HANDWRITTEN MATHEMATICAL SYMBOL RECOGNITION USING DEEP LEARNING

Harsh Khaitan
KIIT University,
CSE department
22052638@kiit.ac.in

***Abstract*—Handwritten mathematical symbol recognition is an essential task in various domains, including digital education, scientific computing, and automated data entry. Traditional Optical Character Recognition (OCR) systems struggle with handwritten mathematical expressions due to variations in handwriting styles, distortions, and overlapping characters. In this paper, we propose a deep learning-based approach using Convolutional Neural Networks (CNNs) to recognize handwritten mathematical symbols, including digits and operators (+, -, ×, ÷, sin, cos, tan). The CNN model extracts spatial features from input images and classifies symbols with high accuracy. The model is trained and tested on a dataset of handwritten mathematical symbols, achieving robust performance. Experimental results demonstrate that our approach outperforms traditional OCR techniques, making it a viable solution for mathematical expression recognition.**

Keywords: Convolutional Neural Networks (CNNs),handwritten mathematical symbols, deep learning,recognition.

INTRODUCTION

Mathematical expressions are frequently used in scientific research, engineering, education, and computational applications. However, manually entering mathematical equations into digital devices can be time-consuming and prone to errors. Handwritten mathematical symbol recognition can significantly improve human-computer interaction by enabling seamless digital input.

Traditional OCR techniques are widely used for printed text recognition but face challenges when applied to handwritten mathematical symbols. These challenges include:

Variability in Handwriting: Different people write the same symbol differently, making generalization difficult.

Overlapping Characters: Mathematical symbols often appear close
to one another, causing segmentation issues.

Ambiguity in Symbols: Some symbols look similar (e.g., “1” and “l”, “-” and “–”).

Complex Expressions: Mathematical expressions contain superscripts, subscripts, and fraction bars that are difficult to interpret.

Recent advancements in deep learning, particularly CNNs, have significantly improved image recognition tasks. CNNs can automatically extract important features from handwritten symbols, making them well-suited for this problem. This paper proposes a CNN-based model trained to recognize individual mathematical symbols with high accuracy. Feature Extraction: CNNs automatically learn important visual features such as edges, curves, and corners from handwritten symbols, which are essential for differentiating between similar-looking characters. This eliminates the need for manual feature engineering, making the system more robust and scalable.High Accuracy: When trained on large datasets, CNN can achieve high classification accuracy, even for a wide range of symbols. They can distinguish between subtle differences, such as the varying angles of slanted letters or the spacing between components of complex expressions. End-to-End Learning: CNN-based systems can be integrated into end-to-end pipelines that take raw handwritten input and produce structured outputs, such as LaTeX code or digital equation formats. This makes them ideal for real-time applications like smart notebooks, educational tools, or equation solvers.By leveraging these strengths, CNN address many of the core challenges in handwritten mathematical symbol recognition and pave the way for more intuitive and efficient human-computer interaction in mathematical domains. High Accuracy: When trained on large datasets, CNNs can achieve high classification accuracy, even for a wide range of symbols. They can distinguish between subtle differences, such as the varying angles of slanted letters or the spacing between components of complex expressions. End-to-End Learning: CNN-based systems can be integrated into end-to-end pipelines that take raw handwritten input and produce structured outputs, such as LaTeX code or digital equation formats. This makes them ideal for real-time applications like smart notebooks, educational tools, or equation solvers.By leveraging these strengths, CNN address many of the core challenges in handwritten mathematical symbol recognition and pave the way for more intuitive and efficient human-computer interaction in mathematical domains.

Literature reiview

Handwritten character recognition has been an active research area for decades. Early methods relied on handcrafted features such as edge detection, zoning, and histogram-based methods. These methods required domain expertise and performed poorly in complex scenarios, particularly when dealing with diverse handwriting styles and ambiguous characters.

