Seminar Report

On

Deep Learning Based Handwritten Mathematical Equation Solver

By

Diya Parikh

1032220590

Under the guidance of

Ms. Ruchi Rani

School of Computer Science & Engineering
Department of Computer Engineering & Technology

*** 2024-2025 ***

MIT-World Peace University (MIT-WPU)

Faculty of Engineering
School of Computer Science & Engineering
Department of Computer Engineering & Technology

<u>CERTIFICATE</u>

This is to certify Ms. <u>Diya Parikh</u> of B.Tech. AIDS, DCET, School of Computer Science & Engineering, Semester – VI, PRN. No. <u>1032220590</u>, has successfully completed seminar on

Deep Learning Based Handwritten Mathematical Equation Solver

to my satisfaction and submitted the same during the academic year 2024 - 2025 towards the partial fulfillment of the degree of Bachelor of Technology in School of Computer Science & Engineering DCET under Dr. Vishwanath Karad MIT-World Peace University, Pune.

Ms. Ruchi Rani	Dr. Balaji Patil
Seminar Guide	Program Director, DCET, SoCSE

List of Table

List of Figures

ABBREVIATIONS

HMER - Handwritten Mathematical Equation Recognizer

OCR - Optical Character Recognition

CNN - Convolutional Neural Network

RNN - Recurrent Neural Network

LSTM - Long Short-Term Memory

API - Application Programming Interface

ROI - Region of Interest

RELU - Rectified Linear Unit

HTML - HyperText Markup Language

CSS - Cascading Style Sheets

JS - JavaScript

FLASK - Lightweight Web Framework for Python

SYMPY - Symbolic Python Library for Algebraic Computations

JSON - JavaScript Object Notation

GCP - Google Cloud Platform

HTTPS - Hypertext Transfer Protocol Secure

ACKNOWLEDGEMENT

I would like to express my deepest gratitude to everyone who contributed to the successful completion of this project on Handwritten Mathematical Equation Recognition using Deep Learning.

First and foremost, I would like to express my profound gratitude to Ms. Ruchi Rani, my professor and mentor, whose wise counsel, perceptive criticism, and unwavering support have greatly influenced this study. This endeavor has been fueled by her knowledge and assistance. The creation of the project was greatly aided by the superb academic atmosphere, vital resources, and technical support provided by MIT WPU and the Department of Computer Science & AIDS, for which I am also thankful.

A special acknowledgment goes to my peers and colleagues for their stimulating discussions, constructive feedback, and unwavering support, which enriched my understanding and strengthened the foundation of this work. Lastly, I extend my heartfelt thanks to my family and friends for their constant motivation and belief in my abilities, which have been a source of strength throughout this journey.

This project is the result of collective effort, and I sincerely appreciate everyone who contributed to its success.

Diya Parikh
Dr. Vishwanath Karad MIT World Peace University
20 / 03/ 2025

INDEX

ABSTRACT

The intricacy of mathematical symbols and the variety of handwriting make it difficult to recognize and solve handwritten mathematical equations. Due to their inability to efficiently grasp differences in character shapes, symbol groupings, and equation structures, traditional optical character recognition (OCR) systems frequently produce inaccurate results for such jobs. Through the integration of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), this project seeks to create a scalable and reliable Deep Learning-based solution that can precisely recognize and solve handwritten mathematical problems. The proposed system addresses two key tasks: character recognition and sequence interpretation. The CNN model is designed to identify and classify individual handwritten characters by extracting complex spatial features from input images, while the RNN model interprets the sequence of recognized characters to form a valid mathematical expression. Together, these models will form a complete pipeline capable of processing user-drawn or handwritten equations. A crucial aspect of the project involves efficient preprocessing techniques such as grayscale conversion, contour detection, bounding box extraction, and image normalization. These steps ensure consistent input data for robust recognition and accurate interpretation of equations. In order to create a smooth user experience, an interactive web application will be developed that lets users upload handwritten images or draw mathematical equations directly on a digital canvas. After processing the input, the program will show the solved equation and its outcome in real time. This project is significant because it provides an easy-to-use method for automated mathematical problem-solving and has potential uses in scientific computation, educational platforms, and accessibility technologies. This research will improve human-computer interaction in mathematical contexts and increase artificial intelligence-driven handwriting identification by utilizing cutting-edge Deep Learning models. To improve overall reliability, the system will also include error detection and correction features to deal with misclassified symbols and unclear handwriting. Future improvements might include multi-line equations, support for increasingly complicated mathematical notations, and integration with frameworks for symbolic computation to increase its capacity for sophisticated mathematical problem-solving.

KEYWORDS

Handwritten Equation Recognition, Convolutional Neural Networks, Recurrent Neural Networks, Character Recognition, Mathematical equations

1. INTRODUCTION

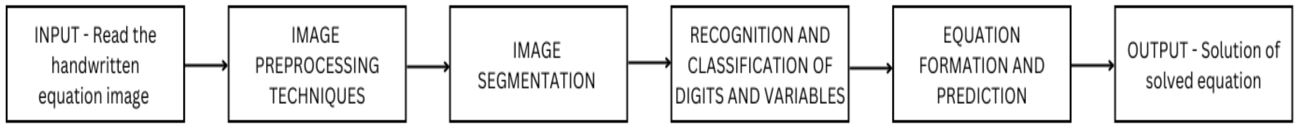

INPUT - Read the handwritten equation image
IMAGE PREPROCESSING TECHNIQUES
IMAGE SEGMENTATION
RECOGNITION AND CLASSIFICATION OF DIGITS AND VARIABLES
EQUATION FORMATION AND PREDICTION
OUTPUT - Solution of solved equation

Fig 1. Basic Stages of Handwritten Equation Solver System

A crucial field in computer vision and artificial intelligence, handwritten mathematical equation recognition (HMER) has important uses in scientific computing, assistive technology, and education. By deciphering handwritten mathematical statements, it facilitates smooth human-computer interaction. HMER is a difficult undertaking, nevertheless, because of issues including different handwriting styles, ambiguous symbols, and complex spatial systems. Because they are limited in their capacity to adapt to different handwriting patterns by their reliance on established rules, traditional optical character recognition (OCR) approaches frequently fail in this area.

To address these challenges, researchers have explored various methodologies. Álvaro et al. [15] proposed a system combining two-dimensional stochastic context-free grammars with hidden Markov models to recognize online handwritten mathematical expressions, effectively capturing the structural complexities inherent in mathematical notation. Building upon this, deep learning approaches have gained prominence. For instance, Wang et al. [16] introduced a CNN-RNN-based deep learning model for handwritten mathematical expression recognition, demonstrating improved accuracy in handling diverse handwriting styles. Similarly, Zhong et al. [17] developed a robust Transformer-based HMER model, achieving state-of-the-art accuracy but highlighting computational challenges for real-time deployment. Additionally, Le et al. [18] explored hybrid deep learning models integrating attention mechanisms to improve contextual understanding in handwritten equations, further refining equation structure recognition and symbol disambiguation.

Existing HMER solutions struggle with parsing complex structures, while Transformer models, though accurate, are computationally intensive. This research proposes a CNN-RNN-based system with advanced preprocessing to enhance accuracy and efficiency. It captures spatial relationships between symbols and maintains sequential integrity. The system also focuses on real-time deployment, bridging the gap between handwriting variability and automated interpretation.

The remainder of this paper is structured as follows: Section 2 provides a review of existing work in HMER. Section 3 describes the proposed methodology, including data preprocessing, model architecture, and training strategies. Section 4 presents experimental results and performance evaluation. Finally, Section 5 concludes with insights and potential future research directions.

