iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** V **Month of publication:** May 2024

DOI: https://doi.org/10.22214/ijraset.2024.61902

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue V May 2024- Available at www.ijraset.com

ScriptCalc: Handwriting Recognition with Mathematical Equation Solving

Asst. Prof. Moumita Dey[1], Trisha Das[2], Moushikta Shit[3], Debjit Rana[4], Papiya Biswas[5]
Department of Computer Science, Durgapur Institute of Advanced Technology & ManagementRajbandh, Durgapur

Abstract: ***The present paper proposes a novel algorithm for recognition of handwritten digits. For this, the present paper classified the digits into two groups: one group consists of blobs with/without stems and the other digits with stems only. The blobs are identified based on a new concept called morphological region filling methods. This eliminates the problem of finding the size of blobs and their structuring elements. The digits with blobs and stems are identified by a new concept called 'connected component'. This method completely eliminates the complex process of recognition of horizontal or vertical lines and the property called 'concavities'. The digits with only stems are recognized, by extending stems into blobs by using connected component approach of morphology. The present method has been applied and tested with various handwritten digits from modified NIST (National Institute of Standards and Technology) handwritten digit database (MNIST), and the success rate has been given. The present method is also compared with various existing methods.***

Once the digits are recognized, they are assembled back into their respective positions within the equation. Mathematical equation solving techniques are then applied to evaluate the expression and obtain the result.

Experimental results demonstrate the effectiveness of the proposed approach in accurately recognizing handwritten digits within mathematical equations. The method achieves competitive performance compared to state-of-the-art techniques, even in cases with complex equations and varied writing styles. This approach has potential applications in various domains such as education, document processing, and automated grading systems. By accurately interpreting handwritten digits within mathematical expressions, it can facilitate automated analysis of mathematical documents, assist students in learning mathematics, and streamline administrative tasks in educational institutions.

Keywords: Character recognition, ReLU function, Feature extraction, Handwriting Recognition, Deep learning Mathematical Equation Solving.

I. INTRODUCTION

Artificial intelligence (AI) in simple words is basically making a computer do the work that traditionally requires the human brain. AI has the ability to take in large amounts of data unlike the human and uses that data to recognize patterns, make decisions, and give judgment. In this AI we have a subset which is called Machine learning. ML is used to make computers to learn to behave ashumans. This is done by two ways, supervised learning in which the computer is given a set of input data and the required outputfor it.Now it uses ML to learn the algorithm to understand how that particular input gives this particular output. Now the unsupervised learning is when input data is provided but withno output, so the ML has to learn to analyze and clutter the datasets into categories.

This report presents a novel approach to address the issue of recognizing handwritten digits within mathematical equations. The proposed method integrates deep learning techniques with mathematical equation solving strategies to achieve accurate digit recognition and equation solving simultaneously. The motivation behind this work stems from the practical significance of automating the analysis of mathematical documents. In various domainssuch as education, research, and administrative tasks, the ability to interpret handwritten mathematical expressions efficiently can streamline processes and improve productivity.

The following are the main contributions of the proposed technique:

1) We presented an end-to-end deep learning framework based on the EfficientDet-D4 model for precise identification and classification of handwritten numerals .

2) The presented approach uses the lightweight backbone EfficientNet-B4 to computerobust and discriminative key points that improve the overall performance of HDRwhile reducing model training and execution time .

3) To exhibit the usefulness of the presented framework, a rigorous quantitative and qualitative comparison of the provided technique was undertaken using a standardbenchmark MNIST database. The findings show that the proposed CNN model improves recognition rates when compared to previous CNN-based algorithms.

This report is structured as follows: Section 2 provides an overview of related work in the field of handwritten digit recognition and mathematical equation solving. Section 3 describes the methodology and the proposed approach in detail. Section 4 presents experimental results and performance evaluation. Finally, Section 5 concludes the reportwith a summary of findings and potential avenues for future research. [1]

II. PROJECT OBJECTIVES

The central aim of this project is to develop an assistance system that analyses the hand- written mathematical equations based on handwriting recognition algorithms. The system will be able to recognize images of handwritten equations and output the corresponding characters in LATEX. The specific objectives include:

1) *Input for Digital Platforms:* Converting handwritten mathematical expressions into digital formats allows users to input equations into computers or devices that lack specialized input methods.
2) *Ease of Use:* Handwriting recognition simplifies the process of inputting complex mathematical notations. Instead of typing out equations, users can write them nat- urally, making it more intuitive, especially for individuals who are more comfortable with pen and paper.
3) *Accessibility:* For people with disabilities or specific learning needs, handwriting recognition can facilitate better accessibility to mathematical content by allowing alternative input methods.
4) *Integration with Software:* Integration of handwriting recognition with various software and applications, including note-taking apps, educational platforms, or mathematical software like Mathematica or MATLAB, enables users to seamlessly convert handwritten equations into editable digital formats.
5) *Enhanced Learning:* In educational settings, handwriting recognition can aid in teaching and learning mathematics by allowing students to express their mathemati- cal ideas more naturally and engage with digital resources effectively.
6) *Improving Workflows:* It can streamline workflows in various fields such as science, engineering, finance, etc., where mathematical equations are prevalent, allowing for faster data entry and manipulation.
7) *Versatility:* Handwriting recognition can cater to different handwriting styles, making it adaptable to various users and their individual writing habits.