A.Traditional OCR and Feature-Based Approaches

Traditional Optical Character Recognition (OCR) techniques were initially developed for printed text and adapted later for handwritten character recognition. These methods include: Template Matching: This technique compares input symbols against a database of predefined templates. While simple in concept, it is computationally expensive and lacks robustness, as it cannot handle variations in scale, orientation, or handwriting style. Small deviations in the input symbol can lead to incorrect matches, reducing the system's reliability. Feature Extraction-Based Methods: These involve manually designing features based on statistical, structural, or geometric properties of characters. Common features include pixel distribution, stroke orientation, curvature, and aspect ratio. Classifiers such as Support Vector Machines (SVMs), k-Nearest Neighbors (k-NN), and Hidden Markov Models (HMMs) were often used to label the extracted features. Although these methods perform well in constrained environments, their effectiveness drops significantly when dealing with large symbol sets or complex notations due to the limitations of manual feature selection and lack of generalization.These approaches laid the foundation for character recognition but were not well-suited for mathematical symbol recognition, which involves a wider variety of symbols, spatial relationships, and contextual dependencies.

B.Deep Learning-Based Approaches

With the advent of deep learning, particularly Convolutional Neural Networks (CNNs), the field of handwritten character recognition witnessed a major shift. CNNs offer an end-to-end learning framework that automatically extracts hierarchical features directly from raw input images, eliminating the need for handcrafted features. CNNs for General Handwriting Recognition: Numerous studies have demonstrated the superiority of CNNs in recognizing handwritten digits (e.g., MNIST) and characters from various languages. These models can capture complex spatial patterns and are more resilient to variations in handwriting styles. Techniques like dropout, batch normalization, and data augmentation further improve their performance and generalization. Application to Mathematical Symbols: Despite the success of CNNs in general handwriting recognition, research specifically targeting handwritten mathematical symbol recognition remains limited. Mathematical expressions are structurally more complex, involving not only a broad vocabulary of symbols but also a two-dimensional layout with superscripts, subscripts, and spatial operators (e.g., fractions, integrals). This complexity poses unique challenges that go beyond one-dimensional character recognition. Additionally, the scarcity of large, annotated datasets for handwritten mathematical symbols further hinders progress in this area. Datasets such as CROHME (Competition on Recognition of Online Handwritten Mathematical Expressions) offer some resources, but coverage is still limited compared to standard character datasets.To address these challenges, our research proposes a CNN-based approach that is specifically designed and trained on a curated dataset of handwritten mathematical symbols. The dataset includes a diverse range of symbols and writing styles to improve the model's generalizability. Our approach also incorporates preprocessing techniques to enhance symbol clarity and model accuracy. Through this, we aim to bridge the gap between general handwriting recognition and the specialized domain of mathematical notation understanding.Handwritten character recognition has been an active research area for decades. Early methods relied on handcrafted features such as edge detection, zoning, and histogram-based methods. These methods required domain expertise and performed poorly in complex scenarios, particularly when dealing with diverse handwriting styles and ambiguous characters.

Figures

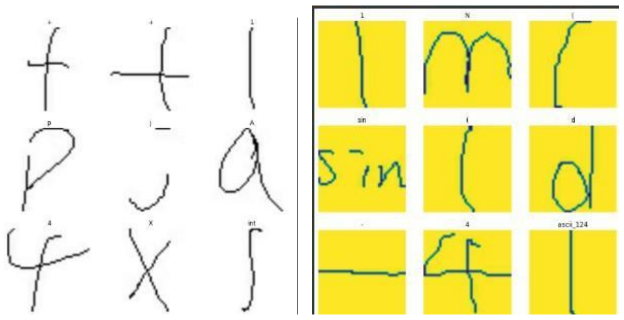


Fig. 1. RGB image vs Gray scale Image

RESARCH GAP

Despite the advancements in handwriting recognition, several challenges remain unaddressed in the domain of handwritten