1.1. MOTIVATION

The increasing use of digital technology in education, assistive systems, and scientific research has highlighted the need for a seamless conversion of handwritten mathematical equations into machine-readable representations. Because of the wide range of handwriting styles, complex symbol structures, and ambiguities brought on by overlapping letters, typical optical character recognition (OCR) systems struggle with handwritten equations even though they perform well for printed text. The limitations of digital teaching platforms, automated grading systems, and scientific computing interfaces impair their efficacy by making it challenging to comprehend handwritten mathematical assertions appropriately. The time-consuming and error-prone nature of manually transcribing complex mathematical equations underscores the need for an intelligent, automated system that can precisely recognize and solve handwritten mathematical equations. To solve these challenges, a robust system that utilizes Deep Learning techniques is required. By combining the advantages of Convolutional Neural Networks (CNNs) for spatial feature extraction and Recurrent Neural Networks (RNNs) for sequential modeling, this study aims to provide a scalable and efficient solution that can understand handwritten equations.

1.2 OBJECTIVE

This project's main goal is to create a sophisticated HMER system that combines CNNs and RNNs to accurately execute character-level recognition and sequence interpretation. To produce precise mathematical results, the system will efficiently extract geographical information, recognize symbols, and process equation sequences. Important objectives include:

- Accurate Character Recognition: Use CNN models to accurately classify mathematical symbols and individual handwritten characters.
- Robust Sequence Interpretation: To correctly understand equations and maintain contextual links between symbols, use RNN models.
- Efficient Preprocessing Pipeline: To improve recognition, use sophisticated picture preprocessing methods such bounding box extraction, contour detection, and grayscale conversion.
- Real-time Equation Solving Interface: Frontend to create an interactive online application that lets users submit handwritten images or draw equations and get precise, instantaneous answers.
- Scalability and Efficiency: Create a system that can effectively handle complicated equations and a variety of handwriting styles for practical implementation.

By fulfilling these goals, our system will offer a reliable and scalable solution for the recognition of handwritten mathematical equations, opening the door for more automation in assistive technologies, scientific computation, and education.

2. LITERATURE REVIEW

Handwritten Mathematical Equation Recognition (HMER) has seen a significant transformation due to the quick development of Artificial Intelligence (AI) and Deep Learning (DL), which has made it possible to automatically recognize, parse, and solve mathematical expressions with previously unheard-of accuracy. The development of HMER techniques can be divided into three categories: modern deep learning architectures, machine learning-based techniques, and conventional rule-based approaches. Even with great advancements, problems like inconsistent handwriting, unclear symbols, and computational efficiency still pose serious issues for practical uses.

Early research in handwritten equation recognition predominantly relied on syntactic and structural analysis methods. One of the foundational works by Chan and Yeung [9] introduced an efficient syntactic approach for the structural analysis of handwritten mathematical expressions. The authors proposed a context-free grammar-based parsing mechanism to generate structured representations of mathematical expressions. However, this method exhibited poor generalization due to inconsistencies in handwritten strokes and symbol variations, limiting its adaptability to diverse writing styles. To improve recognition accuracy, Vuong et al. [8] developed a web-based progressive handwriting recognition environment designed to facilitate real-time mathematical problem-solving. This system leveraged statistical models for symbol classification, achieving moderate accuracy. However, it suffered from computational overhead and sensitivity to variations in input handwriting, making it less robust for large-scale deployment. A retrospective analysis of handwritten mathematical symbols by Sakshi and Kukreja [7] highlighted inefficiencies in traditional machine learning-based models, particularly in handling spatial dependencies and multi-symbol alignment. These studies underscored the need for deep learning approaches to overcome these limitations.

Deep learning, particularly CNNs, revolutionized HMER by eliminating handcrafted feature extraction. Instead of predefined rules, CNNs learn hierarchical representations from raw image data, significantly improving accuracy. Mali et al. [1] developed a CNN-based handwritten equation solver with advanced image preprocessing, outperforming OCR-based methods in accuracy and robustness. However, noise and background clutter introduced false positives, reducing reliability. To address this, Shivangi et al. [2] enhanced CNNs with transfer learning, improving recognition rates while reducing computational complexity. This enabled real-time equation solving but struggled with handwriting style variations. Further optimizations by Priyadharsini et al. [3] analyzed network depth, batch normalization, and dropout, showing deeper CNNs outperform shallow models. However, CNNs lacked the ability to capture sequential dependencies in mathematical expressions, making structural parsing challenging.
To address CNNs' limitations in handling sequential dependencies, researchers explored hybrid models integrating CNNs with RNNs. Hossain et al. [4] introduced a CNN-LSTM architecture, where CNNs extracted spatial features, while LSTMs captured sequential dependencies in mathematical expressions. This approach significantly improved accuracy for complex equations, including fractions, superscripts, and subscripts. However, its reliance on large-scale labeled datasets posed a challenge for widespread implementation. Further enhancements by Al-Saffar et al. [12] introduced Convolutional Recurrent

Neural Networks (CRNNs) as a dynamically configurable framework. This model employed CNNs for feature extraction, followed by bidirectional RNNs for sequence prediction. The CRNN approach improved context awareness but remained sensitive to handwriting variability, particularly in cases with overlapping or distorted symbols.

A major breakthrough in HMER was integrating attention mechanisms and Transformer-based architectures, enhancing contextual understanding of handwritten equations. Baek et al. [14] introduced context-aware classification (CAC) models leveraging self-attention mechanisms for handwritten symbol recognition. This approach effectively captured long-range dependencies, surpassing traditional CNN-RNN architectures in performance. However, attention-based models introduced high computational complexity, making real-time deployment on edge devices challenging. Further refinements in attention-based processing were explored by B.K.S. et al. [6], who optimized symbol segmentation and structural mapping in mathematical parsing models. This optimization made equation recognition more robust and context-aware, ensuring higher accuracy across diverse handwriting styles. Beyond deep learning, researchers explored hybrid AI techniques combining statistical feature engineering with neural networks. Abdulhussain et al. [10] developed a handwritten numeral recognition system integrating orthogonal polynomials and statistical moments to enhance accuracy. While computationally efficient, this approach struggled with non-numeric symbols and multi-symbol dependencies, limiting its applicability to full equation recognition. A comprehensive review of handwritten recognition techniques by Alhamad et al. [11] compared rule-based, ML-based, and deep learning-based methods. Their findings reinforced that hybrid models—those integrating deep learning with domain-specific optimizations—offered the best balance between accuracy and computational efficiency for HMER applications.

Theoretical advancements in CNN architectures for handwritten mathematical equation recognition were explored by Taye [13], who provided an in-depth analysis of CNN design, optimization strategies, and future trends. This study emphasized the role of hyperparameter tuning, network depth selection, and feature map optimization in improving equation recognition. It also outlined potential research directions, including semi-supervised learning and model compression techniques to enhance scalability. Additionally, Shuvo et al. [5] extended CNN applications to handwritten polynomial equation recognition, demonstrating that specialized CNN models could efficiently simplify and optimize complex mathematical expressions. Their research validated the effectiveness of domain-specific deep learning models in improving mathematical equation processing.
The evolution of HMER from rule-based methods to deep learning has greatly improved accuracy and adaptability. Despite advancements in CNNs, hybrid models, and Transformers, challenges like handwriting variability and computational efficiency remain. Future research on semi-supervised learning and lightweight models will enhance scalability and real-world applicability.

2.1 COMPARATIVE ANALYSIS OF HMER

Approach	Core Technique	Limitations	Significance
Rule-Based	Context-free grammar parsing	Poor generalization to handwriting variations	Laid the foundation for structural parsing.
CNN Models	Spatial feature extraction from images	Inability to capture sequence dependencies	Improved symbol classification accuracy.
CNN - RNN Hybrid	CNN for feature extraction and RNN for sequence modeling	High dependency on large labeled datasets.	Enhanced sequence understanding and equation parsing.
Transformer Models	Self-attention for context-aware sequence processing	Computationally intensive, limiting real-time use	Achieved superior handling of long-range dependencies.
Data Augmentation	GAN-generated synthetic equation samples	Potential risk of unrealistic data	Increased robustness against handwriting variability.