These objectives collectively aim to bridge the gap between traditional handwrit- ten mathematical expressions and digital formats, making mathematical content more accessible, editable, and usable across a wide array of applications and platforms. [2]

III. LITERATURE REVIEW

A brief description of the contributions of this thesis is given below:

1) Symbol recognition in handwritten mathematical expressions has been a focal point of research, aiming to enhance the accuracy and efficiency of recognition systems. Simistira, Katsouros, and Carayannis (2016) proposed an innovative method for symbol recognition in their paper titled ”A Template Matching Distance for Recognition of On-Line Mathematical Symbols,” presented at the 11th International Conference on Frontiers in Handwriting Recognition. Their approach introduced a template matching technique that represents symbols as sequences of reduced Freeman chain codes along with corresponding local length proportions. The method addresses variations in writing speed by normalizing the strokes, enabling comparison through multiplexing individual feature sequences. The proposed distance metric evaluates differences in chain codes while considering the local length proportions of normalized sequences, establishing a comprehensive measure for symbol recognition. [3]
2) Our developed system, focusing on handwritten mathematical expression recognition, displayed promising results during evaluation on the CROHME 2011 dataset. To further enhance its performance across evaluation metrics (STrec, SYMseg, SYMrec, and EXPrec), ongoing efforts involve expanding its capabilities to handle more complex expression structures, refining baseline detection using slope correction tech- niques , and integrating grammatical rules for constructing mathML structures. This work is part of our preparation to submit an improved algorithm for consideration in CROHME 2012 (Simistira et al., 2017).
3) Recognizing handwritten mathematical expressions presents challenges due to the extensive symbol variations and ambiguities. Awal, Mouchère, and Viard-Gaudin (2019) emphasized the complexities in their work ”Towards Handwritten Mathemat- ical Expression Recognition,” highlighting the need for robust methods. [4] With over 220 symbols and ambiguities between similar ones, distinguishing among them remains a challenge. Template matching and structural

recognition methodshave been explored, while artificial neural networks (ANNs) and hidden Markov models (HMMs) have shown promise in enhancing recognition efficiency. Despite advancements, improving recognition rates across evaluation measures (STrec, SYMseg, SYMrec, and EXPrec) remains a focus. Research continues to target complex expression structures, refining baseline detection techniques, and integrating advanced algorithms for more accurate recognition.

4) In our study, we focus on training a classifier capable of categorizing symbols into 75 distinct classes. Initially, we employ an SVM classifier, previously trained for CS221, serving as our baseline model. This SVM model achieved an accuracy of 87 percentfor both test and training datasets. Despite our belief in the superior performance of CNNs (Convolutional Neural Networks), we establish another baseline using fully-connected neural networks (NNs) for comparative purposes. Our primary emphasis lies in optimizing CNNs for character classification. Our approach involves extensive efforts directed at training multiple CNN models, varying in architectures and parameters. The objective is to identify and establish the most effective CNN configuration for achieving superior performance in the symbol classification task. [5]

5) Neural networks have been pivotal in document image preprocessing, as highlighted by Rehman and Saba (2014) in their comprehensive review titled ”Neural Networks for Document Image Preprocessing: State of the Art”. This work delves into the current advancements and methodologies employed in utilizing neural networks for document image preprocessing tasks, shedding light on the state-of-the-art practices.

In the realm of optical character recognition (OCR) systems, Shah and Gokani (2018) introduced an effective approach tailored for digit recognition. Their work, titled ”A Simple and Effective Optical Character Recognition System for Digits Recognition Using the Pixelcontour Features and Mathematical Parameters,” focuses on leveraging pixelcontour features and mathematical parameters to build an efficient OCR system specifically for recognizing digits.

Additionally, Simard, Steinkraus, and Platt (2015) provided invaluable insights in their paper titled ”Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis”. This work outlines the best practices and methodologies specifically designed for applying convolutional neural networks (CNNs) in the domain of visual document analysis, offering crucial guidelines for enhancing CNN performance in this context. [6]

These cited works collectively underscore the significance of neural networks, partic- ularly CNNs, in document image preprocessing, optical character recognition, and visual document analysis, contributing to the advancement of efficient methodologies and state-of-the-art practices in these domains.

A. Existing System

Various algorithms used for implementing handwritten digit recognition systems consist of Proximal Support Vector Machine (PSVM), Multilayer Perceptron, Support Vector Machine (SVM), Random Forest, Bayes Net, Naive Bayes, J48, Random Tree. [7]

1) Proximal SVM - 98%
2) Multilayer Perceptron - 90%3. SVM - 87%
3) Random Forest - 85%
4) Bayes Net - 84%
5) Naive Bayes - 81%

Even though these algorithms may prove to be useful in some of the applications basedon this technology, many other applications such as banking industry applications require better results which can be achieved using other algorithms as compared to the algorithms that are mentioned above.

B. Proposed System

To reduce error and obtain more efficiency overall, Convolutional Neural Network (CNN) can be used to implement handwritten digit recognition systems. For achieving so, our proposed system uses CNN with multiple pooling and convolutional layers alongside a kernel of 3x3 size. Our model uses 60,000 28*28 grayscale images during the training process. Our model is trained through a standard 5 epochs to achieve accuracy of the order of 99.16% which is much higher as compared to the traditional algorithms such as SVM, Multilayer Perceptron, Bayes Net, Random Forest, etc. used to implement handwritten digit recognition systems. [8]

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue V May 2024- Available at www.ijraset.com

IV. METHODOLOGY

Importing the libraries: Libraries are useful tools that can make a web developer'sjob more efficient. It's a set of prewritten code, that we can call while programming our own code. Basically, it's the work that's already done by someone else thatyou can make use of, without having to do it yourself. You can also use it in yourown code. Different libraries have different restrictions on fair use, but this is a code that was designed to be used by others, instead of just standing alone. The libraries used in this code are -