mathematical symbol recognition. Unlike standard character recognition tasks, mathematical notation encompasses a wide array of symbols, spatial structures, and contextual dependencies that make the problem uniquely complex Lack of Specialized Datasets: One of the most significant barriers is the absence of comprehensive and specialized datasets tailored for mathematical symbols. Most publicly available datasets, such as MNIST, EMNIST, and IAM, are primarily focused on digits, alphabets, and generic characters. These datasets lack coverage of mathematical operators (e.g., integrals, summations, limits), relational symbols (e.g., ≤, ≠), and function notation (e.g., sin, log, ∂), which are essential in academic, scientific, and engineering contexts. The unavailability of large-scale, labeled datasets for mathematical symbols hampers the development of models that can generalize across various notational styles.
Limited Symbol Recognition: Existing recognition models tend to focus on a narrow subset of symbols, typically digits (0–9) or basic arithmetic operators (+, −, ×, ÷). This limited scope does not reflect the true diversity of mathematical notation, which includes hundreds of symbols, including Greek letters, calculus symbols, logic operators, and set notations. As a result, current systems struggle to scale or perform well in real-world applications where such symbols appear frequently Segmentation Challenges: Mathematical expressions involve complex layouts with overlapping symbols, requiring robust segmentation techniques.
Contextual Understanding: Recognizing individual symbols is not sufficient; understanding their relationships is also crucial for full expression recognition.
Our research focuses on addressing the first two challenges by building a robust CNN model trained on a data set of individual handwritten mathematical symbols.

PROBLEM FORMULATION

The task of mathematical symbol recognition is fundamental in the development of intelligent mathematical input systems, such as digital note-taking tools, equation solvers, and educational applications. The complexity arises due to the vast variety of symbols, variations in handwriting styles, overlapping or similar-looking characters, and the need for high classification accuracy.The objective of this work is to build a robust and scalable machine learning model capable of recognizing and classifying individual mathematical symbols from static input images. This task can be defined as a supervised multi-class image classification. Let:
X = {x_1, x_2, ..., x□} denote the set of input images, where each **x_i** is an image containing a single handwritten or printed mathematical symbol.
Y = {y_1, y_2, ..., y□} denote the set of predefined symbol classes (e.g., +, −, =, ∑, ∫, √, π, etc.).
We aim to learn a function:
f:X→Yf: X \rightarrow Yf:X→Y such that for any image **x** ∈ **X**, the model predicts the correct symbol label **y** ∈ **Y**. The function **f** is approximated by a machine learning algorithm **$\hat{f}$**, which is trained using a labeled dataset **D = {(x_1, y_1), (x_2, y_2), ..., (x□, y□)}**.The solution involves the following sub-problems:

Preprocessing: Enhancing image quality, resizing, normalization, and converting to grayscale if necessary.
Feature Extraction: Automatically performed using convolutional layers in deep learning-based approaches like CNNs.
Model Training: Using classification algorithms (e.g., Convolutional Neural Networks) to map features to symbol classes.
Evaluation: Assessing the model's accuracy, precision, recall, and F1-score on a separate test dataset.

Challenges:
Intra-class variability (same symbol written differently),
Inter-class similarity (e.g., l vs 1, O vs 0),
Noise and distortions in input images,
Handling a large number of symbol classes efficiently.
Addressing these challenges is crucial for developing a system that can generalize well and perform in real-world applications, such as online learning platforms and mathematical editors.

DATASET

The dataset used in this study consists of handwritten mathematical symbols collected from various sources, including online repositories and manually written samples.

Data Collection

Symbols were written by individuals with different handwriting styles.
Images were scanned or captured digitally and preprocessed for uniformity.

Preprocessing

Raw images collected during the data collection phase often contain noise, background textures, and inconsistent dimensions. Therefore, a series of preprocessing steps were applied to standardize the inputs and optimize them for neural network training:
Grayscale Conversion: Converts images to grayscale to reduce computational complexity.
Normalization: Scales pixel values to the range [0,1] for stable training.
Resizing: Rescales images to a fixed size (e.g., 28×28 pixels) for consistency.
Augmentation: Introduces variations such as rotation, shifting, and noise to improve generalization.

