OPTICAL CHARACTER RECOGNITION

Submitted by

Rohith Saxena [RA2011026010250]
Anish Parkhe [RA2011026010285]

Under the Guidance of

Dr. U.SAKTHI

Assistant Professor, Department of Computational Intelligence

In partial satisfaction of the requirements for the degree of

BACHELORS OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING

with specialization in Artificial Intelligence & Machine Learning

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

May 2023

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
Deemed to be University u/s 3 of UGC Act, 1956

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this Course Project Report titled **"OPTICAL CHARACTER RECOGNITION"** is the bonafide work done by ROHITH SAXENA [RA2011026010250], ANISH PARKHE [RA2011026010285] who carried out under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

Sai 11/5/22

SIGNATURE
Faculty In-Charge
Dr.U.Sakthi
Assistant Professor
Department of Networking and Communications
SRM Institute of Science and Technology
Kattankulathur Campus, Chennai

HEAD OF THE DEPARTMENT
Dr. R Annie Uthra
Professor and Head
Department of Computational Intelligence,
SRM Institute of Science and Technology
Kattankulathur Campus, Chennai

INDEX

ABSTRACT

Handwritten character recognition is one of the practically important issues in pattern recognition applications. The main purpose of this project is to build an automatic handwritten digit recognition method for the recognition of handwritten digit strings.

To accomplish the recognition task, first, the digits will be segmented into individual digits. Then, a digit recognition module is employed to classify each segmented digit completing the handwritten digit string recognition task.

The applications of digit recognition include postal mail sorting, bank check processing, form data entry, etc. The heart of the problem lies within the ability to develop an efficient algorithm that can recognize handwritten digits and which is submitted by users by the way of a scanner, tablet, and other digital devices.

INTRODUCTION

2.1 Introduction

Machine learning and deep learning plays an important role in computer technology and artificial intelligence. With the use of deep learning and machine learning, human effort can be reduced in recognizing, learning, predictions and many more areas.

To make machines more intelligent, the developers are diving into machine learning and deep learning techniques. A human learns to perform a task by practicing and repeating it again and again so that it memorizes how to perform the tasks. Then the neurons in his brain automatically trigger and they can quickly perform the task they have learned. Deep learning is also very similar to this. It uses different types of neural network architectures for different types of problems.

For example – object recognition, image and sound classification, object detection, image segmentation, etc. The handwritten digit recognition is the ability of computers to recognize human handwritten digits. It is a hard task for the machine because handwritten digits are not perfect and can be made with many different flavors. The handwritten digit recognition is the solution to this problem which uses the image of a digit and recognizes the digit present in the image.

2.1.1 Digit Recognition System:

Digit recognition system is the working of a machine to train itself or recognizing the digits from different sources like emails, bank cheque, papers, images, etc. and in different real-world scenarios for online handwriting recognition on computer tablets or system, recognize number plates of , numeric entries in forms filled up by hand and so on.

2.2 Problem Statement:

The goal of this project is to create a model that will be able to recognize and determine the handwritten digits from its image by using the concepts of K Nearest Neighbours. Though the goal is to create a model which can recognize the digits, it can be extended to letters and an individual's handwriting. The major goal of the proposed system is understanding KNN, and applying it to the optical character recognition system.

LITERATURE SURVEY

An early notable attempt in the area of character recognition research is by Grimsdale in 1959. The origin of a great deal of research work in the early sixties was based on an approach known as analysisby-synthesis method suggested by Eden in 1968. The great importance of Eden's work was that he formally proved that all handwritten characters are formed by a finite number of schematic features, a point that was implicitly included in previous works. This notion was later used in all methods in syntactic (structural) approaches of character recognition.

1. K. Gaurav, Bhatia P. K. , his paper deals with the various pre-processing techniques involved in the character recognition with different kind of images ranges from a simple handwritten form based documents and documents containing colored and complex background and varied intensities.In this, different preprocessing techniques like skew detection and correction, image enhancement techniques of contrast stretching, binarization, noise removal techniques, normalization and segmentation, morphological processing techniques are discussed.

2. Sandhya Arora , used four feature extraction techniques namely, intersection, shadow feature, chain code histogram and straight line fitting features. Shadow features are computed globally for character image while intersection features, chain code histogram features and line fitting features are computed by dividing the character image into different segments. On experimentation with a dataset of 4900 samples the overall recognition rate observed was 92.80% for Devanagari characters.