1) *OpenCV:* OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software libraries. It's a library in python which is designed to solve computer vision problems. OpenCV (also known as CV2)was built to provide a common infrastructure for computer vision applications andto speed up the use of machine perception in all kinds of products.
2) *NumPy:* NumPy stands for Numerical Python. We use the NumPy library to work with arrays. It can also be used to work in the domain of linear algebra, matrices and Fourier transform. NumPy was created by Travis Oliphant in 2005. It's an open source project and can be used freely.
3) *Pandas:* Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data. The name "Pandas" has a reference to both "Panel Data", and "Python Data Analysis" and was created by Wes McKinney in 2008.
4) *Matplotlib:* It is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. Create publication quality plots. Make interactive figures that can zoom, pan, update.
5) *Keras:* It is the high-level API of the TensorFlow platform. It provides an ap- proachable, highly-productive interface for solving machine learning (ML) problems, with a focus on modern deep learning. Keras covers every step of the machine learning workflow, from data processing to hyperparameter tuning to deployment.
6) *PIL:* It stands for Python Image Library. In this article, we will look at its fork: Pillow. PIL has not been updated since 2011 and so the case for Pillow is obvious. The library provides support for various image formats including the popular JPEG and PNG formats. Another reason you would consider using Pillow is the fact that it is quite easy to use and very popular with Pythonistas. The package is a commontool in the arsenal of most data scientists who work with images.

The project consists of seven chapters, and the organization of the project is as follows: There are ten digits in English language and each digit is differentiated fromthe other digits by some characteristic feature(s). Recognition of the ten numerals appears simple at first. However, the problems that arise due to similarities between different numerals and discre-pancies between the same numeral must be tackledby analysing the similar and dissimilar features and then decisions should be made accordingly.

The present paper divided the ten digits of English language into two groups. Group 1 consists of digits with blobs with/without stems. This group consists of digits 0, 4, 6, 8, and 9. Group 2 consists of digits with only stems, digits 1, 2, 3, 4, 5, and 7. The group 1 is further divided in to two subgroups i.e. the digitswith only two blobs 8and another with a single blob with or without stems0, 4, 6 and

The blobs are identified by region filling method which is different fromprevious methods. It is used to eliminate extra problems due to non-class-specific differences like

a) Size
b) Shear
c) Line thickness
d) Background and Digit colors
e) Resolution, etc.


```
[8]: image = cv2.imread(train_image[140])
     plt.imshow(image)
     plt.title("Label: " + train_label[140])
     plt.show()
```

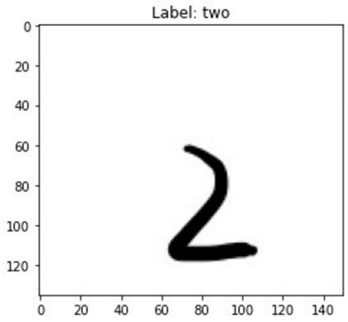


Figure 1:

A. Preprocessing

The process begins with preprocessing the input image containing the mathe-matical equation. Preprocessing steps include image binarization to convert the image into a binary format, which enhances contrast and reduces noise.

```
[9]: image = cv2.imread(train_image[500])
     plt.imshow(image)
     plt.title("Label: " + train_label[500])
     plt.show()
```

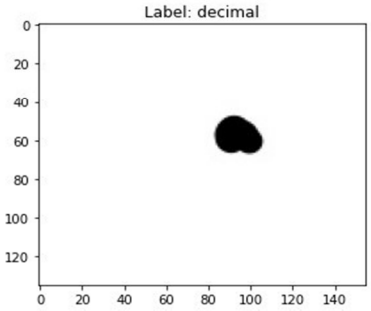


Figure 2:

- Following binarization, contour detection algorithms are applied to identify individual components within the equation, such as digits, operators, and symbols.
- Bounding boxes are then generated around each detected component to isolate them for further processing.

B. Digit Recognition with Convolutional Neural Networks (CNNs)

- Once the components are segmented, handwritten digits are identified usingdeep learning techniques, particularly CNNs.
- A pre-trained CNN model, such as LeNet, is fine-tuned on a dataset of hand- written digits to adapt it to the specific task of recognizing digits withinmathematical contexts.
- The segmented digit images are resized to match the input size expected by the CNN model and then passed through the network for classification.
- The output of the CNN provides the predicted label for each digit, along with confidence scores.

C. Equation Reconstruction

- After recognizing individual digits, they are reassembled into their original positions within the mathematical equation.
- The reconstructed equation is then passed through a mathematical equation solverto evaluate the expression and obtain the result.
- Various mathematical equation solving techniques can be employed depending onthe complexity of the equation, such as parsing and evaluating the expression treeor using symbolic algebra libraries.

Figure 3: Checking Loaded Image

D. Integration and Post-processing

- Finally, the recognized digits and the solved mathematical expression are integrated into a cohesive output format, such as a text-based representation or an annotated image.
- Post-processing techniques may be applied to refine the output and improve read-ability, such as removing redundant symbols or formatting the equation for clarity.

E. Model Evaluation

- The performance of the proposed methodology is evaluated using standard metrics such as accuracy, precision, recall, and F1-score.
- Evaluation is conducted on benchmark datasets containing handwritten mathemati-cal equations with ground truth labels.
- Cross-validation techniques may be employed to assess the generalization perfor-mance of the model on unseen data.

F. Implementation Considerations

- The entire methodology is implemented using appropriate programming languages and libraries, such as Python with OpenCV for image processing, TensorFlow or PyTorch for deep learning, and mathematical equation solving libraries like SymPy or Mathpix.
- The system may be deployed on various platforms depending on the intended application, including desktop applications, web services, or embedded systems.

G. Optional Equation Simplification

- Simplify equations by reducing redundant terms or rearranging expressions.
- Apply mathematical rules or operations to simplify complex equationsOptional Equation

H. Simplification

- Display solved mathematical equations and their corresponding solutions.
- Provide clear and formatted results for easy interpretation.