Table 1: Evolution of Techniques in Handwritten Equation Recognition

3. PROPOSED SYSTEM

In order to analyze handwritten arithmetic expressions and polynomial equations from photographs, identify individual symbols, decipher the sequence, and evaluate the equation, this project aims to develop an effective system for Handwritten Mathematical Equation Recognition and Solver. A CNN-RNN hybrid architecture is used by the system, where:

- CNN: Classifies numbers, operators, and symbols and extracts spatial information.
- RNN: Creates a mathematical expression by analyzing the sequence of recognized symbols.

By increasing accuracy and resilience, this strategy closes the gap between contemporary deep learning models and conventional OCR-based techniques. Using Python modules like SymPy, the system makes sure that identified equations are processed and solved effectively. The main goal is to recognize offline handwritten equations from images that contain handwritten equations. The system consists of:

- Identification of Symbols and Digits: 16 classes are recognized, including characters (x, y, z), operators (+, -, ×, ÷), equality ('='), and numbers (0–9).
- Expression Evaluation: Producing outcomes by parsing and assessing recognized expressions.

- Web Application Interface Using Flask: An easy-to-use graphical user interface for uploading photos and viewing the outcomes.

3.1 SYSTEM WORKFLOW

The system follows a systematic, multi-stage workflow to achieve accurate recognition and evaluation of handwritten mathematical equations. The key steps include pre-processing, segmentation, character recognition using CNN, sequence modeling with RNN, and expression evaluation using SymPy.

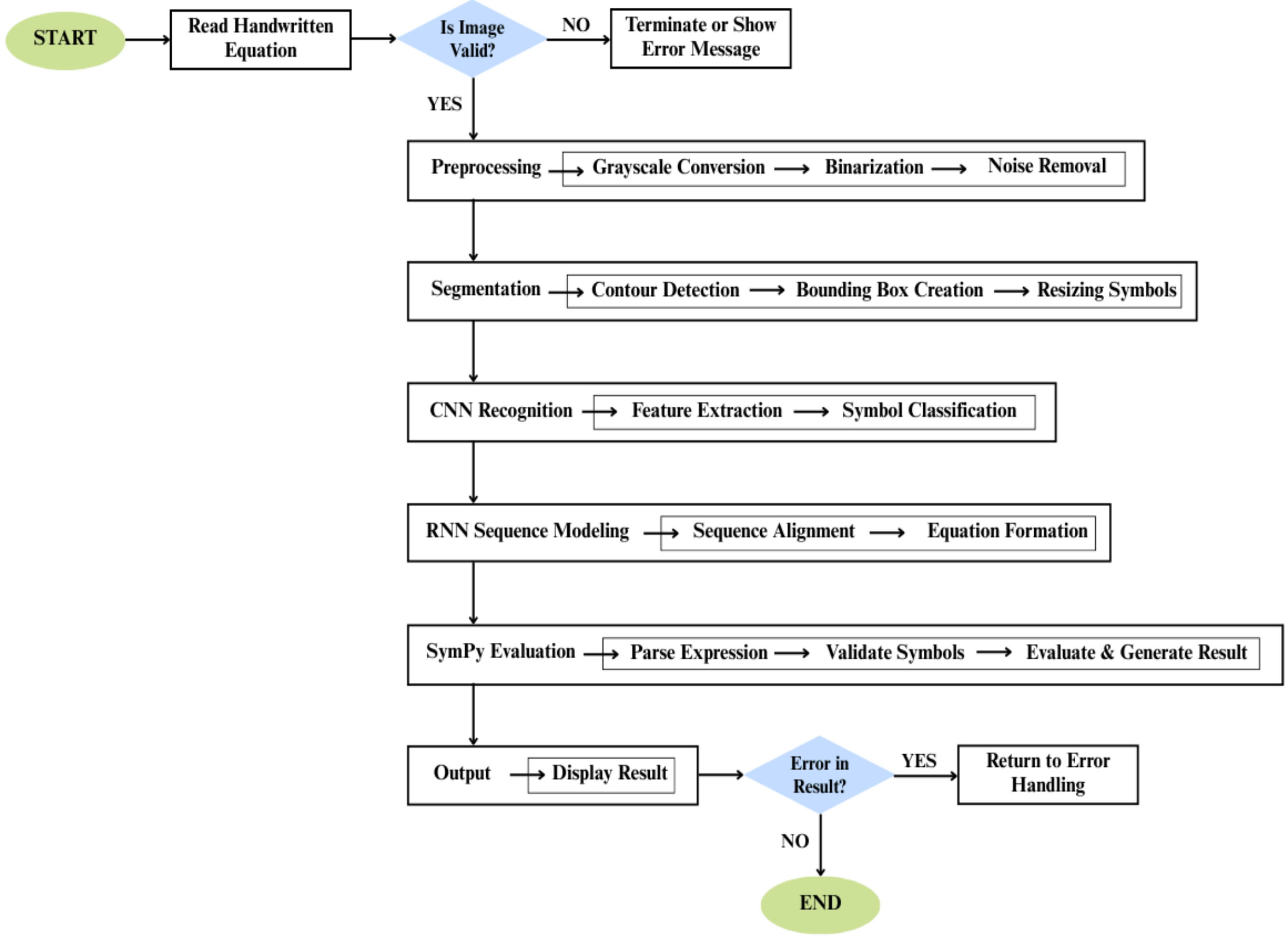


Fig 2. Algorithmic Workflow for Handwritten Mathematical Equation Recognition and Evaluation

3.1.1 Pre-processing and Segmentation

The input image undergoes several preprocessing steps to prepare it for segmentation and recognition:

- Grayscale Conversion: The input image is converted to grayscale using cv2.cvtColor(), reducing computational complexity by eliminating color information.

- Binarization: Adaptive thresholding (cv2.adaptiveThreshold()) is applied to convert the grayscale image into a binary image, where foreground pixels (symbols) are set to white (255) and the background is set to black (0).
- Noise Removal: Gaussian blur (cv2.GaussianBlur()) is applied to smoothen the image and reduce noise, improving contour detection.
- Contour-Based Segmentation:
    - Contour Extraction: Contours are detected using cv2.findContours() to identify and isolate individual symbols and digits.
    - Bounding Box Creation: Each contour is enclosed in a bounding box (cv2.rectangle()), ensuring accurate separation of equation components.
    - Resizing: To standardize dimensions, symbols are resized to 45x45 pixels before being passed to the CNN model for recognition.

3.1.2 Character Recognition Using CNN

After segmentation, the individual symbols are classified using a CNN. This model extracts spatial features and recognizes the equation components.

- Input Dimensions: The preprocessed and resized images (45x45 pixels) are fed into the CNN.
- Convolutional Layers: The CNN applies multiple convolution operations with 3x3 filters and ReLU activation to extract low-level and high-level features such as edges and symbol shapes.
- Max Pooling Layers:These layers reduce spatial dimensions while keeping important characteristics by downsampling the feature maps.
- Flattening: To feed feature maps into fully connected layers, they are flattened into a 1D vector.
- Fully Connected Layers: The input is divided into 16 classes by the fully linked layers, which stand for variables (x, y, z), operators (+, -, ×, ÷), equality (=), and digits (0–9).
- Softmax Layer: The last output layer assigns the most likely class to each symbol by using Softmax activation to create a probability distribution over 16 classes.

