Ibn Al-Haitham Journal for Pure and Applied Sciences
Journal homepage: jih.uobaghdad.edu.iq
PISSN: 1609-4042, EISSN: 2521-3407
IHJPAS. 2025, 38(4)

Enhanced OCR Techniques for Recognizing Mathematical Expressions in Scanned Documents

Yahya Baher Al-Askary[1*], **Saad Al-Momen**[2]
[1,2]Department of Mathematics, College of Sciences, University of Baghdad, Baghdad, Iraq.
*Corresponding Author.

Received: 2 July 2023 Accepted: 1 October 2023 Published: 20 October 2025
doi.org/10.30526/38.4.3640

Abstract

When OCR systems are utilized to recognize mathematical expressions in scanned documents, they encounter numerous challenges. These challenges arise because the mathematical expressions use an extensive array of symbols, variables, numbers, and operations, each with its distinct writing style. Moreover, mathematical expressions have a hierarchical relationship and adhere to logical rules governing the grouping of their components. This paper presents a three-step approach segmentation, symbol recognition, and interpretation to overcome these issues. The segmentation process aims to identify and separate each symbol based on its spatial location. Then, various features are extracted to describe the horizontal and vertical projections of the symbol, enabling effective recognition. Two neural network-based classification methods were proposed for the recognition of the symbols. The first one achieved 96.6% recognition rates, while the second achieved 97.7%. Lastly, the paper introduces three guidelines for interpreting the mathematical meaning of the expression and accurately converting it into textual form. The study demonstrates the potential of these methods in enhancing the capabilities of OCR systems for recognizing mathematical expressions.

Keywords: Classification, Mathematical Expression, Neural Network, Character Recognition, Image Processing.

1. Introduction

There has been a remarkable revolution in the field of Optical Character Recognition (OCR) in recent years, leading to a complete transformation in the way scanned documents are converted into digital format (1). This advancement has made it possible to archive these documents with high accuracy and efficiency. Nevertheless, there is still a notable challenge to overcome: recognizing mathematical expressions within these documents. The complexity lies in the diverse range of symbols, variables, numbers, and mathematical operations that form these expressions. The extensive variety of components in mathematical expressions poses a significant obstacle to their recognition (2, 3). They can involve Greek symbols, subscripts, superscripts, numbers, and mathematical symbols, whereas OCR systems are primarily designed to recognize standard alphabet characters and numbers (4). As a result, there is an urgent need to enhance these systems and broaden their capabilities.

Moreover, mathematical expressions are not merely an assortment of isolated symbols; they exhibit hierarchical and logical relationships governed by mathematical rules. Optical recognition systems encounter difficulty in discerning these relationships, often resulting in ambiguous or erroneous interpretations. Simultaneously, mathematical expressions are composed in various fonts and styles (5). They can be handwritten or printed in scientific and mathematical formats (6,7). Consequently, optical recognition systems must address this vast diversity and adapt their recognition mechanisms accordingly. The intricate nature of this diversity adds complexity to the recognition process.

Based on the preceding facts, it becomes evident that optical recognition systems are prone to errors when it comes to recognizing mathematical expressions. Consequently, these errors can result in misinterpretations, causing a complete alteration in the meaning of a mathematical expression due to the misinterpretation of a single symbol or component within that expression. This can potentially propagate errors that originated in earlier stages, ultimately leading to critical errors in subsequent stages (8).

Researchers have made notable strides in tackling the formidable challenges faced by OCR systems in recognizing mathematical expressions. Specialized algorithms and techniques have been developed, leveraging machine learning methods like deep learning (9) and neural networks (10). These advancements have significantly enhanced the accuracy and resilience of mathematical expression recognition. Various approaches have been suggested, encompassing template-based recognition, syntactic analysis, semantic analysis, and integrated text-image processing (11).

2. Related Work

Recently there has been an increase use of Deep Neural Networks (DNNs) as feature extractors and classifiers. In 2018, Gunawan, Noor, and Kartiwi published a paper that focuses on developing an offline handwritten recognition system using DNN. To train and test the DNN, they selected two well-known English digits and letters databases: MNIST and EMNIST. These databases consist of 10 number digits and 52 English letters (capital and small). They proposed a two-stacked layer DNN system to achieve a 97.7% recognition accuracy for English digits and 88.8% for letters. The results demonstrated that the proposed system excels in accurately recognizing handwritten English digits and letters. (12).

In 2019, Veres, Rishnyak, and Rishniak conducted a study to explore the feasibility of simultaneously performing structural analysis and character classification. They proposed a classification process for symbols and the development of a system based on machine learning techniques. In their study, the researchers presented a multi-classifier system that utilized five different classifiers to achieve the best possible outcome. The researchers provided evidence that this approach is highly effective in the field of structural analysis and character classification (13).

