**HANDWRITTEN CHARACTER RECOGNITION**

**MINI PROJECT REPORT**

**18 APRIL 2022**

## INDEX

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TITLE** | **PAGE NO** |
| 1. | **ABSTRACT** | 3 |
| 2. | **PURPOSE** | 3 |
| 3. | **OBJECTIVE** | 3 |
| 4. | **INTRODUCTION** | 4 |
| 5. | **IMPLEMENTATION** | 8 |
| 6. | **CONCLUSION** | 14 |
| 7. | **REFERENCES** | 14 |

### ABSTRACT

In todays’ world advancement in sophisticated scientific techniques is pushing further the limits of human outreach in various fields of technology. One such field is the field of character recognition commonly known as HCR (Handwritten Character Recognition).

Handwritten character recognition (HCR) is achallenging learning problem in pattern recognition, mainly due to similarity in structure of characters, different handwriting styles, noisy datasets and a large variety of languages and scripts.

This project, ‘Handwritten Character Recognition’ is a software algorithm project to recognize any hand written character that is, English alphabets from A-Z, efficiently on computer with input is either an old optical image or currently provided through touch input, mouse or pen.

### PURPOSE OF THE PROJECT

Advantage of HCR systems is that it can reduce the data entry time, storage space required by documents. Fast retrieval is an alternative advantage.

HCR can be used in diverse fields like banking field where checks can be processed without human interruption and to digitize paper documents in legal industries.

### OBJECTIVE OF THE PROJECT

The objective of this project is to identify handwritten characters with the use of neural networks. We have to construct suitable neural network and train it properly. The program should be able to extract the characters one by one and map the target output for training purpose. After automatic processing of the image, the training dataset has to be used to train “classification engine” for recognition purpose.

### Project Prerequisites

Below are the prerequisites for this project:

1. Python (3.7.4 used)
2. IDE (Jupyter used)

Required frameworks are

1. Numpy (version 1.16.5)
2. cv2 (openCV) (version 3.4.2)
3. Keras (version 2.3.1)
4. Tensorflow (Keras uses TensorFlow in backend and for some image preprocessing) (version 2.0.0)
5. Matplotlib (version 3.1.1)
6. Pandas (version 0.25.1)

### Dataset Details

The dataset contains 26 folders (A-Z) containing handwritten images in size 2828 pixels, each alphabet in the image is centre fitted to 2020-pixel box. All present in the form of a CSV file.

Each image is stored as Gray-level

[**Handwritten character recognition dataset**](https://www.kaggle.com/sachinpatel21/az-handwritten-alphabets-in-csv-format)

### INTRODUCTION

This project, ‘Handwritten Character Recognition’ is a software algorithm project to recognize any hand written character that is, English alphabets from A-Z, efficiently on computer with input is either an old optical image or currently provided through touch input, mouse or pen.

This we are going to achieve by modelling a neural network that will have to be trained over a dataset containing images of alphabets.

The handwritten character recognition is the capability of computer applications to recognize the human handwritten characters. It is a hard task for the machine because handwritten characters are not perfect and can be made with many different shapes and sizes. The handwritten character recognitionsystem is a way to tackle this problem which uses the image of a character and recognizes the character present in the image.

The idea is to device efficient algorithms which get input in digital image format. After that it processes the image for better comparison. Then after the processed image is compared with already available set of font images. The last step gives a prediction of the character in percentage accuracy.

### APPROACH

The computation code is divided into the next categories:

* Pre-processing of the image
* Feature extraction
* Creating an Artificial Neural Network
* Training & Testing of the network
* Recognition

### CNN (Convolutional Neural Networks)

CNN stands for Convolutional Neural Networks that are used to extract the features of the images using several layers of filters.