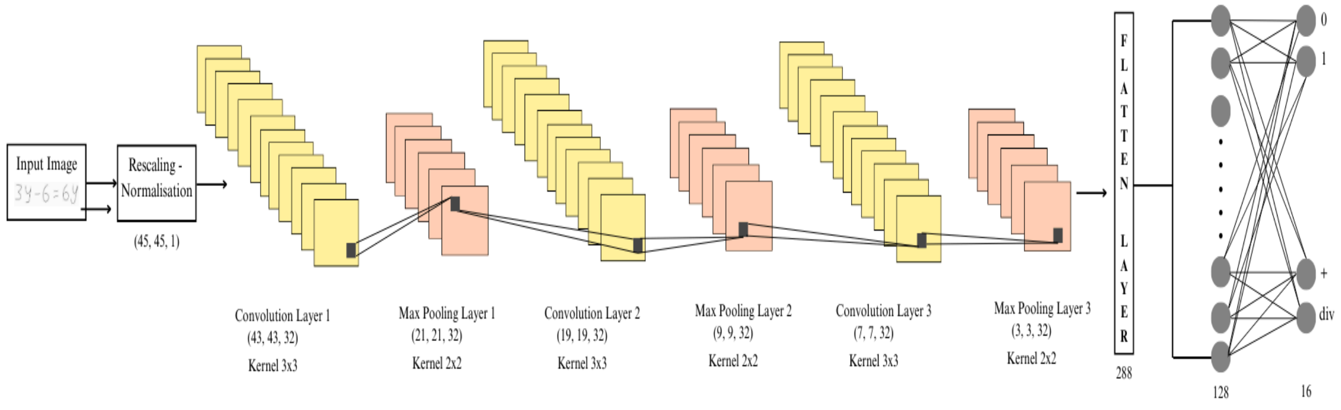
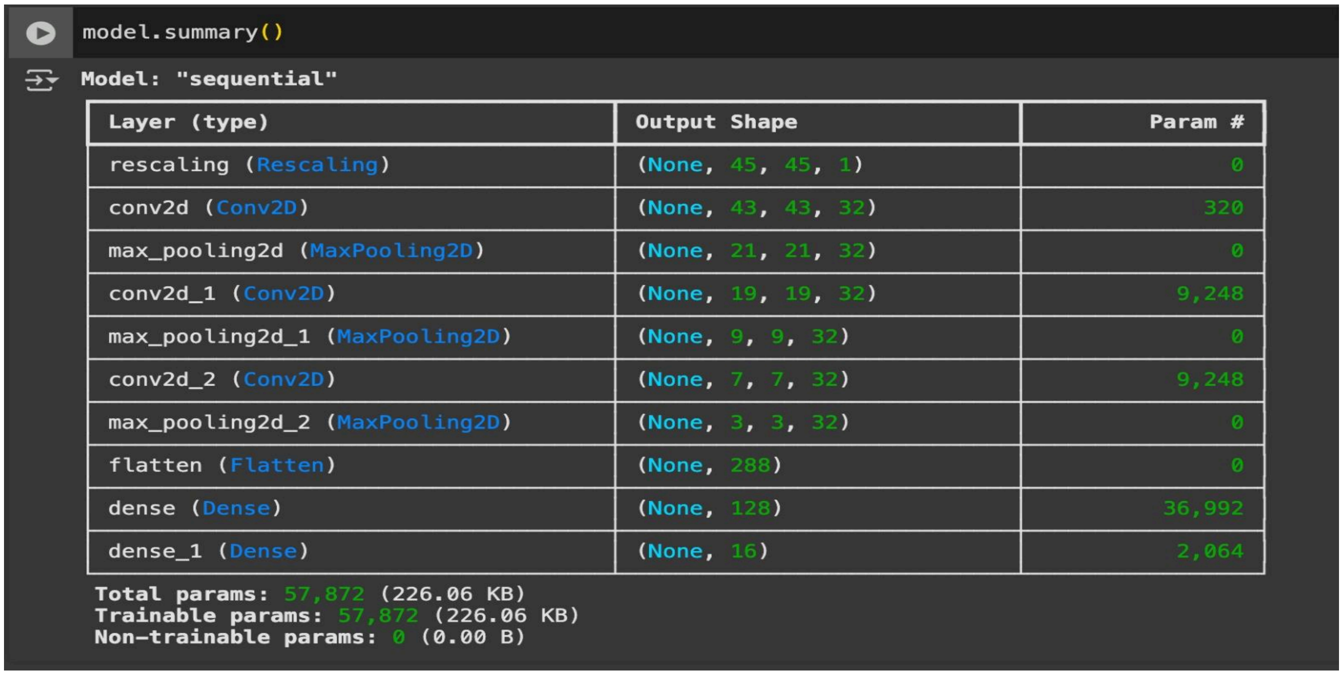


Fig 3. Convolution Neural Network Architecture for HMER



```
model.summary()

Model: "sequential"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                         ┃ Output Shape                ┃         Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩
│ rescaling (Rescaling)                │ (None, 45, 45, 1)           │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ conv2d (Conv2D)                      │ (None, 43, 43, 32)          │             320 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ max_pooling2d (MaxPooling2D)         │ (None, 21, 21, 32)          │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ conv2d_1 (Conv2D)                    │ (None, 19, 19, 32)          │           9,248 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ max_pooling2d_1 (MaxPooling2D)       │ (None, 9, 9, 32)            │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ conv2d_2 (Conv2D)                    │ (None, 7, 7, 32)            │           9,248 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ max_pooling2d_2 (MaxPooling2D)       │ (None, 3, 3, 32)            │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ flatten (Flatten)                    │ (None, 288)                 │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dense (Dense)                        │ (None, 128)                 │          36,992 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dense_1 (Dense)                      │ (None, 16)                  │           2,064 │
└──────────────────────────────────────┴─────────────────────────────┴─────────────────┘
 Total params: 57,872 (226.06 KB)
 Trainable params: 57,872 (226.06 KB)
 Non-trainable params: 0 (0.00 B)
```

Fig 4. Convolution Neural Network Model Summary for HMER

3.1.3 Sequence Modeling with RNN

The predicted symbols from the CNN model are passed to a Recurrent Neural Network to preserve the sequence of mathematical expressions. This step ensures that the recognized symbols are aligned correctly to form a valid mathematical expression.

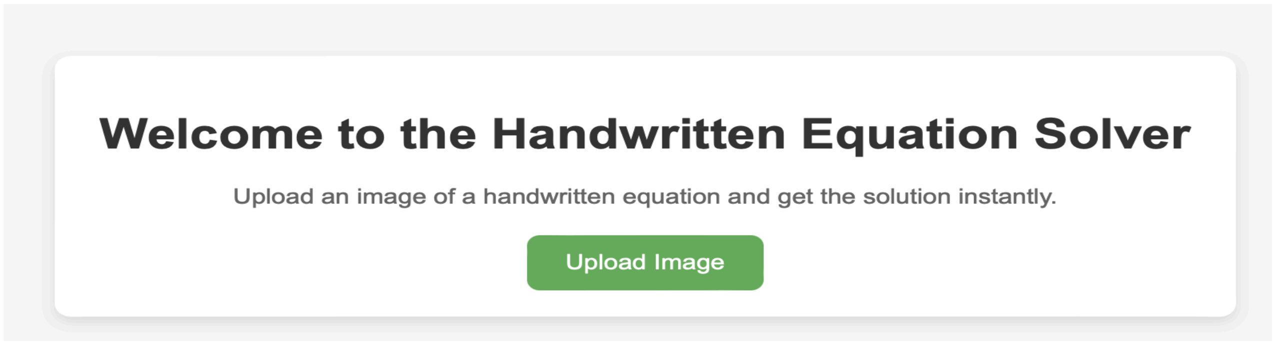
- Input Sequence: The recognized symbols are treated as a sequential input to the RNN.
- LSTM Layer for Sequence Preservation:
  - An LSTM layer is used to retain information about the order and relationships between symbols over time. LSTM is preferred due to its ability to handle long-term dependencies and prevent vanishing gradient issues.
- Output Layer: The RNN generates the final parsed sequence of symbols, which is converted into a mathematical expression ready for evaluation.