Furthermore, in 2019, Nazemi, Tavakolian, Fitzpatrick, Fernando, and Suen employed deep learning techniques to recognize offline handwritten mathematical symbols. Their study focused on proposing symbol segmentation methods and achieving precise classification for a wide range of over 300 mathematical symbol classes. The paper introduces the utilization of Simple Linear Iterative Clustering for symbol segmentation and evaluates the accuracy of various classifiers for symbol recognition. The classifiers utilized in this study are kNN, salient features, Histogram of Oriented Gradient (HOG), Linear Binary Pattern (LBP), modified LeNet, and SqueezeNet (14).

Wang and Shan in 2020 used a multiscale neural network to recognize the handwritten mathematical expressions. They highlight the challenges of recognizing handwritten mathematical expressions due to the variability in writing styles and the complexity of mathematical symbols. They proposed a model that can effectively analyze the two-dimensional structure of handwritten mathematical expressions and identify different mathematical symbols in a long expression (3).

Also in 2020, Chan proposed an over-segmentation method to automatically extract strokes from textual bitmap images. The process involved breaking down the skeleton of a binarized image into junctions and segments. These segments were then merged to form strokes, and the stroke order was normalized using recursive projection and topological sort techniques. This approach demonstrated high offline accuracy when combined with conventional online recognizers. (15).

In 2021, Yan, Zhang, Gao, Yuan, and Tang introduced ConvMath, a convolutional sequence modeling network, to recognize mathematical expressions. ConvMath integrates an image encoder for extracting features and a convolutional decoder for generating sequences. This study showed significantly improved efficiency compared to other methods. (16).

Truong, Nguyen, Nguyen, and Nakagawa in 2021 recognized online handwritten mathematical expressions by building a symbol relation tree directly from a sequence of strokes using a bidirectional recurrent neural network (17).

3. Proposed Method

Designing a system for mathematical expression recognition required careful consideration of the specific properties of these expressions. These properties arise from the extensive use of diverse symbols and their arrangement in a two-dimensional format, which are key factors that distinguish mathematical expressions from ordinary texts (18).

The recognizing mathematical expressions process has three fundamental components: segmentation, symbol recognition, and mathematical expression interpretation. These three components are indispensable in developing a comprehensive and effective system for mathematical expression recognition.

3.1. Segmentation

In our previous paper (19), the image pre-processing and image segmentation processes were extensively discussed. The pre-processing stage consisted of many steps, such as the image containing the equation being read, the conversion of the image into a binary form, noise reduction, skew correction, and the linking of broken symbols. The result of these steps is a processed image ready for the segmentation process.

The segmentation stage is initiated, wherein the identification and separation of each symbol take place, relying on the horizontal and vertical distances between the components of the equation. The projections are determined by adding up the values of the binary image B_{ij}, alongside rows H_i and columns V_j. The techniques utilized essentially dissect the image into individual elements based on the spatial distribution of the pixels, as portrayed in the succeeding equations:

$$H_i = \sum_{j=1}^{M} B_{ij} \; where \; (i = 1,2, \dots , N) \quad (1)$$

$$V_j = \sum_{i=1}^{N} B_{ij} \; where \; (j = 1,2, \dots , M) \quad (2)$$

Subsequently, the image is partitioned into multiple sub-images, were each sub-image exclusively containing one of the equations Separated component.

3.2 Symbol Recognition

The classification process typically consists of two phases: (i) The training phase where the classifier is taught to identify a specific class of reference feature vectors. Then, (ii) the testing (classification) phase, where unknown vectors are categorized based on a best match criterion. **Figure 1** presents a block diagram that visualizes this process. W represents the matrix resulting from the segmentation stage. This matrix contains the coordinates of the four corners of the rectangle that surrounds the symbol C_i relatively to the original image.

