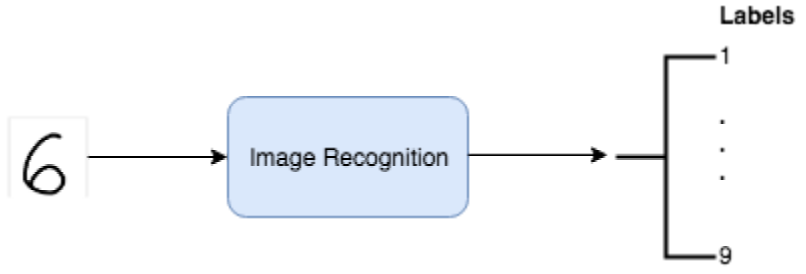
Automatic text recognition aims at limiting these errors by using image preprocessing techniques that bring increased speed and precision to the entire recognition process. Here , we develop such a tool which takes an image as an input and extract characters such as alphabets, digits, symbols from it. The image can be of handwritten document or printed document. It can be used as a form of data entry from printed records.The implementation of such a tool depends on two factors – Feature extraction and classification algorithm.

This work discusses about a method for analysing real world handwritten text samples with the aid of technology. This project is based on Machine learning, We can provide a lot of data set as an input to the software tool which will be recognized by the machine and similar pattern will be taken out from them.

**BACKGROUND**

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 and more advanced intelligent character recognition systems. Most of these systems nowadays implement machine learning mechanisms such as neural networks. Machine learning is a branch of artificial intelligence inspired by psychology and biology that deals with learning from a set of data and can be applied to solve wide spectrum of problems. A supervised machine learning model is given instances of data specific to a problem domain and an answer that solves the problem for each instance. When learning is complete, the model is able not only to provide answers to the data it has learned on, but also to yet unseen data with high precision.

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).

We will look into the [Support Vector Machines](https://en.wikipedia.org/wiki/Support_vector_machine) (SVMs)  techniques to solve the problem.

We will be using the **accuracy score** to quantify the performance of our model. The accuracy will tell us what percentage of our test data was classified correctly. 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.

**Use of Database**: For pattern recognition related applications, data patterns are one of the most necessary requirements. If the data patterns for the particular recognition application is not available, then the first and foremost task in implementing the recognition system is to collect the data patterns. Data collection is one of the tedious task in most of the pattern recognition applications. The handwritten documents are collected and stored ..

The present version of the character image database consists of binary isolated character images extracted from the collected handwritten data sheets using character segmentation algorithm.

The created character image database is available on request[1](https://www.sciencedirect.com/science/article/pii/S2215098618301447" \l "fn1), and is released in the form of comma separated values (CSV) files. Three CSV files representing training, validation and testing images are available. Each row in the CSV files, represents a character image.

**SOURCECODE:**

In [1]:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn import datasets

In [2]:

digits= datasets.load\_digits()

x=digits.data

y=digits.target

In [3]:

x[1]

Out[3]:

array([ 0., 0., 0., 12., 13., 5., 0., 0., 0., 0., 0., 11., 16.,

9., 0., 0., 0., 0., 3., 15., 16., 6., 0., 0., 0., 7.,

15., 16., 16., 2., 0., 0., 0., 0., 1., 16., 16., 3., 0.,

0., 0., 0., 1., 16., 16., 6., 0., 0., 0., 0., 1., 16.,

16., 6., 0., 0., 0., 0., 0., 11., 16., 10., 0., 0.])

In [4]:

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.25,random\_state=42)

In [5]:

from sklearn import svm

clf=svm.SVC(kernel="poly",C=1,gamma=0.1)

clf.fit(x\_train,y\_train)

Out[5]:

SVC(C=1, cache\_size=200, class\_weight=None, coef0=0.0,

decision\_function\_shape='ovr', degree=3, gamma=0.1, kernel='poly',

max\_iter=-1, probability=False, random\_state=None, shrinking=True,

tol=0.001, verbose=False)

In [6]:

pred=clf.predict(x\_test)

In [8]:

from sklearn.metrics import accuracy\_score

In [9]:

accuracy\_score(pred,y\_test)

Out[9]:

0.9888888888888889

In [10]:

clf.predict(digits.data[[100]])

Out[10]:

array([4])

In [11]:

clf.predict(digits.data[[50]])

Out[11]:

array([2])

In [12]:

clf.predict(digits.data[[500]])

Out[12]:

array([8])

In [13]:

plt.imshow(digits.images[100])

plt.show()

In [14]:

plt.imshow(digits.images[50])

plt.show()

In [15]:

plt.imshow(digits.images[500])

plt.show()