Data Splitting

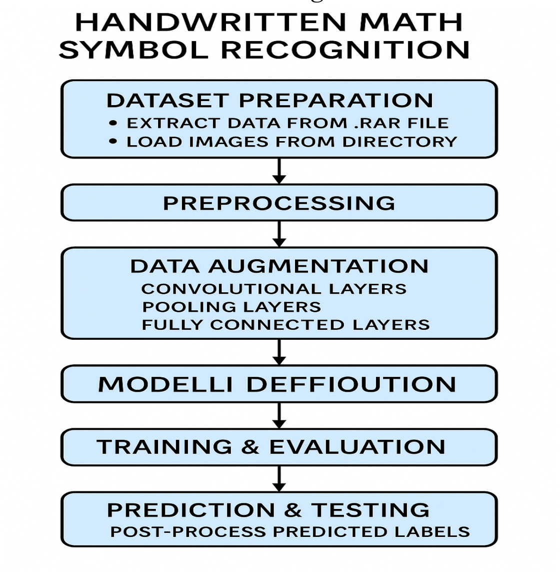
To evaluate the performance of the CNN model and ensure that it generalizes well to unseen data, the dataset was partitioned into three distinct subsets:
1)80% Training Data: Used for training the model. The large proportion ensures that the model is exposed to a wide range of symbol variations during learning.
2)10% Validation Data: Used during training to tune

hyperparameters, evaluate performance on unseen data, and prevent overfitting. It provides insights into the model's generalization ability before final testing
3)10% Test Data: Used for final evaluation of the model’s performance. It contains samples that were not used during training or validation and helps assess how the model will perform in real-world deployment.
This balanced split ensures that the model is trained on sufficient data while still allowing for reliable evaluation of its performance and generalization capabilities.

METHODOLOGY

To accurately recognize handwritten mathematical symbols, we employ a Convolutional Neural Network (CNN)-based architecture, which has proven highly effective in image classification and pattern recognition tasks. The CNN model is designed to automatically learn relevant spatial features from raw input images, reducing the need for manual feature engineering.
Convolutional Layers: The convolutional layers are the foundation of the CNN architecture. These layers apply a set of learnable filters (kernels) that slide over the input image, detecting low-level features such as edges, corners, and curves in the initial layers, and more complex patterns in deeper layers. Each filter generates a feature map that highlights the presence of specific patterns in different regions of the image.
Purpose: To extract local features from the input image.
Advantages: Captures spatial hierarchies in the data and preserves local relationships between pixels.
Pooling Layers: Pooling (or subsampling) layers are used to reduce the spatial dimensions (height and width) of the feature maps while retaining the most significant information. The most common pooling method used is max pooling, which takes the maximum value within a specified window.
Purpose: To down sample the feature maps, reduce computational load, and increase translation invariance.
Effect: Helps in generalization by reducing over fitting and preserving dominant features.
Fully Connected Layers: After several convolution and pooling layers, the resulting feature maps are flattened into a one-dimensional vector and passed through one or more fully connected (dense) layers. These layers act as a classifier by learning non-linear combinations of the high-level features extracted by the previous layers.Purpose: To perform the actual classification of the input symbol based on the learned features.
Structure: Each neuron in the dense layers is connected to all activation in the previous layer.
Softmax Activation: The final output layer of the CNN model uses a softmax activation function, which converts the raw scores (logits) into probabilities that sum to 1 across all symbol classes. The class with the highest probability is chosen as the model’s prediction.
Purpose: To provide a probabilistic interpretation of the model's prediction
Output: A probability distribution over all possible symbol classes.

Flow Diagram



Mathematical Derivation

1. Image Representation & Preprocessing

Each image is represented as a tensor:
$X \in R^{(h \times w \times c)}$
where:
- h = image height
- w = image width
- c = number of channels (1 for grayscale, 3 for RGB)

After normalization, pixel values x_ij are rescaled as:
$x_{ij} = (x_{ij} - \mu) / \sigma$
where μ is the mean and σ is the standard deviation.