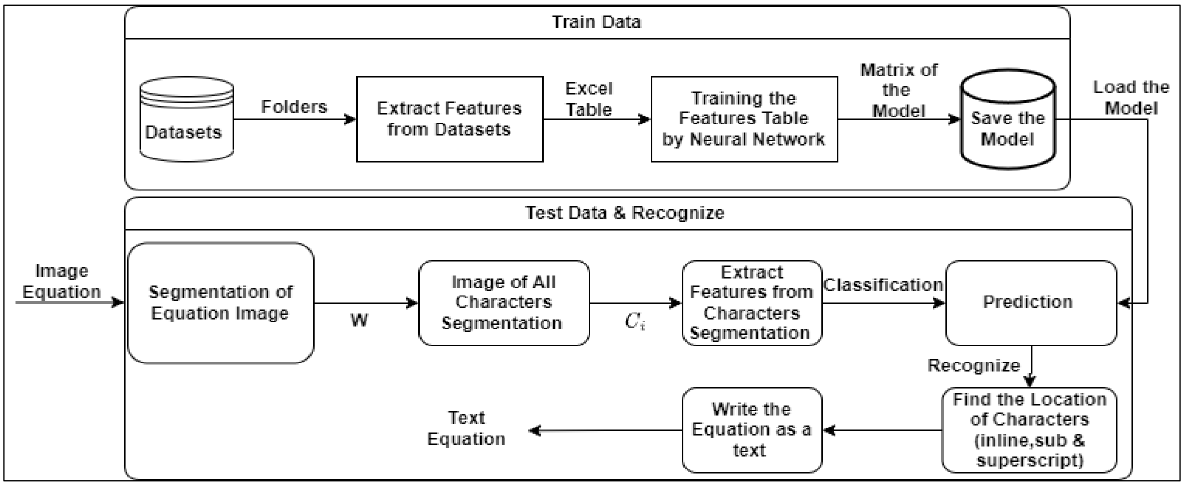

Figure 1. Block diagram of the proposed method.

3.2.1. Feature Extraction

The feature extraction phase is key to drawing out vital information from the text image, aiding in the recognition of characters within the text. Choosing a consistent and representative set of features forms the crux of the pattern recognition system (20).

To streamline the feature extraction process, the images containing symbols were standardized to a size of 200×200 pixels. Subsequently, each image was partitioned into P smaller squares, with dimensions of $m \times n$ pixels per square. It is evident that the density, pixel distribution, geometric characteristics, and statistical properties vary significantly across these squares when the symbols differ. However, in the case of similar symbols, these features exhibit remarkable similarity. See **Figure 2,** which gives some examples

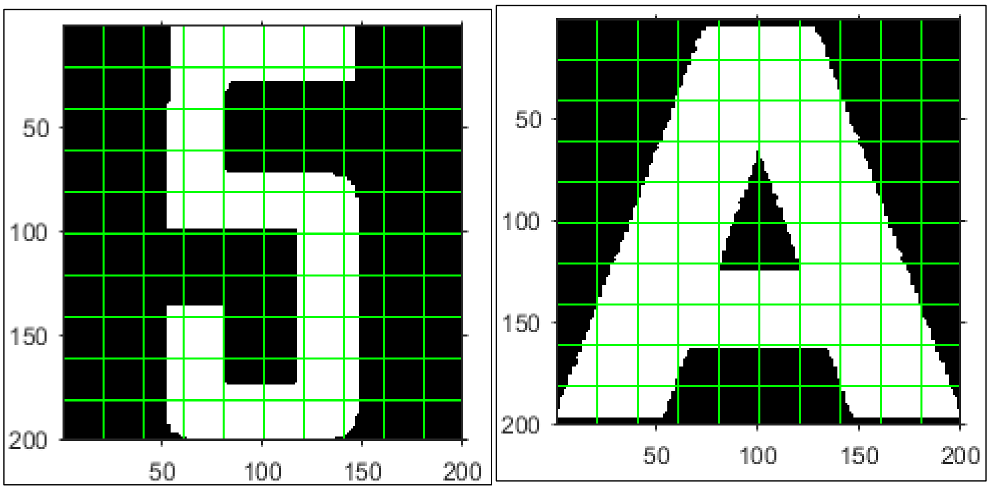

Figure 2. Partitioning the binary cropped image into 100 sub-squares**.**

Each of the P squares represents a matrix of size $m \times n$ pixels. Using **Equations 1** and **2** mentioned earlier, two vectors, H_i and V_i $(i = 1,2,\ldots P)$, are calculated for each matrix.

These vectors represent the horizontal and vertical projections of the respective matrix. The purpose of these two vectors is to facilitate the following calculations:

The mean of the projection vector of the ith square:

$$\mu_{X_i} = \frac{1}{L}\sum_{k=1}^{L} X_i(k), \tag{3}$$

The standard deviation of the projection vector of the ith square:

$$S_{X_i} = \sqrt{\frac{1}{L-1}\sum_{k=1}^{L}\left|X_i(k) - \mu_{X_i}\right|^2}, \tag{4}$$

The variance of the projection vector of the ith square:

$$V_{X_i} = \frac{1}{L-1}\sum_{k=1}^{L}\left|X_i(k) - \mu_{X_i}\right|^2, \tag{5}$$

Kurtosis of the projection vector of the ith square:

$$k_{X_i} = \frac{E\left(X_i - \mu_{X_i}\right)^4}{{S_{X_i}}^4}, \tag{6}$$

This feature provides information about the peakedness or flatness of a distribution compared to a normal distribution.

For **Equations 3-6** (21), $X \in \{H, V\}$ and L represent the length of X. **Figure 3** shows the 8 features that are calculated for the vertical and horizontal projections for an arbitrary square within the image.