In [ ]:

**ExperimentalResults**:

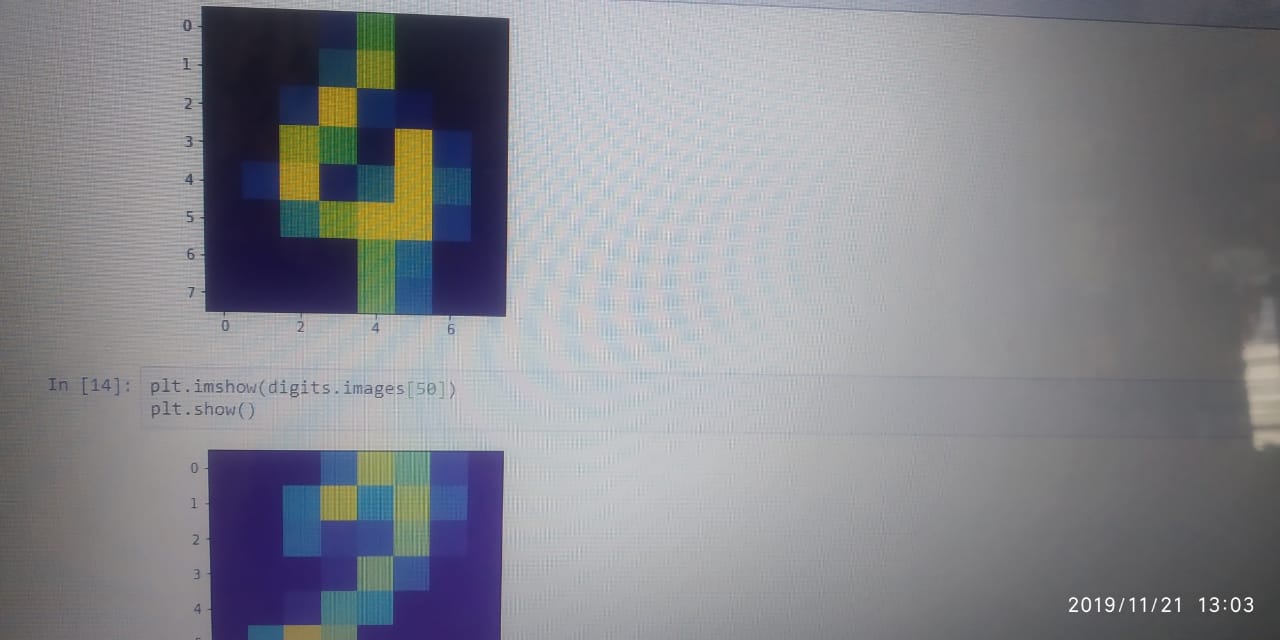
**Test application Analysis:**

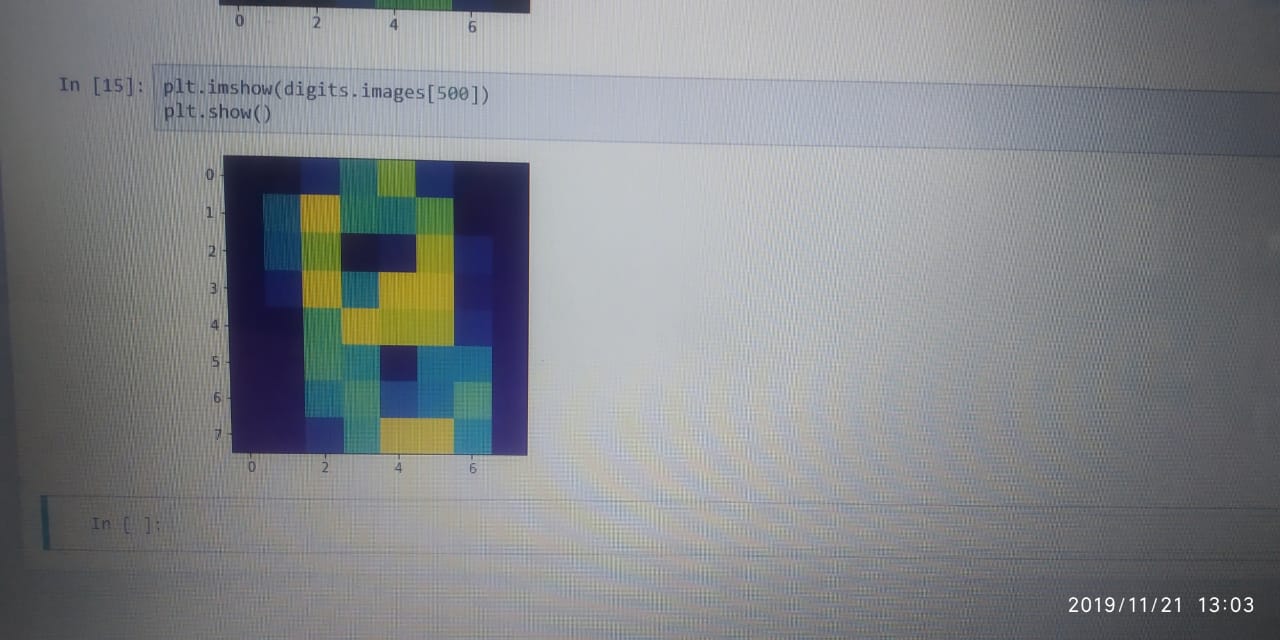
The test application accompanying the source code can perform the recognition of handwritten digits. To do so, open the application (preferably outside Visual Studio, for better performance). Click on the menu File and select Open. This will load some entries from the Optdigits dataset into the application. To perform the analysis, click the Run Analysis button. Please be aware that it may take some time. After the analysis is complete, the other tabs in the sample application will be populated with the analysis' information. The level of importance Experiments were performed on different samples having mixed scripting languages on numerals using single hidden layer.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Data set** | **Training Set Size** | **Testing Set Size** | **Validation Set Size** | **Training Set Accuracy** | **Test Set Accuracy** | **Validation Set Accuracy** |
| Digit | 1778 | 6270 | 5430 | 96 | 97 | 96 |