A CNN consists of a lot of layers. These layers when used repeatedly, lead to a formation of a Deep Neural Network. The fundamental types of layers used to build a CNN are:

1. **Input:** This layer holds the uncooked pixel values of photograph and convert it to grayscale pics using 28x28 matrix of pixels.
2. **Convolutional Layer:** This layer gets the effects of the neuron layer that is linked to the enter regions. The wide variety of filters to be used in this layer is described here. Each filter may additionally be a 5x5 window that slider over the input records and receives the pixel with the most intensity as the output.
3. **Rectified Linear Unit (ReLU) Layer:** This layer applies a thing smart activation function on the picture records and makes use of again propagation techniques. ReLU function is utilized in order to preserve the equal values of the pixels and not being changed by means of the returned propagation.
4. **Pooling Layer:** Down-sampling operation along the spatial dimensions (width, height), resulting in volume is utilized in this layer.
5. **Fully Connected Layer:** This layer is used to compute the score instructions that potential which class has the maximum score corresponding to the enter digits. The category label with the largest likelihood is chosen as the ultimate classification from the network and proven in the output.

### Libraries Used

#### TensorFlow

TensorFlow is used as backend in the application of this project. TensorFlow is a brilliant records circulation in the Machine Learning Library made by means of the Google Brain Team and made open supply in 2015. It is designed to ease the use and greatly relevant to each numeric and neural gadget troubles simply like different spaces. TensorFlow is essentially a low-level math-entangled tool that pursuits experts who apprehend what they’re doing to construct exploratory studying structures, play around with them, and turn them into running programs. For the most part, it can be considered as a programming context in which equations can be entitled as graphs. Math things to do are spoken by using nodes in the graph, and the edges include the multidimensional facts clusters (tensors) linked to them.

#### Python

Python is used for the duration of the implementation of assignment the place several traces of code had been brought in order to accomplish the assignment requirements. Python is typically used globally, and is a high-level programming language. It was once implemented in particular for application dominance, and its language shape allows software program engineers to carry thoughts in fewer traces of code. Python is a programming language which gives the chance to work shortly and more correctly organize frameworks.

#### Anaconda

Anaconda is used as IDE all through the implementation of the project.

Anaconda is a free and open-source appropriation of the Python and R programming for logical figuring such as statistics science, AI applications, instruction of large-scale information, prescient investigation, etc. Anaconda accompanies over 1,400 programs just like the Conda package and digital surroundings director, Anaconda Navigator, so it takes the want to determine out how each library can be freely added.

#### Keras

Keras is used to build model to arrange the layers in the course of the implementation of this project. Keras is a high-level neural community API written in Python that can run on top of TensorFlow, CNTK, or Theano. It used to be developed with a focal point on allowing for quickly experimentation. The key to doing true lookup is being capable to go from notion to result with the least delay viable.

Keras approves for handy and speedy prototyping (through person friendliness, modularity, and extensibility). Similarly, it supports each convolutional networks and recurrent networks, as properly as combinations of the two and runs seamlessly on CPU and GPU.

#### NumPy

NumPy is used for mathematical calculations to print out the predict records in this project. NumPy is the core bundle with Python for scientific computing. It is a versatile sophisticated (broadcasting) with N-dimensional array object characteristic software program for combining C / C++ and Fortran code, advantageous linear algebra, Fourier transform, and random number capabilities.

#### Matplotlib

Matplotlib is used to plot model accuracy and loss in a graphical view in this project. Matplotlib is a Python 2D plotting library that produces pleasant figures for the publication throughout platforms in a variety of hardcopy formats and interactive environments. Matplotlib can be used in Python scripts, Python and IPython shells, Jupyter notebook, Web software servers, and four interface toolkits for graphical users.

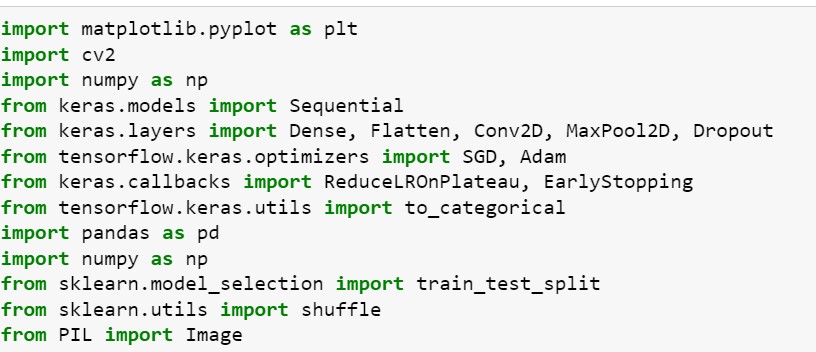
### PANDAS

Pandas is an open-source library that is made mainly for working with relational or labelled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. This library is built on top of the NumPy library. Pandas is fast and it has high performance & productivity for users.