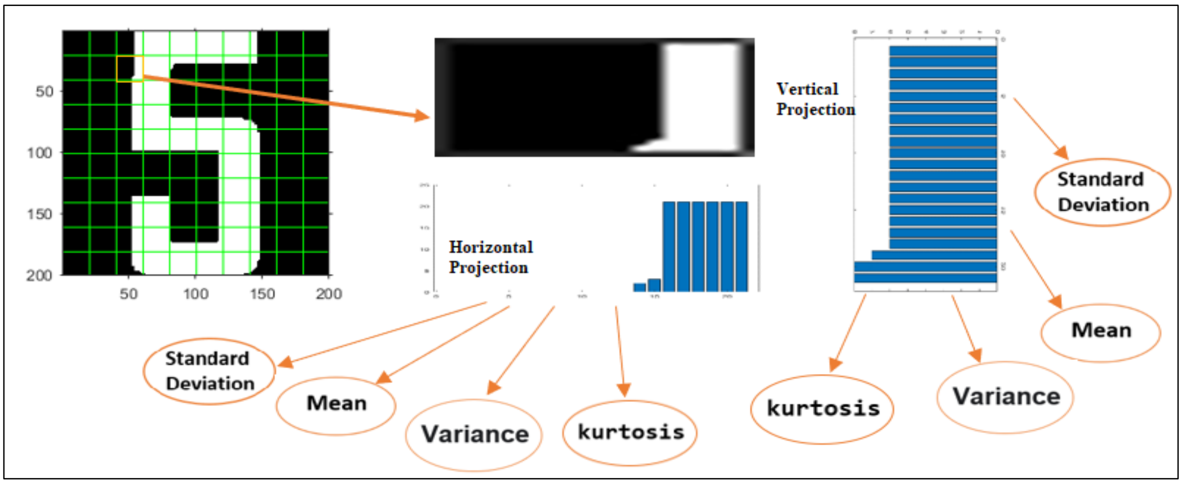

Figure 3. An enlarged image of an arbitrary square and the eight features that have been extracted from it.

In addition to the above features, the aspect ratio of the symbol has been calculated too:

$$AR_1 = \frac{Height}{Width} \tag{7}$$

As the different shapes of symbols in terms of height and width give this feature great importance in distinguishing between symbols.

To mitigate classification errors caused by variations in fonts and formats, symbols were subjected to a thinning process to make them one pixel wide. Subsequently, the Aspect Ratio AR_2 of the thinned symbol was calculated to address this issue.

3.2.2. Classification

Classification serves as the critical decision-making phase in an Optical Character Recognition (OCR) system. This pivotal stage capitalizes on the features identified and isolated in the preceding step, and these are employed to discern the textual components accurately. It operates on the basis of a set of predefined rules (22).

In essence, during this classification phase, the OCR system scrutinizes the extracted features in a detailed manner, identifying and segregating each piece of data. The extracted features can range from simple shapes to complex patterns. The classification process essentially interprets and translates these features into recognizable characters or symbols. Given its significant role, the classification phase serves as the OCR system's defining point, where it begins to demonstrate its true capability to transcribe the image-based text into a machine-readable format. By applying the predetermined rules and algorithms, the system can recognize and interpret each text segment effectively, paving the way for accurate digitization of textual information (23).

Neural networks are powerful tools for the classification of features extracted from images, including those containing letters in OCR systems. A common type of neural network used in these contexts is a Convolutional Neural Network (CNN), which is designed to automatically and adaptively learn spatial hierarchies of features from the provided input images (24). Two distinct methods of neural networks are used here for the classification.

3.2.2.1. Neural Network Method 1

In this study, the tanh (hyperbolic tangent) activation function was employed. This function is chosen to ensure mapping any real-valued number to a range of [-1, 1]. It shares similarities with the sigmoid activation function but offers an advantage by being zero-centered. This zero-centered property aids helps prevent the network from becoming stuck during training. We used 100 neurons in hidden layers, Regularization, α=30 and Maximal number of iterations is 200. L-BFGS-B (Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bound constraints) is used as a valuable tool for solving nonlinear optimization problems. It aids in refining the weights by minimizing the cost function in the backpropagation process of the neural network. L-BFGS-B is renowned for effectively handling constraints on parameters, making it particularly suitable for problems involving a large number of parameters, such as deep learning (25, 26).

3.2.2.2. Neural Network Method 2

A key part of any neural network training involves a solver, or optimization algorithm. RMSprop (Root Mean Square propagation) is one such optimization algorithm that has gained popularity for training deep neural networks.

RMSprop works by keeping a moving average of the squared gradients for each weight, and then dividing the gradient by the root of this average. This method tries to resolve Adagrad's radically diminishing learning rates by using a moving average of the squared gradient. It utilizes the concept of the weighted moving average of the squared gradients to normalize the gradient itself (27, 28). Mathematically, RMSprop is expressed as follows:

$$E[g^2]_t = \gamma * E[g^2]_{t-1} + (1 - \gamma) * g_t^2 \quad (8)$$

$$\theta_{t+1} = \theta_t - \frac{\eta * g_t}{\sqrt{E[g^2]_t + \varepsilon}} \quad (9)$$

Where:

g_t = the gradient at timestep t,

ε = a very small number to avoid any division by zero, usually on the order of $1e^{-8}$,

γ = decay rate, typically set to 0.9,

η = learning rate.