3. Brakensiek, J. Rottland, A. Kosmala, J. Rigoll, in their paper a system for off-line cursive handwriting recognition is described which is based on Hidden Markov Models (HMM) using discrete and hybrid modelling techniques. Handwriting recognition experiments using a discrete and two different hybrid

approaches, which consist of a discrete and semi-continuous structures, are compared. It is found that the recognition rate performance can be improved of a hybrid modelling technique for HMMs, which depends on a neural vector quantizer (hybrid MMI), compared to discrete and hybrid HMMs, based on tired mixture structure (hybrid - TP), which may be caused by a relative small data set.

4. R. Bajaj, L. Dey, S. Chaudhari , employed three different kinds of features, namely, the density features, moment features and descriptive component features for classification of Devanagari Numerals. They proposed multi classifier connectionist architecture for increasing the recognition reliability and they obtained 89.6% accuracy for handwritten Devanagari numerals.

5. G. Pirlo and D. Impedovo in his work on , presented a new class of membership functions, which are called Fuzzymembership functions (FMFs), for zoning-based classification. These FMFs can be easily adapted to the specific characteristics of a classification problem in order to maximize classification performance. In this research, a realcoded genetic algorithm is presented to find, in a single optimization procedure, the optimal FMF, together with the optimal zoning described by Voronoi tessellation. The experimental results, which are carried out in the field of handwritten digit and character recognition, indicate that optimal FMF performs better than other membership functions based on abstract level, ranked-level, and measurement-level weighting models, which can be found in the literature.

6. Sushree Sangita Patnaik and Anup Kumar Panda May 2011, this paper proposes the implementation of particle swarm optimization (PSO) and bacterial foraging optimization (BFO) algorithms which are intended for optimal harmonic compensation by minimizing the undesirable losses occurring inside the APF itself. The efficiency and effectiveness of the implementation of two approaches are compared for two different conditions of supply. The total harmonic distortion (THD) in the source current which is a measure of APF performance is

reduced drastically to nearly 1% by employing BFO. The results demonstrate that BFO outperforms the conventional and PSO based approaches by ensuring excellent functionality of APF and quick prevail over harmonics in the source current even under unbalanced supply.

7. M. Hanmandlu, O.V. Ramana Murthy have presented in their study the recognition of handwritten Hindi and English numerals by representing them in the form of exponential membership functions which serve as a fuzzy model. The recognition is carried out by modifying the exponential membership functions fitted to the fuzzy sets. These fuzzy sets are derived from features consisting of normalized distances obtained using the Box approach. The membership function is modified by two structural parameters that are estimated by optimizing the entropy subject to the attainment of membership function to unity. The overall recognition rate is found to be 95% for Hindi numerals and 98.4% for English numerals.

8. Renata F. P. Neves have proposed SVM based offline handwritten digit recognition. Authors claim that SVM outperforms the Multilayer perceptron classifier. Experiment is carried out on NIST SD19 standard dataset. Advantage of MLP is that it is able to segment non-linearly separable classes. However, MLP can easily fall into a region of local minimum, where the training will stop assuming it has achieved an optimal point in the error surface. Another hindrance is defining the best network architecture to solve the problem, considering the number of layers and the number of perceptron in each hidden layer. Because of these disadvantages, a digit recognizer using the MLP structure may not produce the desired low error rate.

METHODOLOGIES

We used MNIST as a primary dataset to train the model, and it consists of 70,000 handwritten raster images from 250 different sources out of which 60,000 are used for training, and the rest are used for training validation. Our proposed method mainly separated into stages, preprocessing, Model Construction, Training & Validation, Model Evaluation & Prediction. Since the loading dataset is necessary for any process, all the steps come after it.

Import the libraries:

Libraries required are Keras, Tensor flow, Numpy, Pillow, Tkinkter.

1. Keras:

Keras is a powerful and easy-to-use free open source Python library for developing and evaluating deep learning models. 18 It wraps the efficient numerical computation libraries Theano and TensorFlow and allows you to define and train neural network models in just a few lines of code. It uses libraries such as Python, C#, C++ or standalone machine learning toolkits. Theano and TensorFlow are very powerful libraries but difficult to understand for creating neural networks. Keras is based on minimal structure that provides a clean and easy way to create deep learning models based on TensorFlow or Theano. Keras is designed to quickly define deep learning models. Well, Keras is an optimal choice for deep learning applications.

2. TensorFlow:

TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow. TensorFlow tutorial is designed for both beginners and professionals. Our tutorial provides all the basic and advanced concept of

machine learning and deep learning concept such as deep neural network, image processing and sentiment analysis. TensorFlow is one of the famous deep learning frameworks, developed by Google Team. It is a free and open source software library and designed in Python programming language, this tutorial is designed in such a way that we can easily implements deep learning project on TensorFlow in an easy and efficient way. Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems. It can run on single CPU systems, GPUs as well as mobile devices and largescale distributed systems of hundreds of machines.

