2022 International Conference for Advancement in Technology (ICONAT)
Goa, India. Jan 21-22, 2022

Smart Bot for Handwritten Digit String Recognition

Mallikarjuna Rao Gundavarapu
Computer Science Engineering
Gokaraju Rangaraju Institute of Engineering and Technology (Autonomous)
Hyderabad, India
gmr_333@yahoo.in

Vivek Vardhan Reddy Yannam
Computer Science Engineering
Gokaraju Rangaraju Institute of Engineering and Technology (Autonomous)
Hyderabad, India
reddyvivek569@gmail.com

Akash Velagala
Computer Science Engineering
Gokaraju Rangaraju Institute of Engineering and Technology (Autonomous)
Hyderabad, India
akash.velagala@gmail.com

Snehith Reddy Lankela
Computer Science Engineering
Gokaraju Rangaraju Institute of Engineering and Technology (Autonomous)
Hyderabad, India
snehithreddy.lankela999@gmail.com

Saaketh Koundinya G
Assosiate Consultant
SAP
Benguluru, India
three2saki@gmail.com

Sai Chandan Regonda
Computer Science Engineering
Gokaraju Rangaraju Institute of Engineering and Technology (Autonomous)
Hyderabad, India
saichandanregonda10@gmail.com

***Abstract*—Handwritten digit string recognition is more sophisticated than determining a single digit individually. The repeated recognition of single digits is applicable for recognizing a handwritten digit string. A similar approach is exercised in this paper. The proposed approach could be advantageous in banks to recognize the digits written on the cheque and processes the cheque. Further, the banks could send the audio message of the recognized handwritten digits to the cheque issuer for confirmation before cashing the cheque. The pro-posed model is developed in the python platform and is lightweight, robust, and cross-platform.**

In this approach, we have trained a neural network model with MNIST handwritten digits dataset and some samples of our own for recognizing the handwritten digits. The Convolution Neural network models are widely used in the present-day technologies for object recognition, image processing, segmentation, face recognizing and also many identifications related tasks. The CNN model used in the project determines the digit in the image provided. Finally, the system plays the [pre-]recorded audio and displays the output for the recognized digits in the given digit string.

Keywords—Handwritten string detection, Digit determination, Audio output, Image seg-mentation, Neural Networks

I. INTRODUCTION

Handwritten digit recognition has been a topic that interests many with both aca-demic interest and commercial interests. And many new and advanced approach-es relating to the recognition of handwritten digits using neural networks, multi-layer perceptron models, and other classification models are introduced in recent times. Handwritten digit string recognition, a subject falling into a similar category as handwritten digit recognition. Determining the symbol of handwritten digits is called handwritten digit recognition. Handwritten digit string recognition put forth results in various fields involving data entry, postal mail checking, bank cheque processing. Digit string recognition has been a complicated problem for many years but still evolving in respects including identifications of digits writ-ten in various styles and having various characteristics. The computer cannot interpret the digits that a human eye can identify and match them with their name in a split second. The eye transmits the signals related to the shape of the digits that a human is observing. And human brain receives those signals and maps the transmitted signal to the related digits. Such a phenomenon is used repeatedly for recognizing the digits of a handwritten digit string. Whereas the computer cannot recognize them due to lack of data about patterns and features of the individual digits. The methodology executed in this project is detecting the handwritten digits individually of a given digit string after segmentation. The application starts with prompting the user to provide an input image path. The image is further segmented and processed into the splits of individual digits. Both the seg-mentation and the input image processing are handled using the OpenCV library. The segmentation process begins after preprocessing the image. The preprocessing proceeds with changing the image format from color to binary format followed by denoising using non-local means denoising and edge identification using the Canny edge detection algorithm. The binary image is further processed into binary inverse for splitting the string into digits. The segmentation involves almost all the border pixels of the rectangle enclosing the as black. The width of the rectangle contours is taken to be minimum to decrease the noise intervention in segmented digits recognition. The size is increased afterwards by the same amount so that no part of the digit is lost. The segmented splits are then deter-mined based on the probability of matching features and thus simultaneously voicing out the detected digits audio. The recognition of individual digits uses a trained NN model with Adam optimizer of learning rate 0.01 and categorical cross entropy as loss function, consisting of CNN layers combined with Max-pooling layers and concluding the model with two dense layers having ten nodes in the last layer, on MNIST dataset and some other samples consisting of up to 60000(MNIST) training data samples up to an accuracy 99%. The pydub library is used to play the recorded audio of the respective digits using the play method from the playback module and AudioSegment class of pydub.