Here, θ represents the parameters of the model (e.g., the weights and biases in a neural network), and $E[g^2]$ is the moving average of the squared gradient and Hinton suggests $\gamma = 0.9$, with a good default for η as 0.001. The options used for training the model are 100 neurons in hidden layers, Max Epochs is 12, Mini Batch size is 10 and Squared Gradient Decay Factor is 0.9, the model was trained in MATLAB.

3.3 Interpretation of Mathematical Expressions

The neural network approach effectively trains the model to recognize and learn the letters and symbols present in mathematical equations. The result enables the transformation of these equations from an image-based representation into a readily understandable text format. This process facilitates interaction with the equations, supporting further computational or analytical operations.

Once a letter is identified as a component of the equation, the next step is to ascertain its position within the equation whether it is inline, in superscript, or in subscript. To achieve this, three guide lines were used: one in the middle of the equation, one at the topmost part of the equation, and another at the bottom. By computing the difference in the distances between these lines and the midpoint of each character, it can establish to which line the character is closest. The closest line will determine the position of the character in the equation, as illustrated in **Figures 4** and **5**.

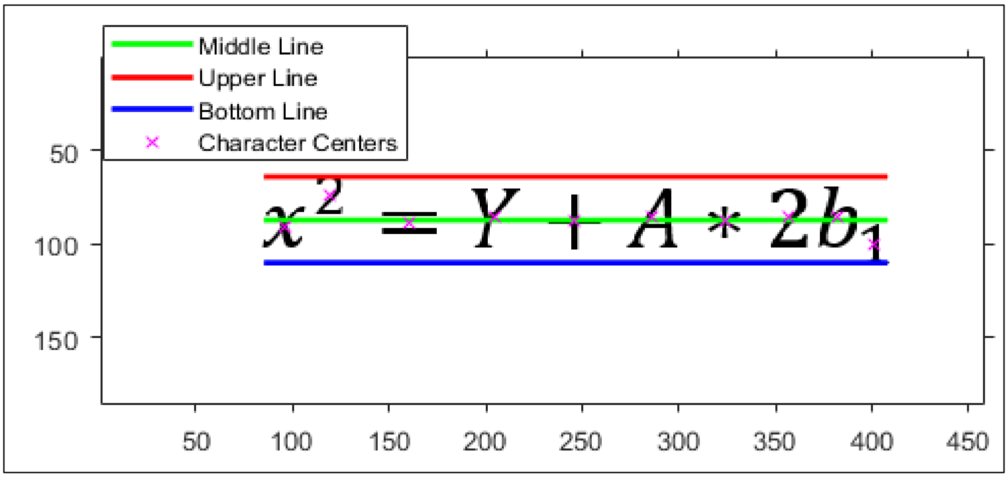

Figure 4. Classification of subscripts, superscripts, and inline elements in a mathematical equation.

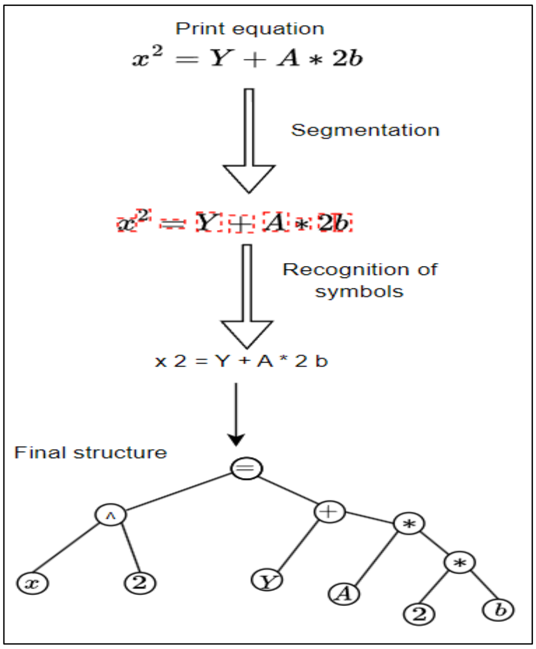

Figure 5. Overview of the recognition process.