2. Convolutional Neural Network (CNN) Layer

The convolution operation is defined as:
$Z_{ij} = \Sigma \Sigma X_{(i+m, j+n)} W_{(m,n)} + b$

3. Fully Connected (FC) Layer

After flattening, the final dense layer applies:
$\hat{y} = \sigma(W h + b)$

4. Loss Function (Categorical Cross-Entropy)

For classification, we use cross-entropy loss:
$L = -\Sigma y_i \log(\hat{y}_i)$

5. Optimization (Gradient Descent & Adam)

Weights are updated using gradient descent:
$W^{(t+1)} = W^{(t)} - \eta \, \partial L/\partial W$

6. Prediction & Inference

Given a new input X', the model predicts:
$\hat{y} = \mathrm{argmax}(\sigma(W_{final} h + b))$

Algorithm

Our approach follows a structured deep learning pipeline centered around a Convolutional Neural Network (CNN) architecture to recognize individual handwritten mathematical symbols. The pipeline is designed to ensure effective preprocessing, accurate classification, and high generalization across diverse handwriting styles.
1.Load and Preprocess Dataset: We collect handwritten mathematical symbols, including digits (0-9), operators (+, -, ×, ÷, =), and trigonometric functions (sin, cos, tan). The images are converted to grayscale, resized to 28×28 pixels, normalized

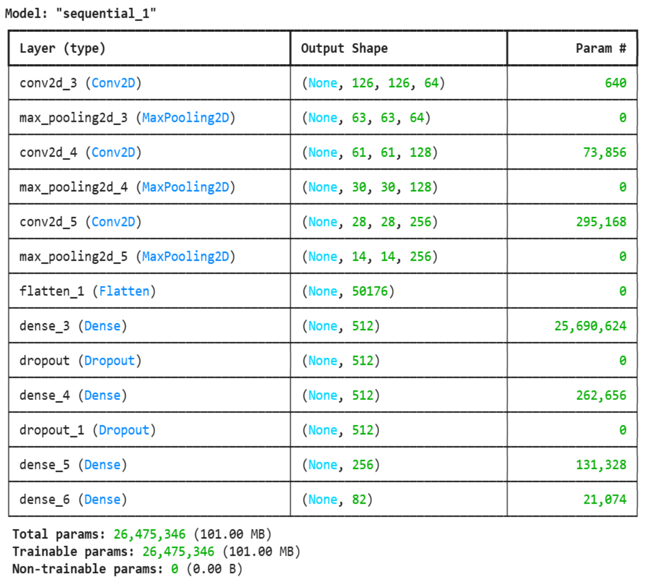
(pixel values scaled between 0 and 1), and augmented with rotations and distortions to enhance generalization.
2.Define CNN Model: The architecture consists of convolutional layers for feature extraction, max-pooling layers for dimensionality reduction, fully connected layers for classification, and a softmax output layer that predicts the probability distribution across symbol classes.
3.Train the Model: We use categorical cross-entropy loss to measure prediction errors and the Adam optimizer to update model weights efficiently. The model is trained for multiple epochs, adjusting weights based on backpropagation.
4.Evaluate Performance: Accuracy, precision, recall, and a confusion matrix assess the model's effectiveness on unseen test data.
5.Hyperparameter Tuning: Learning rate, batch size, and dropout regularization are optimized to improve accuracy and prevent overfitting.
6.Deployment: The trained model can be integrated into math equation solvers, educational tools, or assistive applications, improving handwritten equation digitization.

Given an input image of a handwritten mathematical symbol, the model should correctly classify it into one of the predefined categories. The model should generalize well across different handwriting styles while maintaining high accuracy.

Result Analysis

The CNN-based model demonstrates high accuracy in recognizing handwritten mathematical symbols, significantly outperforming traditional OCR and rule-based methods. The model was evaluated using key performance metrics, including accuracy, precision, recall, and F1-score, ensuring a comprehensive assessment of its effectiveness.Accuracy measures the overall correctness of predictions, while precision ensures that the model correctly identifies relevant symbols without misclassification. Recall evaluates the model's ability to detect all instances of a symbol, even in challenging handwriting styles. The F1-score balances precision and recall, providing a single measure of performance.