II. LITERATURE SURVEY

Handwritten digit recognition, character recognition, digit string recognition dates before the late 20th century. Humans tend to have different styles and write digits and characters in different shapes. This problem isn't as simple as recognizing digits for humans to machines. Humans have written and observed digits since their preschool age. Since those times human brain evaluates patterns and extract

features relating to respective digits from various digits. But whereas a machine is void of such a vast collection of data tending it to misstep using limited data samples. Even humans with a complex neural system have a hard time determining handwritten digits. It's no wonder machines are drawn back to perfectly recognize these handwritten digits. Moreover, some of these digits have similarities with others - 4 and 9, 1 and 7, 5 and 6, 8 and 9, etc. - rendering them even harder for machines to recognize. Even with all these constraints, the approach to recognize handwritten digits advances day by day and likely to achieve perfect recognition of handwritten digits.

Handwritten digit string recognition is an extension of individual handwritten digit recognition. Briefly, repeated determination of individual digits of the digit string leads to handwritten digit string recognition, which might not be as easy as it sounds. Handwritten digit string recognition is a two steps process. It is composition of splitting string into individual digits using either heuristics or image segmentation based on various factors like common points, edges. The digits segmented from the digits string are passed as inputs to a model for recognition of digits. Secondly, classification of separated digits, the model used for classification of digits including the technique employed for feature extraction plays major role in determination of handwritten digit. Ultimately, concatenating the individual digits forms the digit string to be determined.

The main goal of our project is to recognize the handwritten digit string and al-so provide voice output of recognized digits. This approach helps in conforming to the digits written without paying attention to them. Such addition of voice out-put also helps children in learning the digits with their name at the early stages of schooling. In this model, we designed a neural network model, consisting of con-volution neural network layers with activation function rectified linear unit, followed by max-pooling layers and dense layers associated with dropout layers on MNIST handwritten digits dataset, containing digits from 250 different authors containing up to 70000 data samples on whole – 60000 samples in train dataset and 10000 samples in training dataset, and some of own samples of size 28x28. In these recent years, the fame of handwritten digit and character recognitions have risen high due to their applications, including cheque analysis, recognizing postal codes, and data entry jobs in various fields. Our approach extends the recognition with the voice output of recognized digits. This model is advantageous in attesting the money written on cheques and reciting students' marks in classrooms. Also helps blind people to recognize digits in the given numeral string with a simple snap. This model on further improvements can be used as teaching tool that helps preschoolers to learn digits and their names with the model's voice output module.

III. Proposed System Architecture

The architecture diagram represents the outline of the proposed software system. It provides summarization about the associations between various components and also the conditions involved between these components. It minimizes entire view of the software system into understandable format. Firstly, the user provides the image path of digit string to the application. Then the application reads and preprocesses the input image. The preprocessing includes image resizing, gray scaling, and denoising. The preprocessed image then undergoes the segmentation process, where the image is segmented into various individual digit image splits. Later, these digit splits are preprocessed individually. The CNN model takes the preprocessed splits for digit recognition. After determining the digits splits, the audio output for the recognized digits is played. Finally, the digits in the digit string are displayed on the screen.

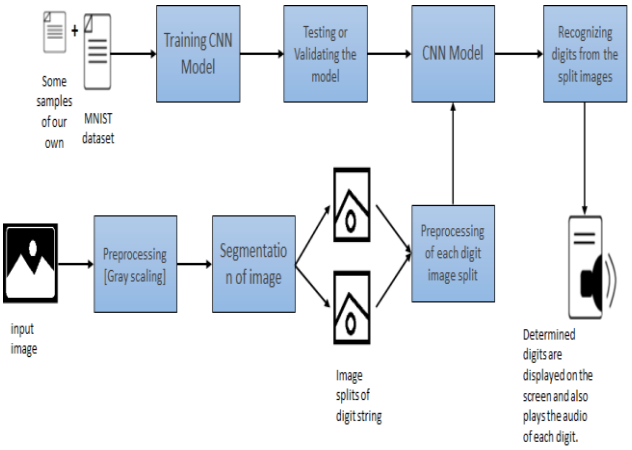

Fig. 1. System Architecture

The architecture above gives a clear picture about working of project. It also depicts the importance of various library functions in the project including OpenCV, Keras, imutils, pydub. The architecture is a sophisticated model that hides the underlying source code. The above figure helps us to determine the complexity of the application and can also be used as a reference for measuring the time required to determine the results.