4. Experimental Results and Discussion

The proposed system was trained using a dataset consisting of 70 distinct symbols, encompassing capital English letters (A-Z), lowercase letters (a-z), numbers (0-9), and mathematical symbols (=, +, -, *, /, >, <). Each symbol was represented by 50 distinct images varying in sizes, fonts, and styles. Approximately 25 fonts were used, including Arial, Times

New Roman, and others. Furthermore, the system's efficacy was evaluated through the examination of 25 equations of different sizes and contents. These equations used a wide range of letters and symbols, allowing for comprehensive testing. All images of the symbols and equations used in this study are publicly available at:
https://www.kaggle.com/datasets/yahiabaherjaber/print-mathematical-dataset-print-characters
Table 1 presents the training data acquired through two distinct methodologies: Neural Network Method 1 and Neural Network Method 2.

Table 1. Comparison of Training Data Acquired via Neural Network Method 1and Neural Network Method 2

Methods	Test on Train data	Train data 80% Test data 20% Repeat train/test = 5				
		CA	**F1**	**Prec**	**Recall**	**MCC**
Neural Network Method 1	98.8%	94.5%	94.5%	94.8%	94.8%	93%
Neural Network Method 2	97.5%	89.7%	89.5%	90.2%	90.1%	87.7%

The term "testing on data" refers to the practice of training the model on the entire dataset and then selecting a subset of it for testing purposes, effectively evaluating the model on the same data it was trained on. Conversely, the "80% training data and 20% test data" approach involves splitting the total dataset into two parts: 80% is used for training the model, and the remaining 20% is reserved for testing its performance. This train/test procedure is repeated five times to enhance the robustness of the results.
Classification accuracy (CA) refers to the percentage of instances that have been classified correctly by the model, while it is calculated using **Equation 10**:

$$CA = \frac{TP+TN}{TP+TN+FP+FN} \tag{10}$$

Recall, also known as sensitivity, represents the proportion of all positive cases in the dataset that were correctly identified as such, and it is determined based on **Equation 11**:

$$Recall = \frac{TP}{TP+FP} \tag{11}$$

Precision is the proportion of true positives among cases classified as positive, and it is determined based on **Equation 12**:

$$Pr = \frac{TP}{TP+FP} \tag{12}$$

F1 is a harmonic mean between precision and recall, and it is determined based on **Equation 13**:

$$F1 = 2 \times \frac{Pr \times Recall}{Pr+Recall} \tag{13}$$

The Matthews correlation coefficient (MCC), which considers both true and false positives and negatives, is typically viewed as a balanced metric. This measure remains effective even when the sizes of the classes under consideration are significantly different, and it is determined based on **Equation 14**:

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)\times(TP+FN)\times(TN+FP)\times(TN+FN)}} \tag{14}$$

In **Equations 10-14**, True Positive (TP) and True Negative (TN) are correct classifications. False Positive (FP) happens when a sample that is truly negative is predicted as positive. Also, False Negative (FN) happens when a positive sample is predicted as negative (29,30).
Table 2 presents the recognition rates for each mathematical character in a dataset that consists of 25 simple printed mathematical equations.
Moreover, **Table 3** illustrates an assortment of strategies and methodologies employed for the recognition of mathematical symbols and expressions. It details the recognition rates, training samples, and testing samples as reported by various researchers.

Table 2. Recognition Rate of Dataset.

Mathematical Expression Name	Total no. of Characters	Recognized Characters METHOD1	Recognized Characters METHOD2	Recognition Rate METHOD1	Recognition Rate METHOD2
EQ1	6	6	6	100%	100%
EQ2	6	6	6	100%	100%
EQ3	13	13	13	100%	100%
EQ4	6	6	6	100%	100%
EQ5	11	11	11	100%	100%
EQ6	4	4	4	100%	100%
EQ7	7	7	7	100%	100%
EQ8	7	7	7	100%	100%
EQ9	5	4	4	80%	80%
EQ10	5	5	5	100%	100%
EQ11	8	8	8	100%	100%
EQ12	4	3	3	75%	75%
EQ13	7	6	7	85%	100%
EQ14	9	9	9	100%	100%
EQ15	9	9	9	100%	100%
EQ16	6	4	4	66%	66%
EQ17	8	8	8	100%	100%
EQ18	8	8	8	100%	100%
EQ19	8	8	8	100%	100%
EQ20	10	10	10	100%	100%
EQ21	10	9	10	90%	100%
EQ22	6	6	6	100%	100%
EQ23	6	6	6	100%	100%
EQ24	6	6	6	100%	100%
EQ25	5	5	5	100%	100%
	180	174	176	96.6%	97.7%

Table 3. Comparative study for various classification methods.