**Table:** Detail Recognition performance of SVM

It is observed that recognition rate using SVM is higher than other model, i.e. Hidden Markov Model. However, free parameter storage for SVM model is significantly higher. The memory space required for SVM will be the number of support vectors multiply by the number of feature values. This is significantly large compared to HMM which only need to store the weight. HMM needs less space due to the weight-sharing scheme. However, in SVM, space saving can be achieved by storing only the original online signals and the penup/ pen-down status in a compact manner. During recognition, the model will be expanded dynamically as required. SVM clearly outperforms in all three isolated character cases. The result for the isolated character cases above indicates that the recognition rate for the hybrid word recognizer could be improved by using SVM instead of HMM.

****

****

**Results:**

After the analysis has been completed and validated, we can use it to classify the new digits drawn directly in the application. We can see the analysis also performs rather well on completely new and previously unseen data.

**Figure:** Graph representation of accuracy of SVM

**CONCLUSION**

In this article, we detailed and explored how (Kernel) Support Vector Machines could be applied in the problem of handwritten digit recognition with satisfying results. The suggested approach does not suffer from the same limitations of Kernel Discriminant Analysis, and also achieves a better recognition rate. Unlike KDA, the SVM solutions are [sparse](http://en.wikipedia.org/wiki/Sparse_matrix), meaning only a generally small subset of the training set will be needed during model evaluation. This also means the complexity during the evaluation phase will be greatly reduced since it will depend only on the number of vectors retained during training.

SVM model requires the most space since each support vector (SV) consist of many feature values . However, space saving can be achieved by storing only the original online signals and the pen-up/pen-down status corresponding to the SV in a compact manner. During recognition, the model will be expanded dynamically as required.Experiments using SVMs with probabilistic output were also performed on the same datasets for comparison. In many experiments, the results have shown that at character level, SVM recognition rates are significantly better due to structural risk minimization implemented by maximizing margin of separation in the decisionfunction. However, the increase in recognition rate isnot without some impact. The number of support vectors obtained in the training characterizes SVM model size.Storing these support vectors for recognition requires larger memory as compared to NN weights since each support vector is a multidimensional feature vector. The number of support vectors can be reduced by selecting better C and gamma parameter values through a finer grid search and by reduced set selection . The comparison of recognition results of SVM with probabilistic output and SVM distance output shows that both are comparable. In some datasets, SVM distance gives slightly higher while in some others the probabilistic output gives higher recognition rates.

**FUTUREWORK**

Future works on the database includes extending the character class collection by including all the presently used valid orthographic shapes for specific language script and creating word, line and page level collection of document images so that the researchers can focus on other stages of document recognition system as well.

It has been shown that Support Vector Machines (SVMs) can be applied to image and hand-written character recognition . 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 bigger datasets dataset containing thousand of samples which is quite large. To deal with this issue, a techniquecan be proposed ,which is to train a support vector machine on the collection of nearest neighbours in a solution they called “SVM-KNN” . 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 can 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 . This process of a quick categorization, followed by successive finer but slower discrimination was the inspiration behind the “SVM-KNN” technique.