IV. Development Framework

A. Software Installation

1) Visit the following website https://www.python.org /downloads /release/python-3.9.6/
2) Download python software according to your system requirements
3) Now Run the downloaded installer
    a. Right Click on the installer
    b. Select Run as Administrator and click on Yes
4) Now Python Installer Wizard appears
    a. Check the option Add Python 3.9 to PATH
    b. Click on Install Now
5) After the setup was successful, click on close
6) Open command prompt
    a. On the taskbar, to the bottom left, there will be an option to search. Type cmd in search and open the command prompt.
    b. Execute the following command to check if python is properly installed.
    • python –version
    c. If the version is displayed then the python is properly installed.
7) Execute the following command to update pip
    • pip install --upgrade pip
8) Install OpenCV by executing the following command.
    • pip install opencv-python
9) Install imutils by executing the following command.
    • pip install imutils

10) Install Keras by executing the following command.
- pip install tensorflow
- pip install keras

11) Install Numpy by executing the following command.
- pip install numpy

12) Install pydub by executing the following command.
- pip install pydub

B. Execution

1. Input Image: The execution of the program starts with taking an image as input from the user. For accurate results use a good resolution image as input. The user provides the path of the image as input. In order for the application to run correctly the user must provide a valid path as an input. It is important to pro-vide the input as the absolute path rather than relative path.

2. Loading Image: OpenCV module in python provides a method, imread, to read an image form the given path. The image should be present in the same directory as the program or else the absolute path of the input image must be pro-vided as input. cv2.imread is used to read the image from the path provided.

img = cv2.imread("image_path_input")

3. Preprocessing of the image: This phase involves the preprocessing of the input image (img). The img undergoes various transformations - resizing, gray scaling, denoising, binary inversion, dilation using a rectangle shape. In order to convert the given image into an array of values of color intensity. The segmentation process starts after preprocessing of image. The preprocessing proceeds with changing the image format from color to binary format followed by de-noising using non local means denoising and edge identification using the Canny edge detection algorithm. The OpenCV methods cvtColor and threshold are used for gray scaling and binary inversion.

4. Training the CNN model: Before proceeding further, train the CNN model for determining the digits. It is not a mandatory step to execute if you have a saved model for recognizing handwritten dataset. In this phase, we build a neural network model. And train it with the MNIST dataset (consisting for approximately 60,000 28x28 images) and a dataset containing some of our own samples of 28x28 images for recognizing the handwritten digits. We use Keras library to build and train the CNN model using CNN layers and fit method. Testing of the model is also implemented in this phase using evaluate method.

5. Save model: After training the model, in order to prevent the redundant training of the model, we can use save method in Keras to save the trained model. The saved model can be used again using load_model method in Keras. There would be no data loss for saving and loading the model.

model.save ("model_name")

model = load_model("model_name")

6. Segmentation of input image: After the preprocessing step of image, the image undergoes segmentation based on the splits identified by findContours function in OpenCV module. The segmentation involves almost all the border pixels of the rectangle enclosing the as black. The width of the rectangle is taken to be minimum to decrease the noise intervention in segmented digits recognition. The size is increased afterwards by the same amount so that no part of the digit is lost. These segmented parts are extracted from the original image.

7. Processing the splits individually: From here on, every phase is executed individually for each split of original image. The split image is resized into 28x28 image to pass it to CNN model. Before passing it to the model, the image undergoes gray scaling and binary inversion The resize function in imutils module is used for this purpose. After resizing, the image is fed into the CNN model.

8. Recognizing the digits in image split: The resized image after the preprocessing is fed into the Convolution Neural network model. The model recognizes the dig-it present in the splits based on the probability of matching features. The predict method in the Keras library is used for predicting the digit in the split image. After recognizing the digits separately. It's time to combine the recognized digits in the order of splits into an array.

9. Audio Output: In this phase a set of 10 recording, each for a digit from 0 to 9 has been initialized. Finally, the recorded audio is played based on the order digits stored in an array in previous step, using pydub module. The AudioSegment class of pydub is used to read the recorded audio files using pydub.playback.play function. Finally, the program terminates itself at the end of execution.