V. FLOWCHART

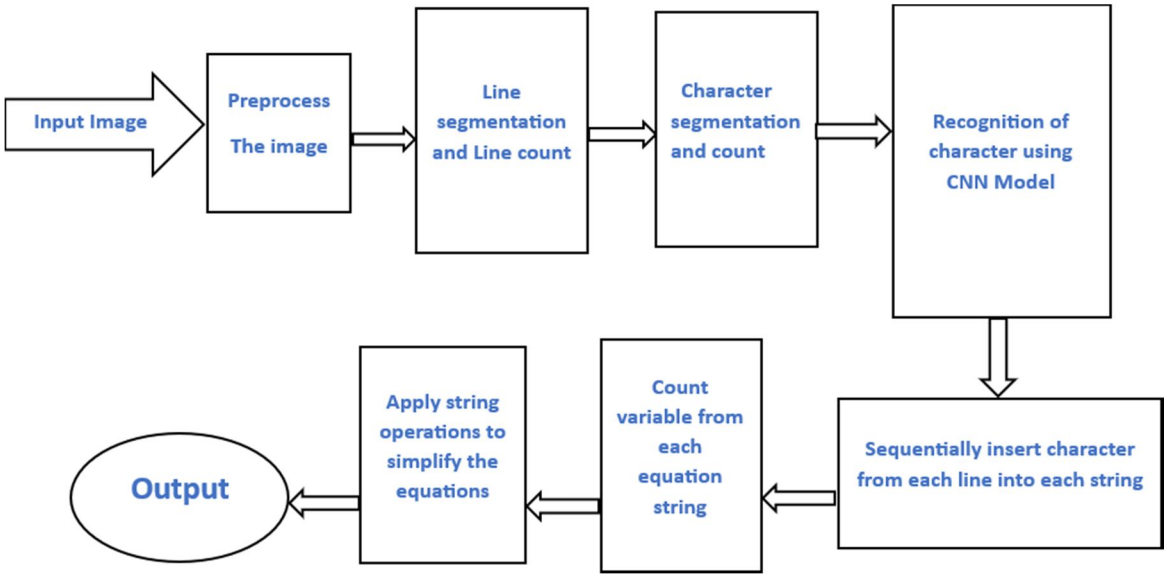


Figure 4: Methodology used in Implementation of Handwritten Recognition

VI. TOOLS AND ALGORITHMS USED

1) The neural network used isSequential Model - Sequence models are basically the models in machine learning that analyze input sequences and produce output sequences of data. Sequential data includes audio clips, text streams, time-series data, video clips, etc. Recurrent Neural Networks (RNNs) is a majorlyused algorithm used in sequence models. It's basically a linear path which connects all the input, hidden and the output layers withoutskipping any layers in between, as shown . A Sequential modelis ideal for a plain stack of layers where each layer has exactly one input tensor, oneor more hidden tensors and one output tensor.

2) Optimizer used in this code isADAM –stands for Adaptive Moment Estimation.It is an algorithm and a technique to obtain gradient descent. This method is very effective in terms of working with large problems which also involves many parameters and data. Less memory is required and also, it's a combination of the "RMSP" algorithm and the "gradient descent with momentum" algorithm. Thisis used to accelerate the above algorithm by taking the weighted average of the gradients into notice. For the algorithm, making use of the averages makes the minima lean towards a faster pace. [9]

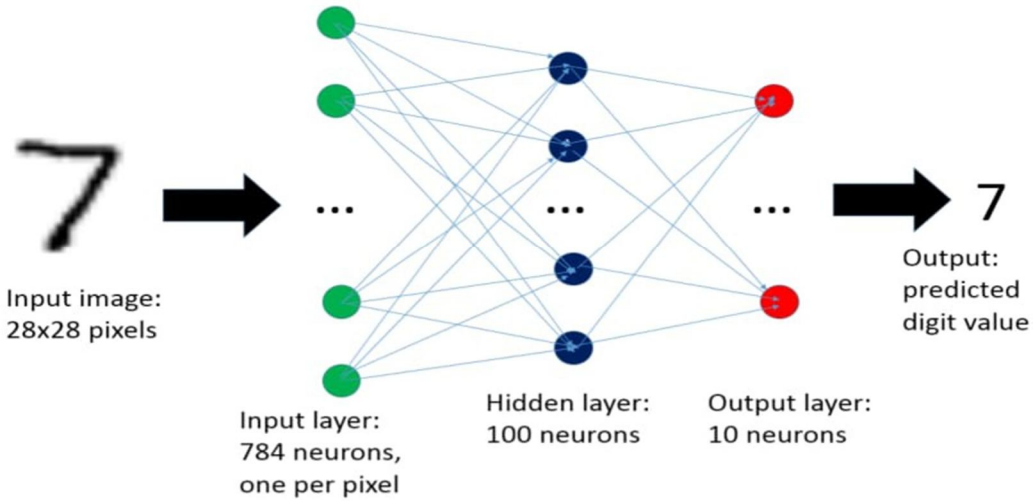


Figure 5: Digit recognition.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue V May 2024- Available at www.ijraset.com

VII. IMPLEMENTATION

In this paper, Neural network is implemented wherein the model recognizes and predicts a handwritten digit. Initially Tensorflow and Keras are used to form the bones of the implementation. We load the datasets from both of these open-source libraries and make our model to analyze thousands of images. The model learns all the patterns, pixel placements of the greyscale images and all the neural connections. Keras is an API (Application Programming Interface), which is designed for machine learning and deep learning. It's an open source library which has a lot of inbuilt data. It's the interface of Tensorflow library. [3]

1) It follows the best practices available for reducing cognitive load.
2) It provides simple & consistent APIs (Application Programming Interface).
3) It has extensive documentation and has developer guides.
4) It minimizes the amount of user actions needed for common use cases, and it providesactionable & clear error messages.