3. Numpy:

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices. Numpy which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. It is an open source project and you can use it freely. NumPy stands for Numerical Python. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

4. Pillow:

Pillow is a free and open source library for the Python programming language that allows you to easily create &s manipulate digital images. Pillow is built on top of PIL (Python Image Library). PIL is one of the important modules for image processing in Python. However, the PIL module is not supported since

2011 and doesn’t support python 3. Pillow module gives more functionalities, runs on all major operating system and support for python 3. It supports wide variety of images such as “jpeg”, “png”, “bmp”, “gif”, “ppm”, “tiff”. You can do almost anything on digital images using pillow module. Apart from basic image processing functionality, including point operations, filtering images using built-in convolution kernels, and color space conversions.

5. Tkinkter:

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit. We need to import all the modules that we are going to need for training our model. The Keras library already contains some datasets and MNIST is one of them. So we can easily import the dataset through Keras. The mnist.load_data() method returns the training data, its labels along with the testing data and its labels.

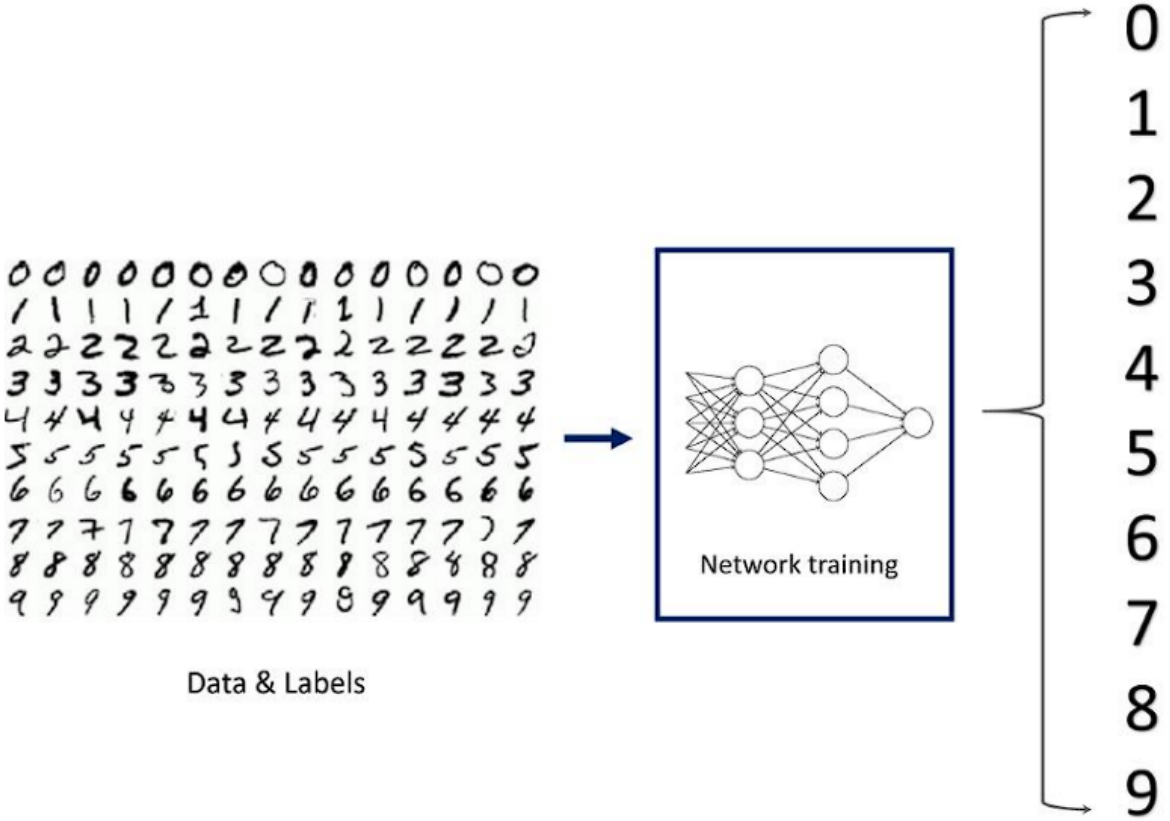
DATASET

MNIST Data Set:

Modified National Institute of Standards and Technology (MNIST) is a large set of computer vision dataset which is extensively used for training and testing different systems. It was created from the two special datasets of National Institute of Standards and Technology (NIST) which holds binary images of handwritten digits. The training set contains handwritten digits from 250 people, among them 50% training dataset was employees from the Census Bureau and the rest of it was from high school students. However, it is often attributed as the first datasets among other datasets to prove the effectiveness of the neural networks.