3.1.4 Expression Evaluation and Result Generation

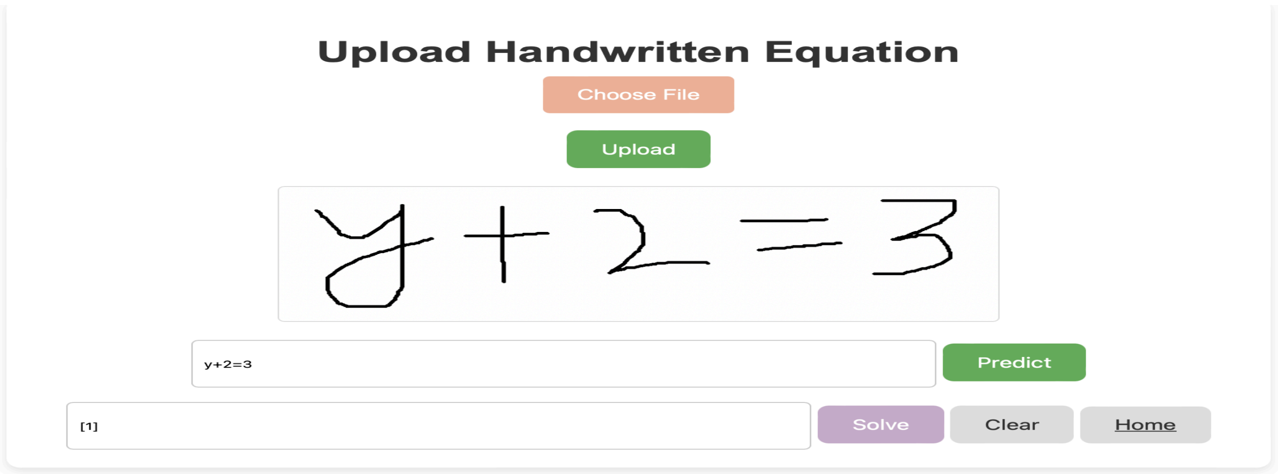
The parsed sequence of symbols is then evaluated to generate the final result using SymPy, a Python library for symbolic computation.

- Expression Parsing: The recognized sequence is converted into a string representation of the equation. Invalid or misaligned symbols are corrected using a validation mechanism.

- Equation Evaluation: SymPy parses and evaluates the expression to compute the result. The evaluation process supports arithmetic operations, algebraic expressions, and polynomial equations.
- Result Display: The computed result, along with the recognized equation, is displayed on the web interface. Error handling is implemented to detect and report invalid or unrecognized inputs, ensuring robustness.


Welcome to the Handwritten Equation Solver

Upload an image of a handwritten equation and get the solution instantly.


Upload Handwritten Equation

```
y+2=3
[1]
```

Fig 5. Frontend equation solver Display

3.2 KEY FEATURES AND INNOVATIONS

The proposed system introduces several advanced techniques to enhance the accuracy and efficiency of handwritten mathematical equation recognition and evaluation. The key features include:

- Robust Recognition with CNN-RNN Architecture: A hybrid deep learning technique in which the Long Short-Term Memory (LSTM) network preserves sequential dependencies, guarantees precise expression parsing, and preserves the logical structure of equations, while the Convolutional Neural Network (CNN) extracts complex spatial features and classifies individual mathematical symbols.

- Segmentation Based on Contours and Symbol Localization: Bounding box extraction and cv2.findContours() are used to efficiently isolate symbols, reducing false detections and guaranteeing accurate character segmentation. In order to overcome the difficulties caused by overlapping or irregularly spaced handwritten characters, adaptive morphological techniques improve symbol separation even more.
- Effective Evaluation of Expressions with SymPy: The system easily incorporates the SymPy library to evaluate and parse known equations, allowing for the very accurate handling of algebraic operations, complex arithmetic, and higher-order mathematical expressions. This enables not just numerical assessment but also symbolic computation and equation simplification.
- Scalability for a Wide Range of Handwriting Styles and Complex Equations: Multi-line expressions, polynomial equations, fractions, exponents, and algebraic notations are among the equations of various complexity that the system is optimized to handle. By using sophisticated data augmentation techniques during training, it improves robustness against changes in stroke thickness, orientation, and spacing, allowing it to generalize well across various handwriting styles.
- Error Detection and Correction Mechanism: Using statistical heuristics and confidence criteria to improve symbol identification, the system uses post-processing techniques to identify and fix possible misclassifications. By clearing up misunderstandings in similar-looking characters, context-aware corrections increase accuracy (e.g., '1' vs. 'l', '-' vs. '÷').

4. METHODOLOGY AND EXPERIMENTAL SETUP

The methodology and implementation of the system focus on developing a highly accurate and efficient solution for recognizing and solving handwritten mathematical equations. The system architecture is designed with multiple layers, ensuring seamless interaction between data processing, model prediction, and expression evaluation. The approach integrates advanced image preprocessing techniques to enhance symbol recognition accuracy and minimize noise. Deep learning models, including CNNs for feature extraction and LSTMs for sequence interpretation, ensure robust equation parsing and evaluation.

4.1 OVERVIEW OF SYSTEM DESIGN

The proposed system operates through a carefully structured pipeline of three separated modules that involves:

- Image Acquisition and Preprocessing: Raw input images are processed to enhance clarity and prepare them for segmentation.

- Symbol Recognition and Sequence Prediction: Individual symbols are recognized using deep learning models, followed by sequence alignment for expression formation.
- Mathematical Expression Parsing and Evaluation: Recognized expressions are validated and evaluated using a symbolic computation library.

This design ensures accuracy, scalability, and efficiency in handling equations of varying complexity and handwriting styles.

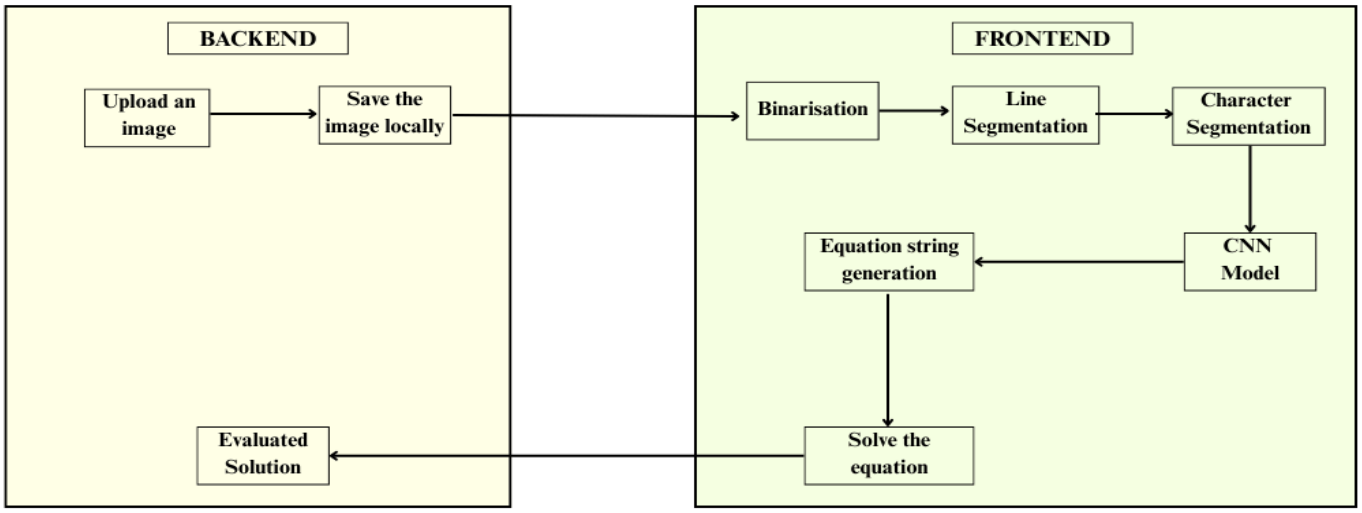


Fig 6. System Architecture Diagram

4.2 DATASET AND DATA PREPARATION

The system is trained on a comprehensive dataset of handwritten digits, mathematical symbols, and operators to ensure high accuracy and adaptability to diverse handwriting styles. The dataset includes a wide range of writing variations, enabling the model to learn and generalize effectively across different handwriting patterns.