V. Existing Approaches

There are quite a few approaches in this field of study. An approach using both segmented and segment-free methods is employed to recognize handwritten numeral string off-line [1]. A method is proposed to recognize the handwritten numeral string using the RNN model and Connectionist Temporal classification with approximate recognition rates of 89.75% and 91.14% [2]. Another approach used graph-based properties of the digit string bases matching subgraph inputs with prototype symbol graphs [3]. A method is employed using (convolution neural network) CNN on ORAND-CAR-A and ORAND-CAR-B datasets with recognition rates of 92.2% and 94.02% respectively [4]. This problem is approached using various classification problems including SVM, KNN, and other classification algorithms over both the segmentation approach and the segment free approaches of splitting digit string [5, 6]. Some of these methods involve dealing with the overlapping of digits, connecting points in the digit string. Some other methods involving geometric context of digits and convolution neural network models [7, 8].

VI. Dataset

The MNIST dataset simply is thought of as "Hello World" in machine learning. Being derived from NIST dataset combined has 70000 images of size 28 pixels x 28 pixels black and white images, consisting up to 60000 images in the training dataset and approximately 10000 images for testing dataset contributed from more than 250 different writers. Each of the digit samples has varying styles and characteristics. The training dataset, 60000 images are the combination of ap-proximately 6000 images of each digit from zero (0) to nine (9) with maximum samples of 6742 for one (1) and minimum samples of 5421 for five (5) and others ranging in-between. And similarly, the testing dataset combines approximately 1000 images of each digit ranging from 892 (for five (5)) to 1135 (for one (1)). Although the

dataset is related to handwritten digit images, they can be stored in the form of a 28x28 2D matrix each cell representing the color intensity ranging from black to white. Due to its simple nature, the MNIST dataset is used in various handwritten digit recognition projects with varying accuracies and different models. Other than MNIST dataset we used some other handwritten digit samples for accurate results. And the dataset consists of few samples of each digit valuing a total of 5000 images, where each digit has approximately 500 samples extra from that of the MNIST dataset written in various styles and shapes.

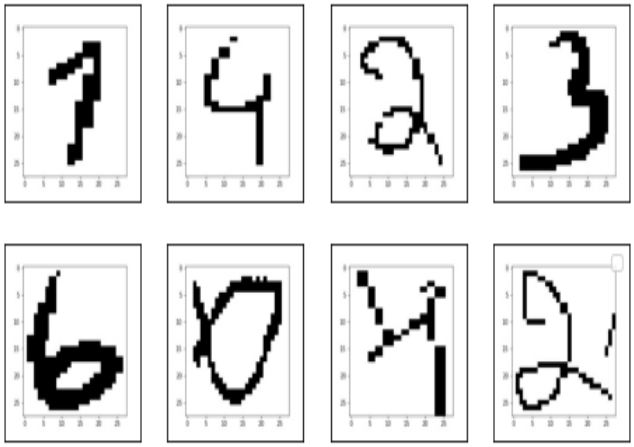

Fig. 2. Samples included into MNIST dataset

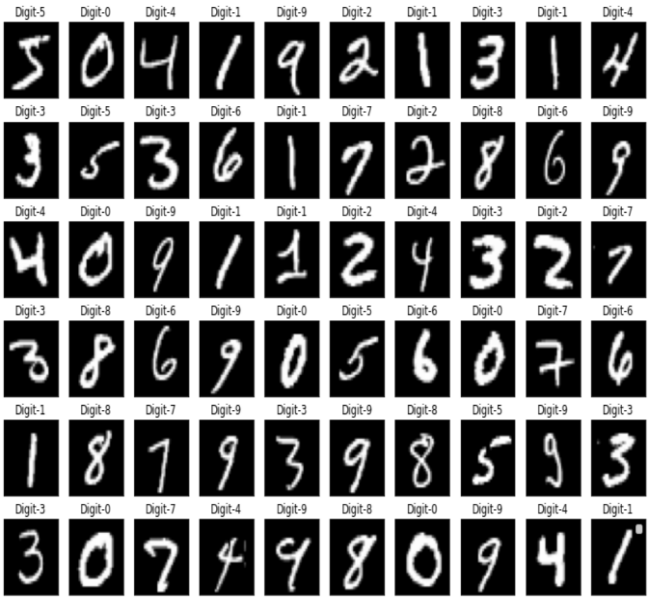

Fig. 3. Samples of MNIST dataset

VII. Experimental Results

After installing python environment and the required modules. The application is ready for execution and results. Firstly, the application accepts the path of the image as an input from the user. Then, the input string is converted into path with normpath method in os.path module. Care to be taken while providing the image, high quality images provide accurate results. Next, the imread method is used to the read the method from the given path. If the given path doesn't exist then the program terminates.