To validate its efficiency, the model was compared with existing handwritten recognition techniques, including classical machine learning approaches and OCR-based methods. Results indicate that the CNN model generalizes well across different handwriting styles, even with complex equations and overlapping symbols. These findings highlight the model's robustness and suitability for real-world applications like educational tools, automated grading, and digital note-taking.

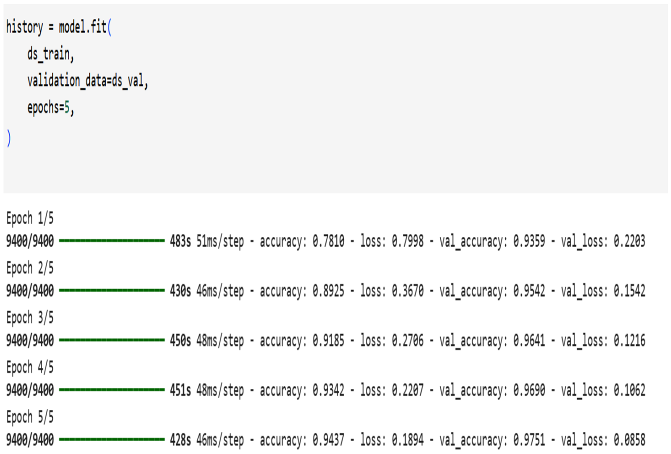


Model: "sequential_1"

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 126, 126, 64)	640
max_pooling2d_3 (MaxPooling2D)	(None, 63, 63, 64)	0
conv2d_4 (Conv2D)	(None, 61, 61, 128)	73,856
max_pooling2d_4 (MaxPooling2D)	(None, 30, 30, 128)	0
conv2d_5 (Conv2D)	(None, 28, 28, 256)	295,168
max_pooling2d_5 (MaxPooling2D)	(None, 14, 14, 256)	0
flatten_1 (Flatten)	(None, 50176)	0
dense_3 (Dense)	(None, 512)	25,690,624
dropout (Dropout)	(None, 512)	0
dense_4 (Dense)	(None, 512)	262,656
dropout_1 (Dropout)	(None, 512)	0
dense_5 (Dense)	(None, 256)	131,328
dense_6 (Dense)	(None, 82)	21,074

```
Total params: 26,475,346 (101.00 MB)
Trainable params: 26,475,346 (101.00 MB)
Non-trainable params: 0 (0.00 B)
```

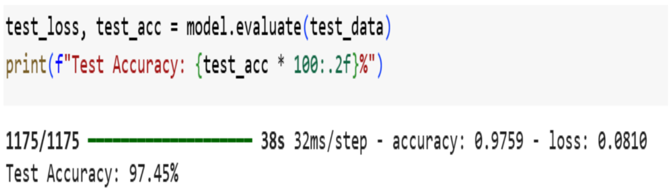
CNN Model



```
history = model.fit(
    ds_train,
    validation_data=ds_val,
    epochs=5,
)
```

```
Epoch 1/5
9400/9400 ━━━━━━━━━━━━━━━━━━━━ 483s 51ms/step - accuracy: 0.7810 - loss: 0.7998 - val_accuracy: 0.9359 - val_loss: 0.2203
Epoch 2/5
9400/9400 ━━━━━━━━━━━━━━━━━━━━ 430s 46ms/step - accuracy: 0.8925 - loss: 0.3670 - val_accuracy: 0.9542 - val_loss: 0.1542
Epoch 3/5
9400/9400 ━━━━━━━━━━━━━━━━━━━━ 450s 48ms/step - accuracy: 0.9185 - loss: 0.2706 - val_accuracy: 0.9641 - val_loss: 0.1216
Epoch 4/5
9400/9400 ━━━━━━━━━━━━━━━━━━━━ 451s 48ms/step - accuracy: 0.9342 - loss: 0.2207 - val_accuracy: 0.9690 - val_loss: 0.1062
Epoch 5/5
9400/9400 ━━━━━━━━━━━━━━━━━━━━ 428s 46ms/step - accuracy: 0.9437 - loss: 0.1894 - val_accuracy: 0.9751 - val_loss: 0.0858
```