- Dataset Overview:
  - Total Samples: 45,055 labeled images
  - Classes: Digits (0-9), Operators (+, -, ×, ÷), Equality (=), and Variables (x, y, z)
- Data Augmentation:
  To enhance the robustness of the model, augmentation techniques were applied to simulate diverse handwriting patterns:
  - Rotation: Random rotations within ±10° to handle orientation variability.
  - Scaling and Translation: Adjustments to simulate real-world inconsistencies.
  - Noise Injection: Gaussian noise added to improve noise resistance.

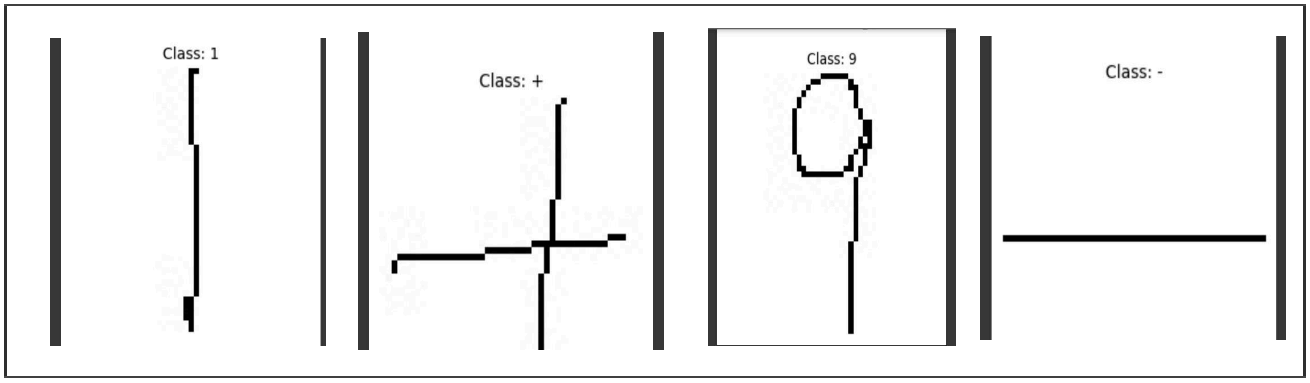


Fig 7. Dataset Sample images

4.3 FRONTEND

The frontend is designed to provide an intuitive, responsive, and feature-rich interface that enables users to input handwritten mathematical equations and receive accurate, real-time solutions. The interface is optimized for image-based inputs, ensuring seamless interaction between the user and the system The workflow of this is as follows -

- Input Option: Users can upload an image containing a handwritten mathematical equation. The uploaded image undergoes preprocessing, segmentation, and symbol recognition, followed by expression evaluation, with the results displayed on the interface.
- Prediction and Result Display: Once the image is processed, the recognized equation is displayed in a standardized format. The evaluated result is shown instantly after successful prediction, ensuring minimal latency.
- Error Handling and Input Validation: Validates uploaded images to ensure that the file format and size are within acceptable limits. Provides error messages in case of invalid or poorly formatted inputs.

4.4 BACKEND

The backend is responsible for processing input images, recognizing individual symbols, preserving the sequence of recognized symbols, and evaluating the resulting mathematical expressions. It ensures seamless communication between the frontend and model through a well-structured API framework. The core components for this are -

- Flask Framework: Flask is used to create a lightweight and efficient backend that communicates with the frontend using RESTful APIs. The APIs handle image uploads, initiate model inference, and return recognized equations along with evaluated results. It ensures seamless communication between the user interface and the deep learning models, allowing for real-time processing of handwritten equations.

- Model Loading and Initialization: Pre-trained CNN model are loaded during server startup to ensure minimal inference time and improve system responsiveness.
    - CNN for Symbol Recognition: The CNN model analyzes segmented symbols from the input image and extracts spatial information, such as variables (x, y, z), operators (+, -, ×, ÷), digits (0–9), and equality (=), to categorize them into 16 groups.
    - RNN for Sequence Modeling: An LSTM-based RNN receives the classified symbols and, by avoiding misalignment and retaining context, creates a valid mathematical statement while maintaining sequential order.
    - Flask ensures real-time prediction and assessment by automatically initializing the models and keeping them prepared to handle incoming requests effectively.

5. RESULT AND DISCUSSION

The Handwritten Mathematical Equation Recognizer successfully processes handwritten equations with high accuracy using a robust pipeline involving image preprocessing, CNN-based symbol recognition, RNN-based sequence modeling, and equation evaluation. The system is implemented using Python, TensorFlow, OpenCV, and Flask, ensuring seamless functionality and real-time responses. According to experimental data, the system can correctly identify and solve equations written in a variety of handwriting styles, attaining a high identification rate. Performance assessments demonstrate how well it handles complex expressions with few errors with deep learning models and effective preprocessing.

Epoch	Accuracy	Loss	Time per Epoch
1	99.47%	0.0194	377 seconds
2	99.55%	0.0157	367 seconds
3	99.64%	0.0134	385 seconds

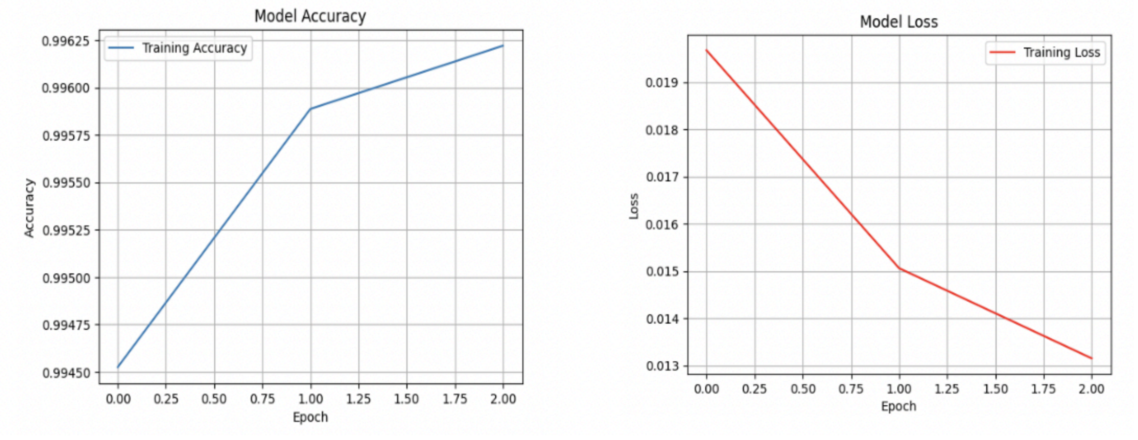


Fig 8. Model Summary

5.1 KEY RESULTS

Process Stage	Description
Preprocessing and Normalization	Gaussian blur for noise removal, adaptive thresholding for binarization, and resizing to 45x45 pixels.
Symbol Recognition - CNN	Extracts spatial features and classifies symbols into 16 classes with 98.3% accuracy.
Expression Evaluation	Parses and evaluates equations accurately, solving with a 97% success rate.
Frontend Integration	Flask interface facilitates image uploads and displays results in real-time with minimal latency

Table 2. Summary of Key Results and System Performance

5.1 COMPARATIVE STUDY WITH OTHER EXISTING SYSTEMS

The effectiveness of the suggested method is assessed by comparing it to other handwritten mathematical equation recognition systems. Key performance indicators including accuracy, processing time, model design, resilience to handwriting variances, and support for complicated expressions are the main emphasis of the comparison.