A. *Implementation Details*

1) *Digit Recognizer File:* The handwritten digit recognition system required to be created for this project will be made using the MNIST dataset. For doing so, the MNIST dataset willbe loaded to our digit recognition python script. After that a sequential CNN model will be created and a combination of convolution and pooling layers will be added. A 3x3 sized kernel will be used for filtering the digital image data. After the data is processed through pooling and convolution layers, the data is convertedusing the 'flatten' function to transform the multidimensional data input into a single dimension to transition to a fully connected layer. Activation function used for training the proposed CNN model will be 'relu'. After this the image data is converted from 28*28 grayscale image format to binary format matrix which is known as binarization of the image. The MNIST dataset is then divided to 60,000 training samples and 10,000 testing samples. Ultimately the model is compiled by training it through 5 epochs using the 'rmsprop' optimizer with a batch size of 64. The loss and accuracy of the proposed CNN model are then evaluated and the model is saved for using it in the GUI python file later. [10]
2) *GUI File:* At first the saved model is loaded into the GUI python file and the main GUI window which displays the canvas widget is created using the 'Tk' function.

A mainloop is created for the master window which is run infinitely until the user shuts the window down. A title related to the proposed project is then provided tothe main GUI window. The main window consists of two buttons namely 'Recognize Number' and 'Clear Canvas'.

After that the functions to implement functionalities such as clearing the canvas, drawing the digits, activation event for doing so, and digit recognition are defined.

To recognize the handwritten user-defined digit strings on the canvas displayed within the GUI, a list of contours i.e., a line that connects every point around the borderline of an image which has similar intensity, is created which is very useful for detecting the digit and analyzing its shape.

The 'Recognize Number' button is then pressed which initiates the model to predict each and every digit one by one. It displays the result in a new window where each digit is recognized separately and the accuracy with which they are recognized is also displayed. This new window is given the required title and consists of three other options alongside the recognized number. These three options ultimately provide the functionality of converting the recognized decimal number to binary/hexadecimal/octal numbersystem according to the user's choice. [11]

VIII. RESULTS AND DISCUSSION

In this section, we have provided a detailed description of the employed dataset along with the metrics which are used to assess the performance of the proposed model. Moreover, wehave performed a series of experiments to check the numeral detection and classification performance of the presented approach.

We have implemented the proposed method in Python language and executed it on an Nvidia GTX1070 GPU-based system. Table 2 displays the details of the training parameters of the proposed work. We have reported training and loss curves to show the optimized learning behavior of the proposed approach [12]

A. Accuracy Checking

```
# training the model
history = model.fit(
    X_train,
    y_train,
    batch_size=50,
    epochs=100,
    validation_split=0.2,
    shuffle=True
)
from joblib import dump

# Save the training history
dump(history.history, '/kaggle/working/training_history1.pkl')

# Save the trained model
model.save('/kaggle/working/trained_model1.h5')
```

```
Epoch 92/100
121/121 ━━━━━━━━━━━━━━━━━━━━ 4s 36ms/step - accuracy: 0.9933 - loss: 0.0219 - val_accuracy: 0.1389 - val_los
s: 31.9700
Epoch 93/100
121/121 ━━━━━━━━━━━━━━━━━━━━ 4s 36ms/step - accuracy: 0.9955 - loss: 0.0136 - val_accuracy: 0.1396 - val_los
s: 28.5338
Epoch 94/100
121/121 ━━━━━━━━━━━━━━━━━━━━ 4s 36ms/step - accuracy: 0.9962 - loss: 0.0080 - val_accuracy: 0.1402 - val_los
s: 27.0792
Epoch 95/100
121/121 ━━━━━━━━━━━━━━━━━━━━ 4s 36ms/step - accuracy: 0.9961 - loss: 0.0090 - val_accuracy: 0.1376 - val_los
s: 29.3367
Epoch 96/100
121/121 ━━━━━━━━━━━━━━━━━━━━ 4s 36ms/step - accuracy: 0.9962 - loss: 0.0122 - val_accuracy: 0.1409 - val_los
s: 26.3958
Epoch 97/100
121/121 ━━━━━━━━━━━━━━━━━━━━ 4s 36ms/step - accuracy: 0.9979 - loss: 0.0055 - val_accuracy: 0.1396 - val_los
s: 27.0742
Epoch 98/100
121/121 ━━━━━━━━━━━━━━━━━━━━ 4s 36ms/step - accuracy: 0.9971 - loss: 0.0171 - val_accuracy: 0.1396 - val_los
s: 26.7894
Epoch 99/100
121/121 ━━━━━━━━━━━━━━━━━━━━ 4s 36ms/step - accuracy: 0.9936 - loss: 0.0161 - val_accuracy: 0.1409 - val_los
s: 20.6966
Epoch 100/100
121/121 ━━━━━━━━━━━━━━━━━━━━ 4s 36ms/step - accuracy: 0.9954 - loss: 0.0148 - val_accuracy: 0.1409 - val_los
s: 27.4478
```