Methods	Feature Extraction	Classifier	Recognition Rate	Details
Proposed Method 1	See section 3.2.1.	Neural Network	96.6%	activation function= tanh solver = L-BFGS-B
Proposed Method 2	See section 3.2.1.	Neural Network	97.7%	Solver= RMSprop
(31)	Zoning	KNN	72.54%	For single character
	Zoning	SVM	73.02%	Polynomial Kernel
	Transition	KNN	86.57%	
(20)	ZD ,Distance Profile Projection Histograms, BDD	SVM, KNN	95.01%	190 samples
(32)	projection profile histogram	SVM	97.58%	124 no. of mathematical symbols
(33)	___	CNN	83%	ReLU as the activation function
		Neural Network	91%	neural net uses both softmax and sigmoid as the activation functions

According to the above results, it is clear that Method 1 exceeds Method 2 in terms of recognition rates and evaluation metrics. Since it achieved an overall recognition rate of 96.6% compared to 97.7% achieved by Method 2. Both methods have a high level of competitiveness compared with similar studies.

5. Conclusion

The paper introduces a three-step approach aimed at enhancing the recognition of mathematical expressions in scanned documents using OCR systems. The approach involves extracting various features from the vertical and horizontal projection of the segmented symbols. For symbol recognition, the paper suggested two neural network-based classification methods. The first method employs the tanh activation function with the L-BFGS-B optimization method to achieve recognition rates of 96.6%. While the second method utilizes the RMSprop optimization algorithm to achieve recognition rates of 97.7%. Additionally, the paper presents three guidelines for accurately interpreting the mathematical meaning of the expression and converting it into textual form. The experimental results of the paper demonstrate improvements in recognition rates and the accurate interpretation of mathematical expressions for their conversion from image format to textual format.

Acknowledgment

None.

Conflict of Interest

The authors declare that they have no conflicts of interest.

Funding

No funding.

References

1. Memon J, Sami M, Khan RA. Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR). IEEE Access. 2020;8:142642-68. https://doi.org/10.1109/ACCESS.2020.3012542
2. Sakila A, Vijayarani S. Mathematical Symbol Extraction From Document Images A Comprehensive Review. Int J Sci Technol Res. 2020;9(3):648-51. https://doi.org/10.31142/ijstr.v9.i03.2020.25
3. Wang H, Shan G. Recognizing handwritten mathematical expressions as LaTex sequences using a multiscale robust neural network. arXiv:2003.00817. 2020. https://doi.org/10.48550/arXiv.2003.00817
4. Awal A-M, Mouchère H, Viard-Gaudin C. The problem of handwritten mathematical expression recognition evaluation. In: 2010 12th International Conference on Frontiers in Handwriting Recognition; 2010 Sep 19-22; Kolkata, India. IEEE; 2010: 646-51. https://doi.org/10.1109/ICFHR.2010.106
5. Awal AM, Mouchère H, Viard-Gaudin C. Towards Handwritten Mathematical Expression Recognition. In: International Conference on Document Analysis and Recognition; 2009 Jul 26-29; Barcelona, Spain. IEEE; 2009. https://doi.org/10.1109/ICDAR.2009.285
6. Baldominos A, Saez Y, Isasi P. A Survey of Handwritten Character Recognition with MNIST and EMNIST. Appl Sci. 2019;9(15):3169. https://doi.org/10.3390/app9153169
7. Aggarwal R, Pandey S, Tiwari A, Harit G. Survey of mathematical expression recognition for printed and handwritten documents. IETE Tech Rev. 2022;39(6):1245-53. https://doi.org/10.1080/02564602.2021.2008277
8. Huang J, Tan J, Bi N. Overview of Mathematical Expression Recognition. In: Pattern Recognition and Artificial Intelligence, International Conference, ICPRAI; 2020 Oct 17-21; Zhongshan, China. IEEE; 2020. https://doi.org/10.1109/ICPRAI51034.2020.9307062
9. Ogwok D, Ehlers EM. Detecting, Contextualizing and Computing Basic Mathematical Equations from Noisy Images using Machine Learning. In: 2020 The 3rd International Conference on

Computational Intelligence and Intelligent Systems; 2020 Nov 20-22; Online. ACM; 2020. p. 8-14. https://doi.org/10.1145/3440787.3440798