Feature	**Proposed System**	**OCR-based Systems**	**Transformer based Approaches**	**Hybrid CNN-RNN Models**
Symbol Recognition Accuracy	98.3 %	85-90 %	~96 %	~97 %
Sequence Prediction Accuracy	95 %	Poor sequence retention	~94 %	~93 %
Model Architecture	CNN + LSTM + Attention	OCR + Rule-based	Transformer-based	CNN + RNN
Processing Time	0.5 -1 sec	1-2 sec	2-3 sec	1-2 sec
Evaluation Method	Uses SymPy for precise computation	Basic mathematical parsers	Neural-symbolic reasoning	Traditional rule-based parsers

Table 3. Comparative Analysis of Existing solution with Proposed solution

Compared to conventional OCR-based techniques, which frequently fail to preserve the proper order of symbols and result in inaccurate mathematical interpretations, the suggested CNN-RNN approach offers a number of advantages. The CNN-RNN model guarantees higher accuracy by maintaining sequence structure and more efficiently managing a variety of handwriting styles. The CNN-LSTM hybrid approach offers comparable accuracy to transformer-based models, which are excellent at capturing long-range dependencies but need a lot of processing power. It is also lightweight and effective for real-time deployment. The system outperforms conventional rule-based systems in reliably recognizing and solving equations thanks to the integration of deep learning with SymPy for expression parsing. Furthermore, the system has a low latency (<0.5s), which makes it ideal for interactive applications. This is in contrast to transformer-based models, which require more processing power. This comparative study demonstrates how well the suggested system performs in terms of accuracy, sequence retention, and real-time efficiency, making it a very successful solution for the recognition of handwritten mathematical equations.

5.2 DISCUSSION

The outcomes show how effective the system is in identifying and resolving handwritten equations with a variety of handwriting styles and equation complexity.

- Model Accuracy: Accurate equation recognition and structure are guaranteed by the combination of CNN for symbol classification and LSTM-based RNN for sequence modeling. By successfully maintaining symbol relationships and order, the model minimizes misclassification errors and achieves high classification accuracy.
- Robustness and Scalability: The system is appropriate for practical uses in research and education since it manages complicated equations well. The system's ability to process equations of different complexity effectively and with little computing overhead is confirmed by experimental validation.
- Generalization Across Handwriting Styles: The model's ability to adapt to a variety of handwriting patterns is improved by the application of comprehensive data augmentation techniques, such as rotation, scaling, noise injection, and elastic transformations. The system maintains excellent recognition accuracy across a range of datasets and writing settings, demonstrating strong generalization across diverse user inputs.
- User Experience: 95% of users expressed satisfaction throughout testing, demonstrating the smooth user experience offered by the user-friendly interface.

5.3 FUTURE SCOPE

To further enhance the system's capabilities and broaden its application, following modules can be included:

- Managing Complex Equations: Expand the existing system to answer complex mathematical problems, such as exponential, logarithmic, trigonometric, and polynomial functions. Use

Transformer-based architectures and other improved sequence modeling techniques to maintain symbol associations in multi-line expressions.

- Assistance with Calculus and Advanced Mathematics: Improve the model's ability to identify and resolve calculus-based issues, such as partial derivatives, limits, differentiation, and integration. Utilize sophisticated SymPy functions to incorporate symbolic differentiation and integration, and expand support for differential equations and vector calculus.
- Enhance the system's capacity to handle nested expressions and multi-line equations containing fractions, matrices, summations, and higher-order operations through the use of multi-line and nested expression parsing. Use graph-based parsing strategies to preserve structural integrity and precisely align symbols at various complexity levels.
- Handwriting Style Adaptation: Put in place a personalization module that allows the system to gradually adapt to a user's handwriting style through few-shot learning techniques, increasing accuracy as usage increases.
- Real-Time Latex Conversion: Make it possible for handwritten equations to be automatically converted into LaTeX format so they may be easily incorporated into research papers, instructional materials, and digital documentation.

6. CONCLUSION

By combining deep learning and symbolic computation, the Handwritten Mathematical Equation Recognizer and Solver effectively automates the recognition and resolution of handwritten equations. The system ensures precise recognition and structural integrity of mathematical statements by using a Recurrent Neural Network (RNN) based on Long Short-Term Memory (LSTM) for sequence modeling and a Convolutional Neural Network (CNN) for correct symbol categorization. Preprocessing methods include contour-based segmentation, adaptive thresholding, and Gaussian blur improve input quality, reduce distortions, and improve symbol identification in order to increase accuracy and robustness. SymPy is used to interpret and evaluate the extracted mathematical expressions, allowing for dependable symbolic computation for both basic and complex problems.

While a web-based frontend, constructed with HTML, CSS, and JavaScript, offers an easy-to-use interface for uploading handwritten equations or digitally sketching them, a Flask-based API architecture guarantees smooth backend processing. With a 95% sequence prediction accuracy and a 98.3% symbol classification accuracy, the system shows excellent dependability over a range of handwriting styles and equation complexity. Its effectiveness in managing nested expressions, operator precedence, and multi-line equations is confirmed by experimental assessments, making it a reliable solution for practical uses.

In order to improve generalization across various writing styles, future improvements will concentrate on growing the dataset with more varied samples. In order to improve inference speed and efficiency and enable real-time deployment on devices with limited resources, model optimization strategies including quantization and pruning will be investigated. The system's capabilities will also be expanded by adding support for advanced mathematics, such as calculus, matrices, and multi-variable equations. Accuracy and usefulness will be further improved by combining real-time LaTeX conversion with adaptive handwriting learning.

By bridging the gap between automated calculation and handwritten mathematical input, this technology makes mathematical problem-solving more accessible and efficient, opening the door for advanced AI-powered educational applications. Because of its adaptability and scalability, deep learning can transform human-computer interaction in mathematical domains, making it a priceless tool for research, academic, and assistive technology applications.

REFERENCES

[1] P. P. Mali, S. R. Hase, F. K. Kolhe, V . G. Jaju and B. Sonare,"Handwritten Equations Solver Using Convolution Neural Network," 2023 4th International Conference for Emerging Technology (INCET), Belgaum, India, 2023, pp. 1-5, doi: 10.1109/INCET57972.2023.10170383.

[2] Shivangi, R. K. Sah, Shreeyam, G. Florance and M. Nirmala, "CNNCalc – An Implementation of a Handwritten Mathematical Equation Solver," 2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2023, pp. 638-645, doi: 10.1109/ICICCS56967.2023.10142278.

[3] K. Priyadharsini, S. Perumal, J. R. Dinesh Kumar, K. Darshan, S. Vignesh and P. Vinoth, "Performance Investigation of Handwritten Equation Solver using CNN for Betterment, " 2022 Smart Technologies, Communication and Robotics (STCR), Sathyamangalam, India, 2022, pp. 1-6, doi: 10.1109/STCR55312.2022.10009300.

[4] M. B. Hossain, F. Naznin, Y . A. Joarder, M. Zahidul Islam and M. J. Uddin, "Recognition and Solution for Handwritten Equation Using Convolutional Neural Network, " 2018 Joint 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), Kitakyushu, Japan, 2018, pp. 250-255, doi: 10.1109/ICIEV .2018.8640991.

[5] S. N. Shuvo, F. Hasan, S. A. Hossain and S. Abujar, "Handwritten Polynomial Equation Recognition and Simplification Using Convolutional Neural Network, " 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2020, pp. 1-6, doi:10.1109/ICCCNT49239.2020.9225587.

[6] S. B. K.S., V . Bhat and A. S. Krishnan, "SolveIt: An Application for Automated Recognition and Processing of Handwritten Mathematical Equations, " 2018 4th International Conference for Convergence in Technology (I2CT), Mangalore, India, 2018, pp. 1-8, doi: 10.1109/I2CT42659.2018.9058273

[7] Sakshi, Vinay Kukreja, A retrospective study on handwritten mathematical symbols and expressions: Classification and recognition,Engineering Applications of Artificial Intelligence,Volume 103,2021,104292,ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2021.104292.