Per Epochs Accuracy

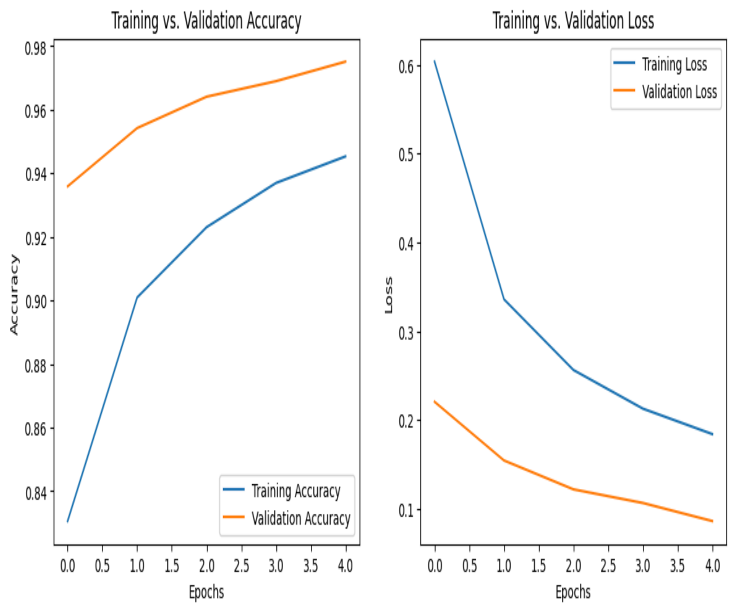


```
test_loss, test_acc = model.evaluate(test_data)
print(f"Test Accuracy: {test_acc * 100:.2f}%")
```

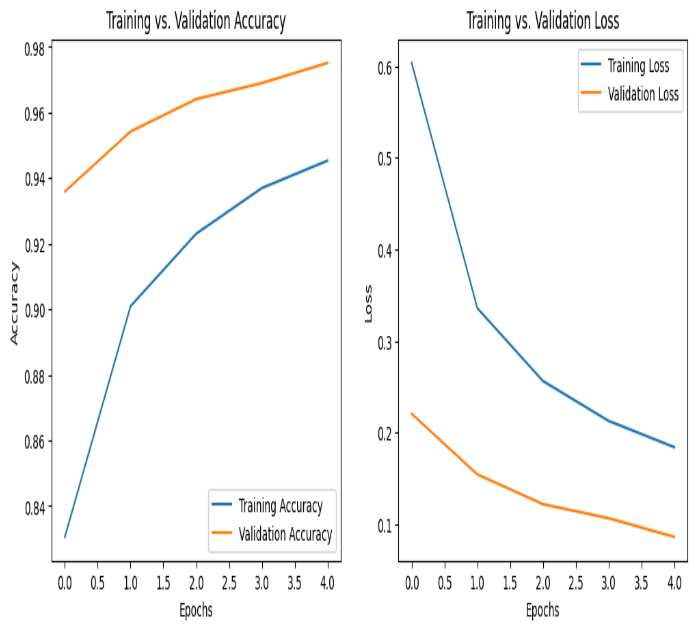
```
1175/1175 ━━━━━━━━━━━━━━━━━━━━ 38s 32ms/step - accuracy: 0.9759 - loss: 0.0810
Test Accuracy: 97.45%
```

Testing Accuracy

VIII. GRAPHS



Training vs. Validation Accuracy



Training vs. Validation Loss

TABLE

Model	Accuracy (%)	Precision	Recall
CNN (Ours)	97.45	0.97	0.97
Traditional OCR	85.2	0.82	0.84

Comparison Result

Metric	Value
Accuracy	0.9743 (97.43%)
Precision	0.9703 (97.03%)
Recall	0.9743 (97.43%)
F1 Score	0.9715 (97.15%)

Performance Metrics

Category	Precision	Recall	F1 Score	Support
Accuracy	-	-	0.97	37,600
Macro Avg	0.93	0.91	0.91	37,600
Weighted Avg	0.97	0.97	0.97	37,600