After reading the image, the image undergoes preprocessing - resizing the image, conversion of image to grayscale, denoising the grayscale - using resize function in imutils and cvtColor and non-local means denoising methods in OpenCV.

The preprocessing step also involves binary inversion of the gray scaled image. Next the binary inverted image undergoes dilation, a process expanding the image, with a unity matrix. Using this dilated image, we find the digit splits from the given image using findContours method in OpenCV module.

Before splitting the image and feeding them to the CNN model, we must train a CNN model using MNIST dataset. The results achieved using MNIST dataset are invalid in string recognition. Using MNIST dataset the model cannot recognize the digits that aren't well structured and has a lot of ambiguity constraints. And after training the model with some of our own samples in addition to the MNIST dataset, we have achieved some possible results for handwritten string recognition.

Each of the splits are individually preprocessed one by one and fed to the CNN model to determine the digits. The preprocessing of splits involves conversion of image to grayscale and binary inverse formats. Further, to fit the image to the CNN model input size, it is resized to size 28x28.

Finally, the audio output of the recognized digits is played using play method in pydub.playback module. Thus, in such a way the application provides the audio of digits in the given string of digits. Points to be noted while providing the image path are image must be of good quality for accurate determination and the image should exist in order to provide output.

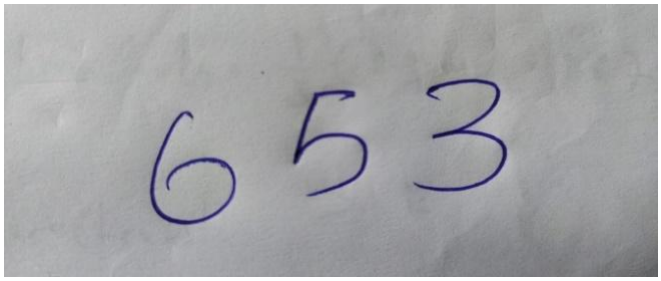

Fig. 4. Input image provided (handwritten digit string)

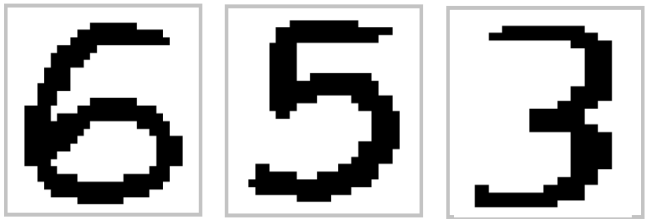

Fig. 5. Digit splits segmented from the original image (after processing the splits)

```
play digit 6
play digit 5
play digit 3
the given digit string contains 653
```

Fig. 6. Output displayed on the screen

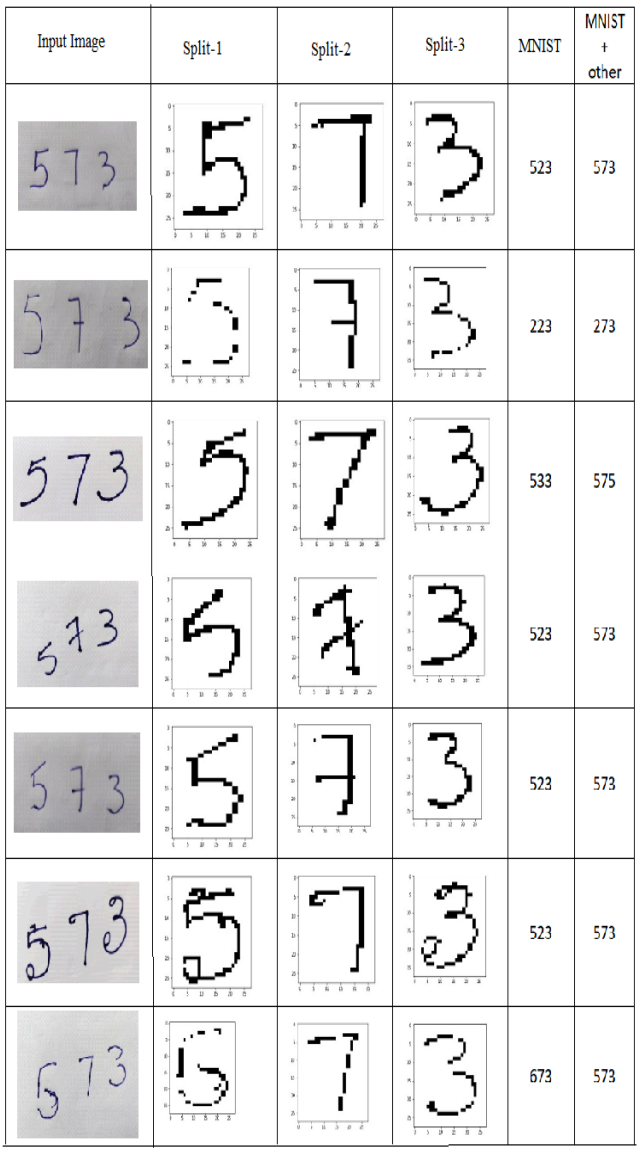

Fig. 7. Comparison of results with models trained using MNIST and MNIST + Additional samples