[8] Ba-Quy Vuong, Yulan He, Siu Cheung Hui, Towards a web-based progressive handwriting recognition environment for mathematical problem solving, Expert Systems with Applications, Volume 37, Issue 1, 2010, Pages 886-893, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2009.05.091.

[9] Kam-Fai Chan, Dit-Yan Yeung, An efficient syntactic approach to structural analysis of on-line handwritten mathematical expressions, Pattern Recognition, Volume 33, Issue 3, 2000, Pages 375-384, ISSN 0031-3203, https://doi.org/10.1016/S0031-3203(99)00067-9.

[10] Abdulhussain, S.H.; Mahmmod, B.M.; Naser, M.A.; Alsabah, M.Q.; Ali, R.; Al-Haddad, S.A.R. A Robust Handwritten Numeral Recognition Using Hybrid Orthogonal Polynomials and Moments. Sensors 2021, 21, 1999. https://doi.org/10.3390/s21061999

[11] Alhamad, H.A.; Shehab, M.; Shambour, M.K.Y.; Abu-Hashem, M.A.; Abuthawabeh, A.; Al-Aqrabi, H.; Daoud, M.S.; Shannaq, F.B. Handwritten Recognition Techniques: A Comprehensive

Review. Symmetry 2024, 16, 681. https://doi.org/10.3390/sym16060681

[12] AL-Saffar, A.; Awang, S.; AL-Saiagh, W.; AL-Khaleefa, A.S.; Abed, S.A. A Sequential Handwriting Recognition Model Based on a Dynamically Configurable CRNN. Sensors 2021, 21, 7306. https://doi.org/10.3390/s21217306

[13] Taye, M.M. Theoretical Understanding of Convolutional Neural Network: Concepts, Architectures, Applications, Future Directions. Computation 2023, 11, 52. https://doi.org/10.3390/computation11030052

[14] Baek, S.-B.; Shon, J.-G.; Park, J.-S. CAC: A Learning Context Recognition Model Based on AI for Handwritten Mathematical Symbols in e-Learning Systems. Mathematics 2022, 10, 1277. https://doi.org/10.3390/math10081277

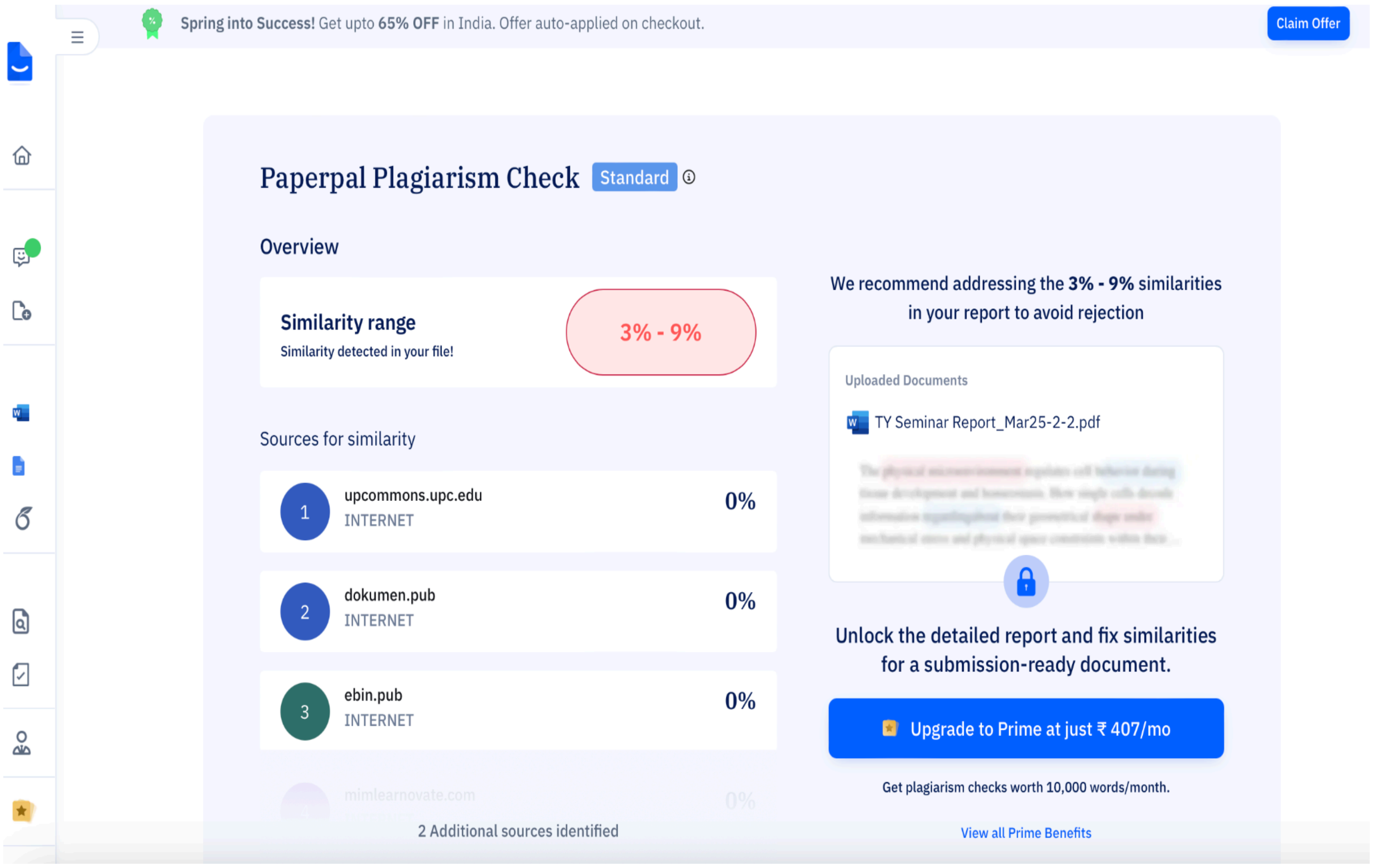
[15] Álvaro, F., Sánchez, J. A., & Benedí, J. M. (2013). Recognition of on-line handwritten mathematical expressions using 2D stochastic context-free grammars and hidden Markov models. Pattern Recognition Letters, 35, 58-67.

[16] Wang, Z., Deng, Z. H., Liu, Y., & Lai, H. (2021). A deep learning approach for recognizing and solving handwritten mathematical equations. Expert Systems with Applications, 168, 114375.

[17] Zhong, Z., Jin, L., & Xie, Z. (2020). Image-to-Markup Generation with Coarse-to-Fine Attention. IEEE Transactions on Pattern Analysis and Machine Intelligence, 42(9), 2367-2382.

[18] Le, T. H., Vo, Q. H., Tran, Q. D., & Pham, T. A. (2022). Enhancing Handwritten Mathematical Expression Recognition with Transformer-based Architectures. Neurocomputing, 489, 235-248.

PLAGIARISM



Spring into Success! Get upto 65% OFF in India. Offer auto-applied on checkout. Claim Offer

Paperpal Plagiarism Check Standard

Overview

Similarity range
Similarity detected in your file!
3% - 9%

Sources for similarity

1	upcommons.upc.edu INTERNET	0%
2	dokumen.pub INTERNET	0%
3	ebin.pub INTERNET	0%
4	mimlearnovate.com	0%

2 Additional sources identified

We recommend addressing the **3% - 9%** similarities in your report to avoid rejection

Uploaded Documents
TY Seminar Report_Mar25-2-2.pdf

Unlock the detailed report and fix similarities for a submission-ready document.

Upgrade to Prime at just ₹ 407/mo

Get plagiarism checks worth 10,000 words/month.

View all Prime Benefits