The database contains 60,000 images used for training as well as few of them can be used for crossvalidation purposes and 10,000 images used for testing. All the digits are grayscale and positioned in a fixed size where the intensity lies at the center of the image with 28×28 pixels. Since all the images are 28×28 pixels, it forms an array which can be flattened into 28*28=784 dimensional vector. Each

component of the vector is a binary value which describes the intensity of the pixel.



K-NN ALGORITHM:

o K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.

o K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. o K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

o K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.

o K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data. o It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

o KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

The K-NN working can be explained on the basis of the below algorithm:

o Step-1: Select the number K of the neighbours

o Step-2: Calculate the Euclidean distance of K number of Neighbours

o Step-3: Take the K nearest Neighbours as per the calculated Euclidean distance.

o Step-4: Among these k Neighbours, count the number of the data points in each category.

o Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

o Step-6: Our model is ready.

Advantages of KNN Algorithm:

o It is simple to implement.

o It is robust to the noisy training data.

o It can be more effective if the training data is large.

Disadvantages of KNN Algorithm:

o Always needs to determine the value of K which may be complex some time.

o The computation cost is high because of calculating the distance between the data points for all the training samples.

CODE

```
# import the dataset
import numpy as np
from sklearn.datasets import fetch_openml
mnist = fetch_openml('mnist_784', version=1)
# fetch the data into input features X and target y
X = mnist.data
y = mnist.target.astype(int)
# define a function to print the images
import matplotlib.pyplot as plt
def print_image(index):
  # get the desired data and reshape in the correct form
  example = X.loc[index].values
  example = example.reshape(28,28)
  # plot the image
  plt.imshow(example, cmap='GnBu')
  plt.axis('off')
# print an example image
print_image(43)
# divide into train & test sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_val_score
# create an instance of the classifier
knn_default = KNeighborsClassifier()
# fit the model and perform cross validation
knn_default = KNeighborsClassifier()
cv_scores = cross_val_score(knn_default, X_train, y_train, cv=5, scoring="accuracy")
# take the average of the scores of different folds
final_score_default = np.mean(cv_scores)
print("Accuracy with defaul KNN:", final_score_default)
```

```
parameters = {'n_neighbors':list(range(4,7)), 'weights': ['uniform', 'distance']}
from sklearn.model_selection import GridSearchCV
# create an instance of the knn classifier
knn_grid_tuned = KNeighborsClassifier()
# create an instance of grid search with the above parameters
grid_search = GridSearchCV(knn_grid_tuned, parameters, cv=2, scoring='accuracy', return_train_score=True, verbose=10)
# fit the grid search with training set
grid_search.fit(X_train, y_train)
# retrieve the best estimator
knn_tuned = grid_search.best_estimator_
from sklearn.metrics import accuracy_score
accuracy_score(y_test, knn_tuned.predict(X_test))
pred = knn_tuned.predict(X_test)
pred
```

Output:

```
Accuracy with default KNN: 0.9686666666666666
Final accuracy: 0.9719428571428571
```

CONCLUSION AND FUTURE WORK

Conclusion:

Our project OPTICAL CHARACTER RECOGNITION deals with identifying the digits. The main purpose of this project is to build an automatic handwritten digit recognition method for the recognition of handwritten digit strings. In this project, we have used KNN machine learning algorithm, which is further improved with Grid Search to increase the final accuracy.

Future Work:

The proposed system takes 28x28 pixel-sized images as input. The same system with further modifications and improvements in the dataset and the model can be used to build Handwritten Character Recognition System which recognizes human handwritten characters and predicts the output.

REFERENCES

1. https://www.tensorflow.org/learn

2. https://www.geeksforgeeks.org/python-tkinter-tutorial/

3. https://medium.com/the-andela-way/applying-machine-learning-to-recognize handwrittencharacters-babcd4b8d705

4. https://nanonets.com/blog/handwritten-character-recognition/

5. https://www.geeksforgeeks.org/python-tkinter-tutorial/

6. https://medium.com/the-andela-way/applying-machine-learning-to-recognize handwrittencharacters-babcd4b8d705

7. https://nanonets.com/blog/handwritten-character-recognition/

8. https://www.tensorflow.org/learn

9. R. Alhajj and A. Elnagar, "Multiagents to separating handwritten connected digits," in IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, vol. 35, no. 5, pp. 593-602, Sept. 2005. https://doi.org/10.1109/TSMCA.2005.843389

10. https://towardsdatascience.com/useful-plots-to-diagnose-your-neural-network-521907fa2f45

11. https://towardsdatascience.com/the-best-machine-learning-algorithm-for-handwritten-digitsrecognition-2c6089ad8f09

12. https://towardsdatascience.com/image-recognition-with-machine-learning-on-python-imageprocessing-3abe6b158e9