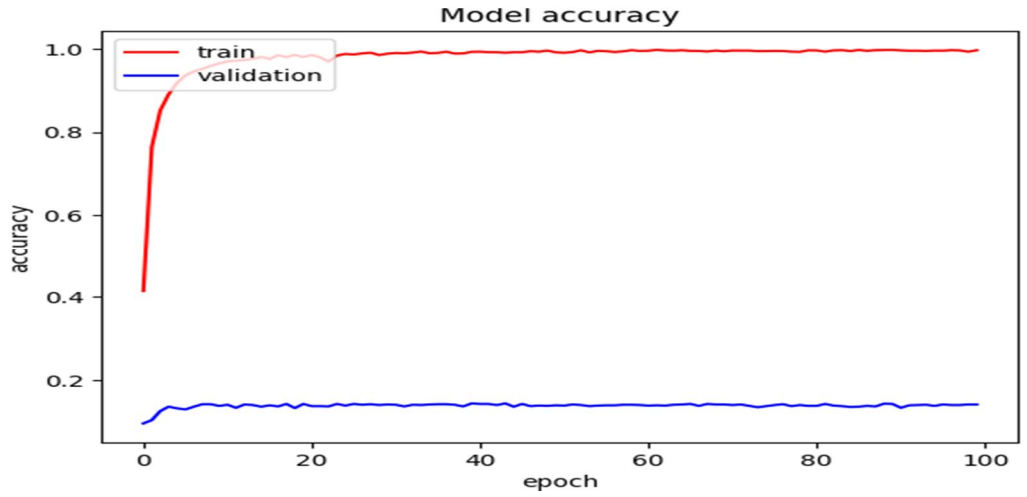


Figure 6: Model Accuracy

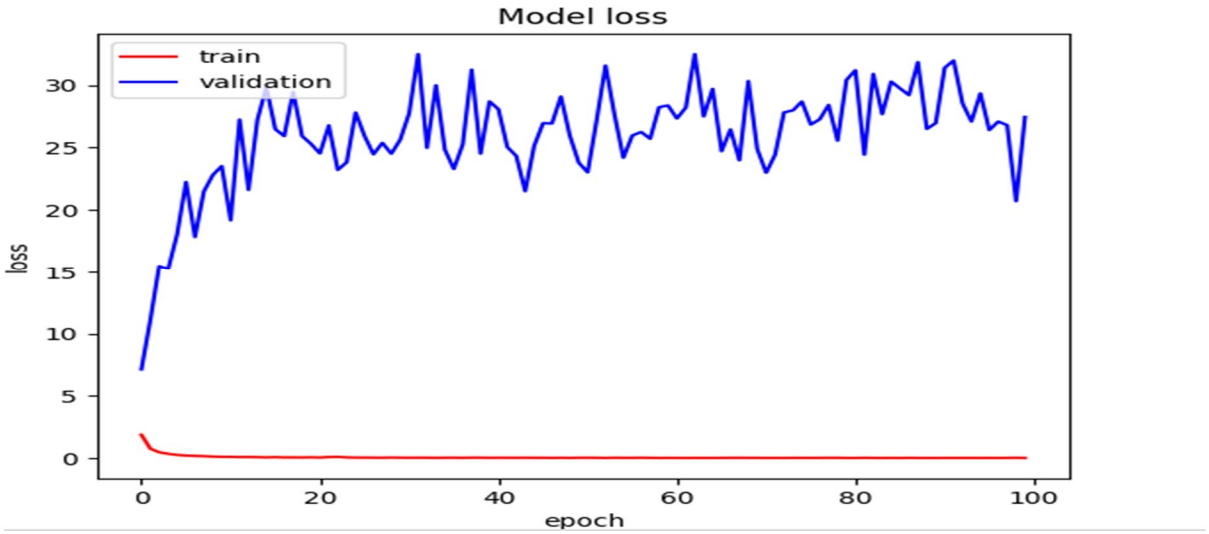
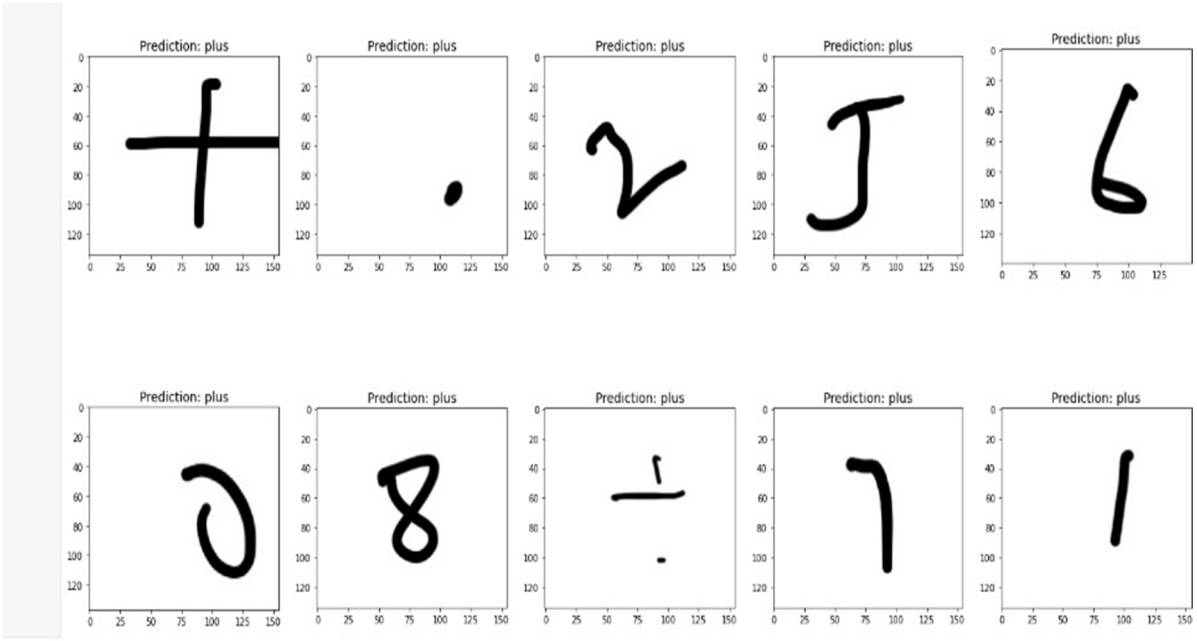


Figure 7: Model loss

Recognition of digits is aided by the use of morphological decision tree. In group 1 blobs are identified by the above said process and numbers of blobs are counted. Based on the number of blobs, the group1 digits are categorized in to two subgroups. A digit with two blobs is identified as and the digits with one blob are categorized as subgroup 1b. They are further recognized by looking at number of stemsand their position.