#### IMPLEMENTATION

**1. Importing Required libraries**

We will import libraries that we have installed in our system, whenever we require them.



##### 2. Read Dataset

Now we will read our dataset that is a csv file using the pandas read\_csv() method and printing the first 10 images using **data.head(10)**. We will read the data as data type float32 as the csv file is very large and it will be better to read it as float



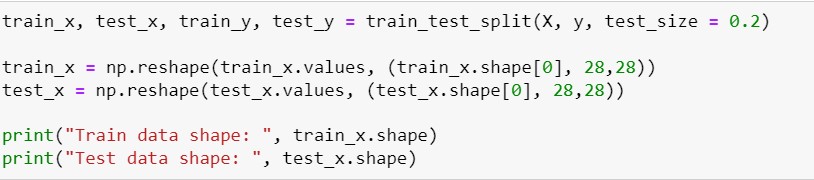
##### 3. Splitting of Data

Splitting the data read into the images & their corresponding labels. The ‘0’ contains the labels, & so we drop the ‘0’ column from the data dataframe read & use it in the y to form the labels.



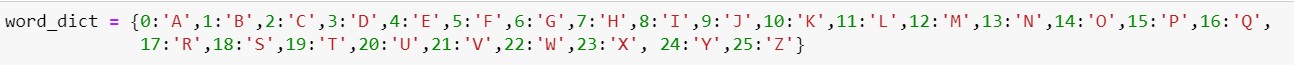
##### 4. Reshaping the Training and Testing data

In this we reshaped our training and testing image data using numpy, so that we can display it in the form of image, as initially in the csv file they are present in 784 columns of pixel data. So, we convert them to 28×28 pixels.



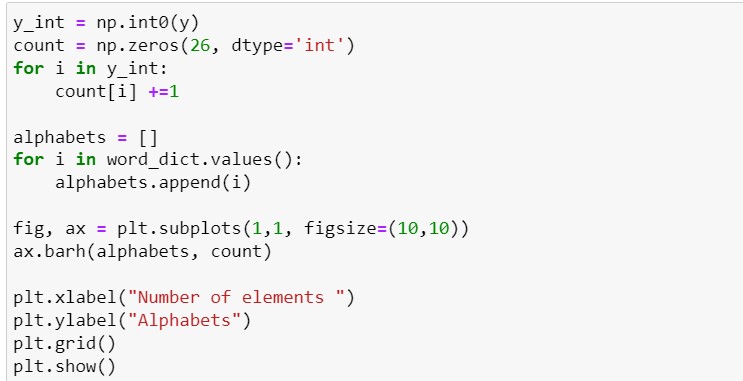
##### 5. Creating Words Dictionary

All the labels are present in the form of floating-point values, that we convert to integer values, & so we create a dictionary word\_dict to map the integer values with the characters.



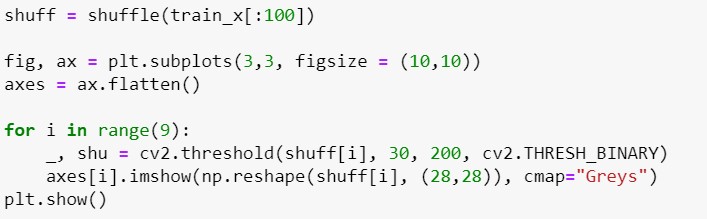
##### 6. Plotting the number of alphabets in the dataset

* Here we are only describing the distribution of the alphabets.
* Firstly, we convert the labels into integer values and append into the count list according to the label. This count list has the number of images present in the dataset belonging to each alphabet.
* Now we create a list – alphabets containing all the characters using the values() function of the dictionary.
* Now using the count & alphabets lists we draw the horizontal bar plot.



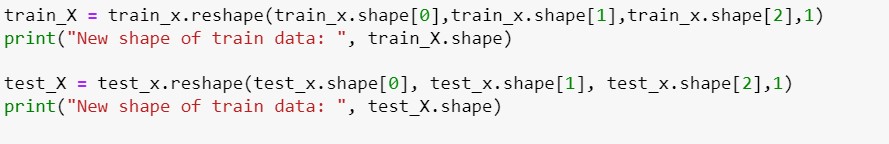
##### 7. Shuffling the data

* Now we shuffle some of the images of the train set.
* The shuffling is done using the shuffle() function so that we can display some random images.
* We then create 9 plots in 3×3 shape & display the thresholded images of 9 alphabets.

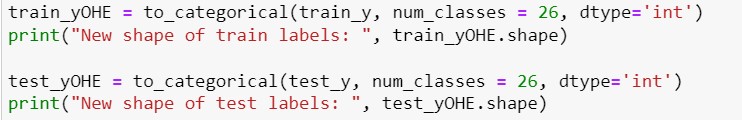


##### 8. Reshaping the training & test dataset so that it can be put in the model

Now we reshape the train & test image dataset so that they can be put in the model.



Here we convert the single float values to categorical values. This is done as the CNN model takes input of labels & generates the output as a vector of probabilities.



##### 9. Model Creation

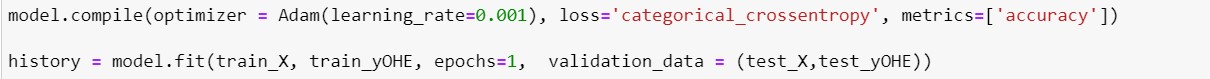
We will be creating a Convolutional Neural Networks (CNN) model which is very popular while classifying images as it extracts the features of images using several hidden layers or we say several layers of filters.

* 3 convolutional layers(Conv2D) of 64,64,64 layers each, followed by MaxPool layers that are used to reduce the number of features extracted.
* And after flatten the layers, we created two fully connected layers (Dense layer) of 128,265 layers respectively.
* And at last we have created our output layer that is also a fully connected layer with softmax as an activation function.



##### 10. Compiling & Fitting Model

* Here we are compiling the model, where we define the optimizing function & the loss function to be used for fitting.
* The optimizing function used is Adam, that is a combination of RMSprop & Adagram optimizing algorithms.
* The dataset is very large so we are training for only a single epoch, however, as required we can even train it for multiple epochs (which is recommended for character recognition for better accuracy).

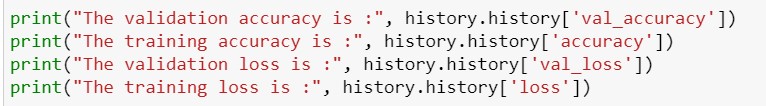


Now we are getting the model summary that tells us what were the different layers defined in the model & also we save the model using **model.save()** function.



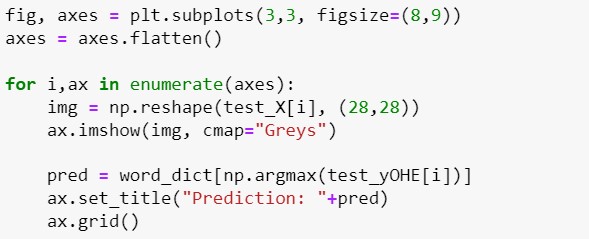
##### 11. Getting the Train & Validation Accuracies & Losses

In this we print out the training & validation accuracies along with the training & validation losses for character recognition.



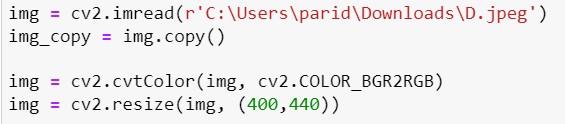
##### 12. Predictions on Test Data

Here we are creating 9 subplots of (3,3) shape & visualize some of the test dataset alphabets along with their predictions, that are made using the **model.predict()** function for text recognition.

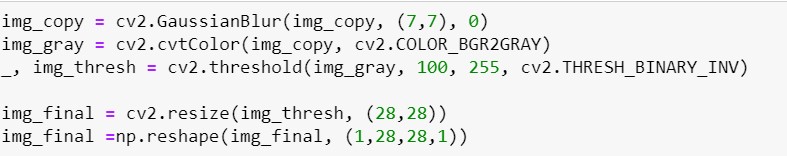


##### 13. Prediction on External Image

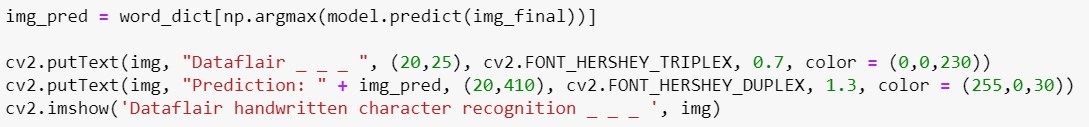
* Here we have read an external image that is originally an image of alphabet ‘D’ and made a copy of it that is to go through some processing to be fed to the model for the prediction that we will see in a while.
* The img read is then converted from BGR representation (as OpenCV reads the image in BGR format) to RGB for displaying the image, & is resized to our required dimensions that we want to display the image in.



* Now we do some processing on the copied image (img\_copy).
* We convert the image from BGR to grayscale and apply thresholding to it. We don’t need to apply a threshold we could use the grayscale to predict, but we do it to keep the image smooth without any sort of hazy gray colors in the image that could lead to wrong predictions.
* The image is to be then resized using **cv2.resize()** function into the dimensions that the model takes as input, along with reshaping the image using **np.reshape()** so that it can be used as model input.



* Now we make a prediction using the processed image & use the np.argmax() function to get the index of the class with the highest predicted probability. Using this we get to know the exact character through the word\_dict dictionary.
* This predicted character is then displayed on the frame.



* Here we are setting up a waitKey in a while loop that will be stuck in loop until Esc is pressed, & when it gets out of loop using cv2.destroyAllWindows() we destroy any active windows created to stop displaying the frame.



#### CONCLUSION

Classification of characters and learning of image processing techniques is done in this project. Also, the scheme through which project is achieved is Artificial Neural Network scheme.

We have successfully developed Handwritten character recognition (Text Recognition) with Python, Tensorflow, and Machine Learning libraries.

Handwritten characters have been recognized with more than 97% test accuracy. This can be also further extended to identifying the handwritten characters of other languages too. There are also different methods through which ‘handwritten character recognition’ can be achieved.

**REFERENCES**

1. [IJSC\_Vol\_12\_Iss\_1\_Paper\_1\_2457\_2463.pdf (ictactjournals.in)](http://ictactjournals.in/paper/IJSC_Vol_12_Iss_1_Paper_1_2457_2463.pdf)
2. [Algorithms | Free Full-Text | Convolutional-Neural-Network-Based Handwritten Character Recognition: An Approach with Massive Multisource Data | HTML (mdpi.com)](https://www.mdpi.com/1999-4893/15/4/129/htm)
3. [IJSRD - International Journal for Scientific Research & Development| Vol. 6, Issue 02, 2018 | ISSN (online): 2321-0613](http://www.ijsrd.com/articles/IJSRDV6I21900.pdf)
4. [Handwritten Optical Character Recognition Calculator using CNN and Deep Learning | by AI](https://medium.com/ai-techsystems/handwritten-optical-character-recognition-calculator-using-cnn-and-deep-learning-fcc432f88b63)
5. [Technology & Systems | AITS Journal | Medium](https://medium.com/ai-techsystems/handwritten-optical-character-recognition-calculator-using-cnn-and-deep-learning-fcc432f88b63)