10. Hossain MA, Ali MM. Recognition of handwritten digit using convolutional neural network (CNN). Glob J Comput Sci Technol. 2019;19(2). https://doi.org/10.34257/GJCSTA.190204
11. Alvaro F, Sanchez J-A, Bened J-M. An integrated grammar-based approach for mathematical expression recognition. Pattern Recognit. 2016;51:135-47. https://doi.org/10.1016/j.patcog.2015.09.006
12. Gunawan TS, Noor AF, Kartiwi M. Development of English Handwritten Recognition Using Deep Neural Network. Indones J Electr Eng Comput Sci. 2018;10(2):562-8. https://doi.org/10.11591/ijeecs.v10.i2.pp562-568
13. Veres O, Rishnyak I, Rishniak H. Application of Methods of Machine Learning for the Recognition of Mathematical Expressions. In: COLINS; 2019 Apr 24-25; Lviv, Ukraine. Cham: Springer; 2019. p. 378-89. https://doi.org/10.1007/978-3-030-33671-1_31
14. Nazemi A, Tavakolian N, Fitzpatrick D, Fernando C, Suen CY. Offline handwritten mathematical symbol recognition utilising deep learning. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW); 2019 Jun 16-17; Long Beach, CA, USA. IEEE; 2019. https://doi.org/10.48550/arXiv.1910.07395
15. Chan C. Stroke extraction for offline handwritten mathematical expression recognition. IEEE Access. 2020;8:61565-75. https://doi.org/10.1109/ACCESS.2020.2984180
16. Yan Z, Zhang X, Gao L, Ke Y, Tang Z. ConvMath: a convolutional sequence network for mathematical expression recognition. In: 2020 25th International Conference on Pattern Recognition (ICPR); 2021 Jan 10-15; Milan, Italy. IEEE; 2021:4566-72. https://doi.org/10.1109/ICPR48806.2021.9413203
17. Truong T-N, Nguyen HT, Nguyen CT, Nakagawa M. Learning symbol relation tree for online mathematical expression recognition [Preprint]. arXiv:2105.06084. 2021 [cited 2025 Jun 19]. https://doi.org/10.48550/arXiv.2105.06084
18. MacLean S, Labahn G. A new approach for recognizing handwritten mathematics using relational grammars and fuzzy sets. Int J Doc Anal Recognit (IJDAR). 2013;16:139-63. https://doi.org/10.1007/s10032-012-0183-7
19. Al-Askary YB, Al-Momen S. Tackling Skewness, Noise, and Broken Characters in Mathematical Expression Segmentation. Iraqi J Sci. 2023;64(6):3998-4013. https://doi.org/10.24996/ijs.2023.64.6.38
20. Siddharth KS, Dhir R, Rani R. Handwritten Gurmukhi Numeral Recognitionusing Different Feature Sets. Int J Comput Appl. 2011;28(2):0975-8887. https://doi.org/10.5120/3565-4927
21. Devore JL, Berk KN. Modern Mathematical Statistics with Applications. New York: Springer; 2012. https://doi.org/10.1007/978-1-4614-0391-3
22. Aaref AM. English character recognition algorithm by improving the weights of MLP neural network with dragonfly algorithm. Period Eng Nat Sci. 2021;9(2):616-28. https://doi.org/10.1016/j.jesit.2015.06.003
23. Abdulameer AT. An Improvement of MRI Brain Images Classification Using Dragonfly Algorithm as Trainer of Artificial Neural Network. Ibn Al-Haitham J Pure Appl Sci (IHJPAS). 2018;31(1). https://doi.org/10.30526/31.1.1895
24. Feng Y. Convolutional Neural Network for English Character Recognition. In: CIBDA 2022; 3rd International Conference on Computer Information and Big Data Applications; 2022 Dec 9-11; Wuhan, China. IEEE; 2022. https://doi.org/10.1109/CIBDA56586.2022.00030
25. Chen W, Wang Z, Zhou J. Large-scale L-BFGS using MapReduce. In: Advances in Neural Information Processing Systems 27 (NIPS 2014); 2014 Dec 8-13; Montreal, Canada. Curran Associates Inc.; 2014. https://doi.org/10.5591/978-1-57753-616-9_73
26. Liu DC, Nocedal J. On the limited memory bfgs method for large scale optimization. Math Program. 1989;45:503-28. https://doi.org/10.1007/BF01589116

27. Soujanya B, Sitamahalakshmi T. Optimization with ADAM and RMSprop in Convolution neural Network (CNN): A Case study for Telugu Handwritten Characters. Int J Emerg Trends Eng Res. 2020;8(9). https://doi.org/10.30534/ijeter/2020/38892020
28. De S, Mukherjee A, Ullah E. Convergence guarantees for RMSProp and ADAM in non-convex optimization and an empirical comparison to Nesterov acceleration. In: International Conference on Learning Representations (ICLR); 2018 Apr 30-May 3; Vancouver, BC, Canada. ICLR; 2018. https://doi.org/10.48550/arXiv.1807.06766
29. Dunlop Gr. A Rapid Computational Method for Improvements to Nearest Neighbour Interpolation. Comp & Maths with Appls. 1980;6:349-53. https://doi.org/10.1016/0898-1221(80)90042-5
30. Samadiani N, Hassanpour H. A neural network-based approach for recognizing multi-font printed English characters. J Electr Syst Inf Technol. 2015;2(2):207-18. https://doi.org/10.1016/j.jesit.2015.06.003
31. Mousavi SM, Omid SS, Abu-Bakar SR. Car Plate Segmentation Based on Morphological and Labeling Approach. In: Advances in Computing Control and Telecommunication Technologies; 2011.
32. Gharde SS, Baviskar PV, Adhiya KP. Identification of Handwritten Simple Mathematical Equation Based on SVM and Projection Histogram. Int JSoft Comp Eng (IJSCE), 2013; 3(2).