The digits with blobs and stems are identified by removing blob portion from the digit.The digit with zero stems is identified as and with two stems is identified as and digits with one stem are identified based on the position of the stem with respect to the blob that is either above the blob or below the blob. For the stem appears above the blob andfor stem appears below the blob. The digit four can be written. By this the first digit four can be placed in group 1 and the second four will be placed in group 2. The group 2 digits that consist of only stems are recognized by the following novel method. Initially, aline is drawn from top left pixel position of the digit to the bottom left positio n of the digit. By this blobs will be formed for digits as shown. [12]

```
fig, axs= plt.subplots(2,5, figsize=[24,12])
count=0
for i in range(2):
    for j in range(5):
        image = cv2.imread(test_image[count + count*100])
        img = cv2.resize(image, (100, 100))
        img = np.array(img)
        img = np.expand_dims(img, axis=0)
        img = img.astype('float32')
        img /= 255
        pred = model.predict(img)
        result = np.argsort(pred)
        result = result[0][::-1]
        final_label = label_encoder.inverse_transform(np.array(result))
        axs[i][j].imshow(image)
        axs[i][j].set_title(str("Prediction: " + final_label[0]), fontsize = 14)
        count += 1
plt.suptitle("All predictions are shown in title", fontsize = 18)
plt.show()
```



IX. CONCLUSION

Accurate recognition of numerals from images plays a significant role in the domain of information processing. However, a huge writing pattern difference and the presence of various sample distortions like noise, blurring, and intensity changes complicate the effective detection of HDR. In this work, a reliable DL-based HDR system, namely, EfficientDet-D4, is presented to resolve the existing issues of this domain. More clearly, input images are initially annotated to locate the position of digits on images, which are later used to train the EfficientDet model to detect and categorize the digits. We have evaluated the presented approach over the complex dataset, namely, MNIST, and attained an average accuracy value of 99.83%. We have confirmed through huge experimentation that the presented work can efficiently recognize the numerals from the test samples and categorize them into 10 categories showing numbers from 0 to 9. Moreover, the approach is capable of accurately identifying and classifying the digits even under the occurrence of various postprocessing attacks, i.e., light and color variations, blurring, noise, angle and size changes, etc. Furthermore, across-dataset evaluation on the USPS dataset is also accomplished to show the efficacy of the proposed method for the unseen cases. Evaluation results have assured that the introduced approach is robust against present modern techniques and can play a vital role in the area of information processing. Based on the computed results, we can say that this approach can play an important role in the area of automated number plate recognition of vehicles for surveillance applications. Furthermore, this work has an application in optical character recognition to facilitate various daily life tasks, i.e., product prices, receipt recognition, etc. In the future work, we plan to extend the proposed approach to be applied to other languages.

. With the results given from this work we are more confident in finding other ways to make this better and to make it easier for complex data like converting handwritten paragraphs into text. Through this research work we understood all the mechanisms usedto identify handwritten data. We understand the importance of hand recognition as it is easy for the user to write data on paper and use handwritten data recognition to convertit into text instead of the typing it on keyboard. Further it is recommended to implementon edge computing platforms like Raspberry Pi 4 system for actual usage. [11]