VIII. Pseudocode

//Load the image from the given input path
img = cv2.imread(image_path)
// convert the image to grayscale and denoise
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
// dilate the gray_img using unity matrix
// find the image splits and extract them from the dilated gray_img using findContours function
// resize the splits into size 28pixels x 28pixels
resized_split = imutils.resize(gray_split, (28, 28))
// recognize the digits from resized image splits using trained CNN model
// play the audio response of each recognized digits
// display the recognized digit string on the screen

IX. Conclusion and Future Scope

The proposed project by using various concepts, (Numpy, OpenCV, imutils, pydub, Keras), we are able to achieve:

* Determination of digits in the digit string.

* Segmentation of digit string image to individual digit split images.

* Recognition the 80% of digits in the digit split images by introducing other samples into MNIST dataset.

* Providing the audio output of the recognized digits.

Handwritten digit string recognitions can be said as one of the complex problems in classification problems of machine learning. The representations of digits change from person to person. There are endless styles for representing the same digits from zero (0) to nine (9) in various fashion. And to classify such vast collection of representation is tedious and complex. This project included a dataset of digit samples including different handwriting styles for accurate digit recognition. This helped the trained model to predict many of the ambiguous cases of the model trained by only MNIST dataset. Inclusion of further samples of digits data can certainly provide accurate results to the problem. Also, the voice function enables the user to recognize the digits in the digit string without looking into the text written. This project is also useful in determining the token numbers written at parking or lockers other than with postal codes, bank cheque processing and, in various field involving numerals.

References

[1] Thien M. Ha, Matthias Zimmermann, Horst Bunke, Off-line handwritten numeral string recognition by combining segmentation-based and segmentation-free methods, Pattern Recognition, Volume 31, Issue 3, 1998, Pages 257-272, ISSN 0031-3203

[2] Zhan H., Wang Q., Lu Y. (2017) Handwritten Digit String Recognition by Combination of Residual Network and RNN-CTC. In: Liu D., Xie S., Li Y., Zhao D., El-Alfy ES. (eds) Neural Information Processing. ICONIP 2017. Lecture Notes in Computer Sci-ence, vol 10639. Springer, Cham. https://doi.org/10.1007/978-3-319-70136-3_62

[3] A. Filatov, A. Gitis and I. Kil, "Graph-based handwritten digit string recognition," Pro-ceedings of 3rd International Conference on Document Analysis and Recognition, 1995, pp. 845-848 vol.2, doi: 10.1109/ICDAR.1995.602033.

[4] H. Zhan, S. Lyu and Y. Lu, "Handwritten Digit String Recognition using Convolutional Neural Network," 2018 24th InternationalConference on Pattern Recognition (ICPR), 2018, pp. 3729-3734, doi: 10.1109/ICPR.2018.8546100.

[5] L. S. Oliveira and R. Sabourin, "Support vector machines for handwritten numerical string recognition," Ninth International Workshop on Frontiers in Handwriting Recogni-tion, 2004, pp. 39-44, doi: 10.1109/IWFHR.2004.99.

[6] C. Zanchettin, B. L. D. Bezerra and W. W. Azevedo, "A KNN-SVM hybrid model for cursive handwriting recognition," The 2012 International Joint Conference on Neural Networks (IJCNN), 2012, pp. 1-8, doi: 10.1109/IJCNN.2012.6252719.

[7] X. -. Zhou, J. -. Yu, C. -. Liu, T. Nagasaki and K. Marukawa, "Online Handwritten Jap-anese Character String Recognition Incorporating Geometric Context," Ninth Interna-tional Conference on Document Analysis and Recognition (ICDAR 2007), 2007, pp. 48-52, doi: 10.1109/ICDAR.2007.4378673.

[8] Ahlawat S, Choudhary A, Nayyar A, Singh S, Yoon B. Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN).Sensors.2020;20(12):3344. https://doi.org/10.3390/s20123344