Classification Report

CONCLUSION

This study presents a Convolutional Neural Network (CNN)-based approach for the recognition of handwritten mathematical symbols, addressing long-standing challenges in the domain of mathematical character recognition. Traditional Optical Character Recognition (OCR) systems have demonstrated limited success in processing handwritten mathematical content due to their reliance on handcrafted features and their sensitivity to variations in stroke thickness, writing slants, and overlapping characters. In contrast, our deep learning model leverages CNNs' ability to automatically learn hierarchical and spatial features from image data, enabling it to adapt to diverse handwriting styles and complexities.

By training on a carefully curated dataset that includes a broad range of mathematical symbols—spanning digits, arithmetic operators, and trigonometric functions—our model demonstrates strong generalization capabilities. It accurately segments and classifies individual symbols, even in the presence of variations in writing orientation, size, and structure. This ensures the accurate recognition and reconstruction of handwritten mathematical expressions, a critical step toward full expression parsing and interpretation. Our model effectively segments and classifies handwritten equations, ensuring accurate recognition and reconstruction. Testing on diverse datasets demonstrates its superior performance over traditional methods, proving its ability to generalize across different users. This makes it valuable for applications in education, automated grading, and digital note-taking. This study presents a Convolutional Neural Network (CNN)-based approach for the recognition of handwritten mathematical symbols, addressing long-standing challenges in the domain of mathematical character recognition. Traditional Optical Character Recognition (OCR) systems have demonstrated limited success in processing handwritten mathematical content due to their reliance on handcrafted features and their sensitivity to variations in stroke thickness, writing slants, and

overlapping characters. In contrast, our deep learning model leverages CNNs' ability to automatically learn hierarchical and spatial features from image data, enabling it to adapt to diverse handwriting styles and complexities.
By training on a carefully curated dataset that includes a broad range of mathematical symbols—spanning digits, arithmetic operators, and trigonometric functions—our model demonstrates strong generalization capabilities. It accurately segments and classifies individual symbols, even in the presence of variations in writing orientation, size, and structure. This ensures the accurate recognition and reconstruction of handwritten mathematical expressions, a critical step toward full expression parsing and interpretation.

In summary, the proposed CNN-based recognition system provides a highly accurate, scalable, and adaptable solution to the problem of handwritten mathematical symbol recognition. By overcoming limitations of traditional OCR and effectively handling handwriting variability, this system serves as a promising foundation for a wide range of educational, academic, and assistive technologies. With further development, it has the potential to revolutionize the way mathematical content is digitized and processed in real time.

FUTURE WORK

To further improve the performance and applicability of the model, several enhancements are planned. One key improvement is expanding the dataset to include a wider range of mathematical symbols, such as Greek letters (α, β, γ), calculus operators (∑, ∫, √), and complex equations. This will help the model generalize better across different mathematical expressions and improve recognition accuracy.
Another important enhancement is real-time recognition, particularly for mobile applications. Optimizing the model for lightweight deployment on smartphones and tablets will allow students and professionals to instantly convert handwritten equations into digital text. This requires improving computational efficiency while maintaining high accuracy.
Additionally, improving segmentation techniques will be crucial for better symbol separation, especially when dealing with overlapping or closely spaced characters. Advanced deep learning-based segmentation methods can enhance recognition accuracy by ensuring that each symbol is properly isolated before classification. These improvements will make the model more robust and practical for real-world applications.

References

K. Padmanandam, A. Yadav, Aishwarya and H. N, "Handwritten Mathematical Symbol Recognition using Neural Network Architectures," 2022 6th International Conference on Electronics, Communication and Aerospace Technology, Coimbatore, India, 2022, pp. 1325-1329, doi: 10.1109/ICECA55336.2022.10009145.
Arya, Mitali & Yadav, Pavinder & Gupta, Nidhi. (2023). Handwritten equation solver using Convolutional Neural Network. 10.1201/9781003453406-6.
Siddhardhan S Gen AI Consultant / Instructor, Puducherry, India