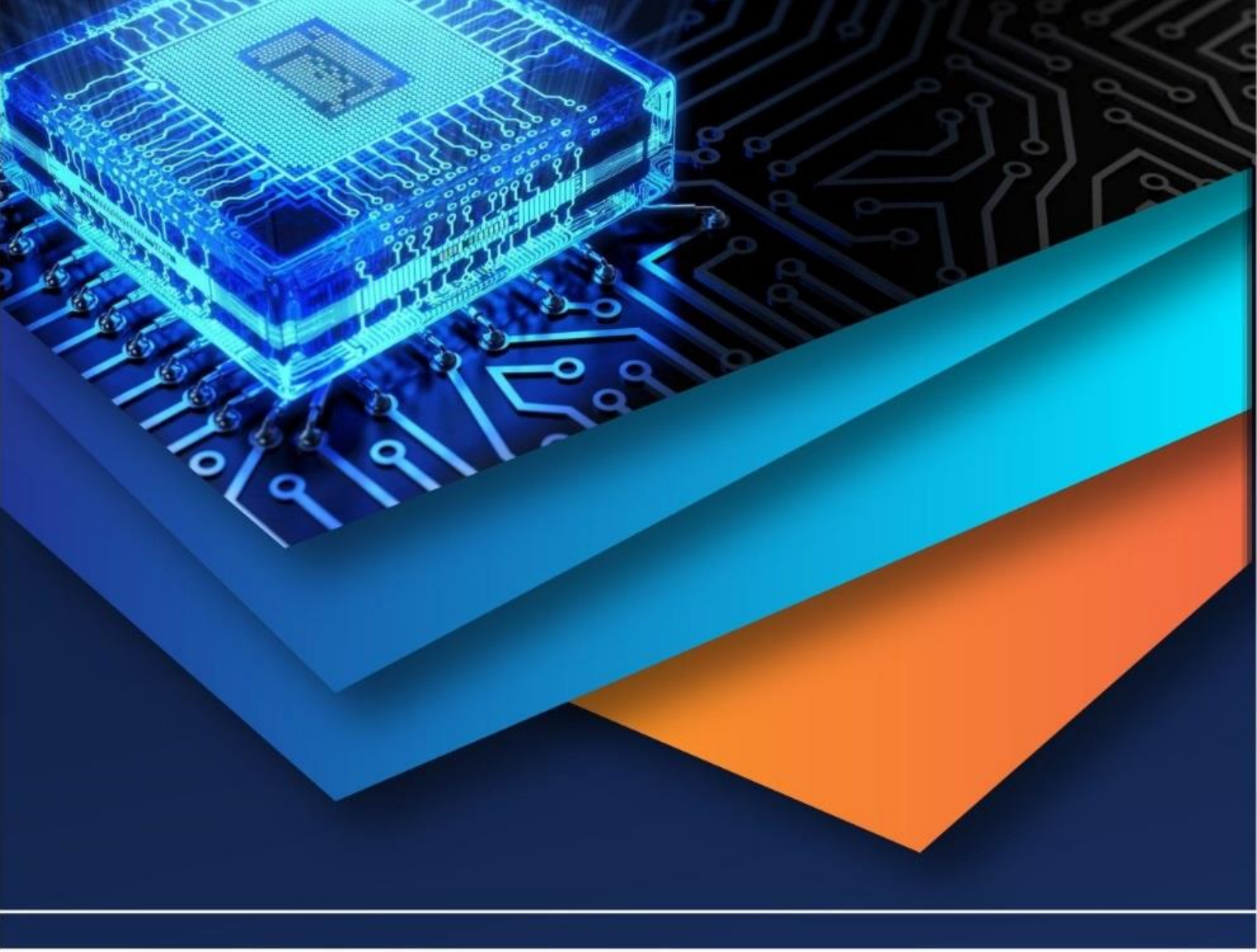
X. FUTURE WORK

A new method can be proposed to cut or segmenting the digit strings still there are some limitation for this method, where improvements has to be made. Thus, there is a place for some future work such as: Different classifications models can be used at a time to improve the performance of the segmentation. To reduce the complexity of the algorithm, it's better to reduce the number of hypothesis to function the algorithm faster.To reduce the computation time, better filters are to be used to eliminate the unnecessary segmentation hypothesis.

Firstly, to have more compelling and robust training, we could apply additional prepro- cessing techniques such as jittering. We could also divide each pixel by its correspondingstandard deviation to normalize the data. Next, given time and budget constraints, we were limited to 20 training examples for each given word in order to efficiently evalu-ate and revise our model. Another method of improving our character segmentation model would be to move beyond a greedy search for the most likely solution.We would approach this by considering a more exhaustive but still efficient decoding algorithm suchas beam search. We can use a character/word-based language-based model to add a penalty/benefit score to each of the possible final beam search candidate paths, along with their combined individual softmax probabilities, representing the probability of the sequence of characters/words. If the language model indicates perhaps the most likely candidate word according to the softmax layer and beam search is very unlikely given the context so far as opposed to some other likely candidate words, then our model can correct itself accordingly. Now, let's look ahead. There's a lot we can do to make this project even better. We want the system to understand different kinds of handwriting and trickier math problems. Making it easier for everyone to use, like on phones and in different languages, is also important. We're working on ways for the system to learn from its mistakes and get even better at recognizing and solving math problems. These improvements will help make the system more helpful and user-friendly for everyone.

REFERENCES

[1] M. F. Bin Othman and T. M. S. Yau. Comparison of different classification techniques using weka for breast cancer. In 3rd Kuala Lumpur Internanal Conference on Biomedical Engineering 2006, pages 520–523. Springer Berlin Heidelberg, 2006
[2] R. R. Bouckaert. Properties of bayesian belief network learning algorithms. In Pro- ceedings of the Tenth international conference on Uncertainty in artificial intelligence, pages 102–109. Morgan Kaufmann Publishers Inc., 1994.
[3] W. Buntine. Theory refinement on bayesian networks. In Proceedings of the Seventhconference on Uncertainty in Artificial Intelligence, pages 52–60. Morgan Kaufmann Publishers Inc., 1991.
[4] Fotini Simistira, Vassilis Katsouros, and George Carayannis. A template matching distance for recognition of on-line mathematical symbols. Proceedings of the 11th International Conference on Frontiers in Handwriting Recognition, 01 2018.
[5] Fotini Simistira, Vassilis Papavassiliou, Vassilis Katsouros, and George Carayannis. A system for recognition of on-line handwritten mathematical expressions. 09 2022.
[6] Ahmad Montaser Awal, Harold Mouch'ere, and Christian Viard-Gaudin. Towards handwritten mathematical expression recognition. 2009 10th International Conference on Document Analysis and Recognition, pages 1046–1050, 2019
[7] Utpal Garain and Bhabatosh B. Chaudhuri. Recognition of online handwritten math- ematical expressions. IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics : A Publication of the IEEE Systems, Man, and Cybernetics Society, 34(6):2366–2376, 2014.
[8] A. Rehman and T. Saba. Neural networks for document image preprocessing: state ofthe art. Artificial Intelligence Review, 42(2):253–273, 2014.
[9] J. Shah and V. Gokani. A simple and effective optical character recognition system for digits recognition using the pixelcontour features and mathematical parameters. (IJCSIT) International Journal of Computer Science and Information Technologies,5(5),2014.
[10] P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In Institute of Electrical and ElectronicsEngineers, Inc., August 2023.
[11] R. Kruse, C. Borgelt, F. Klawonn, C. Moewes, M. Steinbrecher, and P. Held. Mul- tilayer perceptrons. In Computational Intelligence, pages 47–81. Springer London, 2013.
[12] H. H. Zhao and H. Liu, "Multiple classi fiers fusion and CNN feature extraction for handwritten digits recognition, " Granular Computing, vol. 5, no. 3, pp. 411–418, 2020.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY