Combined Proceedings of the 39th iSTEAMS Bespoke Conference – July, 2025
& iSTEAMS Emerging Technologies onference October, 2025

Society for Multidisciplinary & Advanced Research Techniques (SMART - Scientific Projects & Research Consortium (SPaRC)
West Midlands Open University – Projects, Research, Innovations, Strategies & Multimedia (PRISM) Centre
PearlRichards Foundations- Accra Ghana
International Institute for Multidisciplinary and Development Research
Harmath Global Educational Services

39th International Science Technology Education Arts Management
& Social Sciences (iSTEAMS) Bespoke Conference - Accra Ghana 2025

Recognition and Representation of Mathematical Symbols and Equations from Images of Handwritten Equations

Kalu, Grant Kelechi & Woods, Nancy C.
Department of Computer Science
University of Ibadan, Ibadan, Nigeria
Email: grantkalu@gmail.com; chyn.woods@gmail.com
Phone No.: +2348134189679; +2348037273291

ABSTRACT

Mathematical notation is a fundamental means of representing and communicating ideas across scientific disciplines, from pure mathematics and theoretical physics to engineering and computer science. The ability to accurately capture, process, and manipulate mathematical expressions is crucial for various applications, such as scientific document processing, intelligent tutoring systems, and assistive technologies for individuals with disabilities. Interpretation and representation of handwritten mathematical expressions is problematic when it comes to being represented as textual data because of the intricate nature of symbols and equations, ambiguous writing styles, uncertainties in defining spatial relations among others. In addition to this intricate difficulty in representation, the lack of diverse and representative datasets can lead to biases and limitations in the performance of recognition systems, particularly when applied to handwriting from underrepresented regions, such as Africa. The study developed a model for the recognition and representation of Mathematical symbols and equations from handwritten mathematical equations to digital text. A dataset of over 1000 images of handwritten mathematical expressions and equations were gathered from students from several secondary schools in Nigeria. For each image, the LaTeX representation of the contained mathematical expression was typed and used as the image label. The labels were tokenized, and the images were passed through a vision-encoder-decoder architecture fine-tuned for the recognition and representation task. A model for the recognition and representation of the recognized handwritten equation as text was developed and trained using the dataset. The model achieved a validation Character Error Rate of 0.596 and a Bilingual Evaluation Understudy Score of 0.711. These indicate that the model's performance is reasonable, although there is room for improvement by improving the quality of the input images, increasing the size of the dataset and fine-tuning the hyperparameters. The inclusion of primary data from Nigerian secondary school students addresses the lack of diverse and representative datasets, ensuring that the developed model can accommodate the unique handwriting styles and patterns present in the African region, thus fostering inclusivity and accessibility in education, scientific collaboration, and assistive technologies. The model can be applied to any instance of the problem in any global setting.

Keywords: Handwritten mathematical expressions and equations, Vision Encoder-Decoder.

Proceedings Citation Format
Kalu, G. K. & Woods, N. C. (2025): Recognition and Representation of Mathematical Symbols and Equations from Images of Handwritten Equations. Combined Proceedings of the 39th iSTEAMS Multidisciplinary Bespoke Conference 17th–19th July, 2025 & iSTEAMS Emerging Technologies Conference 30th–31st October, 2025. Ghana-Korean Information Resource Centre, Balme Library, University of Ghana, Accra, Ghana. Pp 1-14. www.isteams.net/ghana2025.
dx.doi.org/10.22624/AIMS/ACCRABESPOKE2025P1

1. INTRODUCTION

Mathematical expressions are a cornerstone of scientific communication and problem-solving across various disciplines (Liu et al., 2023). Handwritten Mathematical Symbol Recognition (HMSR) is a subfield of pattern recognition that deals with the automatic recognition and interpretation of mathematical symbols and expressions written by hand. It is a challenging problem due to the wide variety of mathematical symbols, their structural complexity, and the variability in handwriting styles.

The ability to accurately capture, process, and manipulate mathematical expressions is crucial for various applications, such as scientific document processing, intelligent tutoring systems, and assistive technologies for individuals with disabilities, educational software that provides real-time feedback to students, scientific document analysis tools that unlock valuable information from handwritten research notes (Chen & Yan, 2022; Wigington et al., 2022). However, the digitization and processing of handwritten mathematical expressions remain significant challenges in these domains (Álvaro et al., 2014)

The interpretation of handwritten mathematical expressions poses unique challenges due to the complexity and inherent ambiguity of mathematical notation. Unlike natural language, mathematical notation comprises a rich set of symbols, operators, and spatial relationships that convey precise meanings. The recognition process must accurately identify individual symbols while also capturing their structural relationships and adhering to the strict syntax of mathematical equations (Zanibbi & Blostein, (2012); Pratik, A., et al (2023)). Additionally, the handwriting variability among individuals, ranging from writing styles to stroke order and inconsistencies, further exacerbates the complexity of the problem.

Over the past decades, significant research efforts have been devoted to developing methods for recognizing handwritten mathematical expressions (Phan, K., et. al. (2018); Wang, J. et al (2019)). Early approaches relied on statistical models and traditional machine learning techniques, such as hidden Markov models, support vector machines, and decision trees (Chan & Yeung, 2001). These methods typically involved segmenting the input image into individual symbols, extracting relevant features (e.g., shape descriptors, stroke information), and classifying the symbols using trained models (Hu, L., & Zanibbi, R. (2016). The recognized symbols were then combined using syntax rules or structural analysis techniques to reconstruct the complete mathematical expression.

While these traditional approaches achieved promising results, they often struggled with the inherent variability and complexity of handwritten inputs, particularly for more intricate expressions or unfamiliar writing styles. The advent of deep learning techniques, particularly convolutional neural networks (CNNs) and recurrent neural networks (RNNs), has substantially advanced the field of handwritten mathematical expression recognition (Fang & Zhang, 2020). These data-driven approaches can learn hierarchical feature representations directly from raw input data, potentially capturing more complex patterns and nuances in handwritten expressions.

The lack of diverse and representative datasets can lead to biases and limitations in the performance of recognition systems, particularly when applied to handwriting from underrepresented regions, such as Africa. To address the needs of African researchers and end-users of these recognition models, it is crucial to collect and curate primary data from African students and individuals, capturing the unique handwriting styles and patterns present in the region. By incorporating diverse datasets that represent the African demographic, the developed recognition systems can better accommodate the needs of this population, fostering inclusivity and accessibility in education, scientific collaboration, and assistive technologies.

This research aimed to investigate and develop an effective semantic-based recognition model for handwritten mathematical symbols and by extension, formulae and represent them in textual form. Specifically, an image database of offline handwritten equations and symbols was built; a model to accurately identify and represent handwritten mathematical equations as text was developed and the model was then evaluated using metrics such as Character Error Rate (CER) and BLEU (Bilingual Evaluation Understudy) score.

2. LITERATURE REVIEW

Some of the intricate challenges and existing techniques of recognizing and representing handwritten mathematical symbols, entire expressions and equations have been highlighted by some authors (Liu et al., 2023; Bluche et al., 2022). Graves et al. (2016) explored methods for representing these recognized symbols and equations in a way that captures their structure and the underlying mathematical meaning they convey. Figure one shows some simple, yet diverse handwritten representation of $\pi = 3.14$.

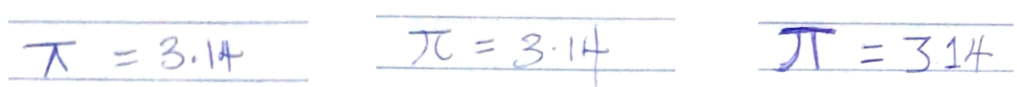

Figure 1: Various handwritten representation of $\boldsymbol{\pi = 3.14}$

From figure one, we see that visually, these images are different in handwriting style as well as symbol representation, which poses a problem in HMSR. In HMSR, once symbols and characters are recognized in an equation, representing the entire equation in a structured format becomes crucial. The two common mathematical equation representation formats are:- Math Markup Language (MathML) which allows for encoding the structure, content, and appearance of equations in a machine-readable format and LaTeX, which enable the storage, exchange, and manipulation of mathematical expressions by computers, facilitating tasks like searching, editing, and analysis. Figure 2 shows the MathML and LaTeX representation of the equation $\pi = 3.14$, this shows how varied the two methods are. One can only image the representation of more complex equations.

<math xmlns="http://www.w3.org/1998/Math/MathML"> <mi>π</mi> </math>	\pi
MathML	LaTeX

Figure 2: A MathML and LaTeX representation of $\boldsymbol{\pi = 3.14}$

2.1 Approaches to Handwritten Mathematical Equation Recognition

Handwritten Mathematical Equation Recognition (HMER) is a relatively young field, but it has seen significant advancements in recent years. Early approaches relied on rule-based systems that attempted to identify individual symbols and their spatial relationships within an equation. Over the past decades, significant research efforts have been devoted to developing methods for recognizing handwritten mathematical expressions. Some of the recognition processes typically involve several steps, including image preprocessing, symbol segmentation, feature extraction, and classification (Nazemi et al., 2019). HMER employs a variety of techniques, each with its own strengths and limitations.

Pattern Recognition Approaches
Early HMER systems relied on pattern recognition techniques. These methods involved defining a set of rules and templates to identify individual symbols based on their shape features. (Bluche et al., 2022). However, pattern recognition approaches generally lack robustness and struggle to handle the complexities of real-world handwritten equations.

Machine Learning Techniques
Machine learning (ML) algorithms have significantly advanced HMER. These techniques learn from large datasets of labeled handwritten symbols and equations, enabling them to recognize patterns and generalize to unseen data. For example, Support Vector Machines (SVMs) can be trained to classify individual symbols based on extracted features and Hidden Markov Models (HMMs) used to capture the sequential nature of symbols within an equation, making them suitable for recognizing connected symbols. Despite their improvements over pattern recognition approaches, ML techniques still face limitations in handling intricate symbols and contextual relationships within equations.

Deep Learning Models
The emergence of deep learning has indeed revolutionized HMER. Convolutional Neural Networks (CNNs) have proven to be well-suited for HMER due to their ability to learn hierarchical features from images. They can effectively extract features like edges, curves, and loops, leading to improved symbol recognition accuracy compared to traditional methods. Recurrent Neural Networks (RNNs) are adept at handling sequential data, making them suitable for recognizing equations that involve connected symbols or nested expressions (Wolf, C. et.al 2023).

They can capture the contextual relationships between symbols within an equation, leading to a more comprehensive understanding of the expression. Attention-based Models focus on specific parts of the input image (equation) that are most relevant for recognizing a particular symbol. This attention mechanism can improve accuracy by directing the model's focus to crucial features while downplaying less important details. Recent advances in deep learning architectures, such as combining CNNs and RNNs (Zhang, J. et. al. 2017), or utilizing transformers with attention mechanisms, have further pushed the boundaries of HMER performance. However, training these models often requires vast amounts of data, and computational resources can be a limiting factor (Álvaro, F., et.al 2012).

2.2 Related Works
Zhelezniakov et. al. (2021) provided a comprehensive overview of state-of-the-art methods for recognizing handwritten mathematical expressions, particularly focusing on online input (i.e., expressions written digitally). The survey categorizes existing recognition methods based on their strengths and weaknesses. It highlights advancements in end-to-end approaches that leverage encoder-decoder architectures and explore the potential of multi-modal input for improved recognition accuracy. Additionally, the paper discusses common evaluation protocols, benchmark datasets like CROHME, and performance metrics like Expression Rate used to assess recognition systems.

Nguyen et al (2016), leveraged deep learning techniques by utilizing max-out based Convolutional Neural Networks (CNNs) and Bidirectional Long Short-Term Memory (BLSTM) networks. CNNs analyze image patterns from online data, while BLSTMs handle the sequential information present in original online drawings (potentially capturing the stroke order). Experiments demonstrate that combining these deep learning architectures outperforms individual models and traditional methods, achieving significantly high HMER accuracy.

Álvaro et. al. (2012) combined Hidden Markov Models (HMMs) and 2D Stochastic Context-Free Grammars (SCFGs) in tackling online HMER. HMMs excel at recognizing individual symbols within the expression because they capture the statistical properties of how a symbol is typically written, accounting for natural variations in handwriting. SCFGs go beyond symbol recognition by modeling the relationships between symbols. They capture the spatial layout and structure of an expression, considering how symbols are positioned and connected relative to each other. This combination allows the model to leverage the strengths of both approaches. The model's effectiveness was validated on the MathBrush3 dataset, a benchmark collection of handwritten mathematical expressions. Notably, the system achieved the best results in a mathematical expression recognition, demonstrating its superior performance and robustness in recognizing both individual symbols and their relationships within the expression.

An end-to-end system for recognizing online handwritten mathematical expressions (OHME) was proposed by Le and Nakagawa (2017). This approach tackled the challenge of OHME recognition by integrating CNNs, Bidirectional Long Short-Term Memory (LSTM) and LSTM with Attention Model. CNN was used to extract features from the handwritten expression image and to capture essential characteristics of the symbols. While the Bidirectional LSTM was used to encode the extracted features, considering the sequential nature of how strokes are drawn in the expression.

In their final stage, they used LSTM with Attention Model to decode the encoded features and generate the corresponding LaTex code, effectively translating the handwritten expression into a digital format. To further enhance performance, the authors introduce local and global distortion models. This process artificially expands the training data, allowing the system to learn from a wider range of variations and improve its generalization capabilities. Their recognition rate increased from 28.09% without generated data to 35.19% using the generated patterns. This research highlights the potential of end-to-end deep learning architectures with data augmentation techniques for achieving high accuracy in handwritten mathematical expression recognition.

The research by Nazemi et. al. (2019) delved into offline recognition of handwritten mathematical symbols, a domain distinct from online recognition which focuses on expressions written digitally. Their approach tackled two key challenges: symbol segmentation and accurate classification for a vast number of classes. Their work utilized Simple Linear Iterative Clustering (SLIC) to segment individual symbols from the handwritten equation. This is crucial, as accurate classification relies on analyzing symbols in isolation. However, the authors acknowledge limitations in segmenting certain symbols, like the root symbol, which might require additional strategies.

For classification, their work explored various feature extraction techniques like Histogram of Oriented Gradients, Local Binary Pattern, and salient features to represent the symbols. These features were then fed into different deep learning classifiers, with SqueezeNet. This research highlights the complexities of offline handwritten mathematical symbol recognition, particularly in segmentation and dealing with a large number of symbol classes.

Our research will explore the use of sematic-based model for offline recognition of handwritten mathematical symbols and by extension, formulae and represent it in textual form.

3. METHODOLOGY

Figure 3 depicts the general architecture of the methodology used in this work. The model architecture utilizes a pre-trained VisionEncoderDecoderModel, specifically "microsoft/trocr-small-printed," tailored for image-to-text tasks. In training the model, the AdamW optimizer was used to save checkpoints, and report training loss. The encoder-decoder architecture employs attention mechanisms, enabling the encoder to process images while the decoder generates corresponding text. Subsequent sections further explains the methodology further.

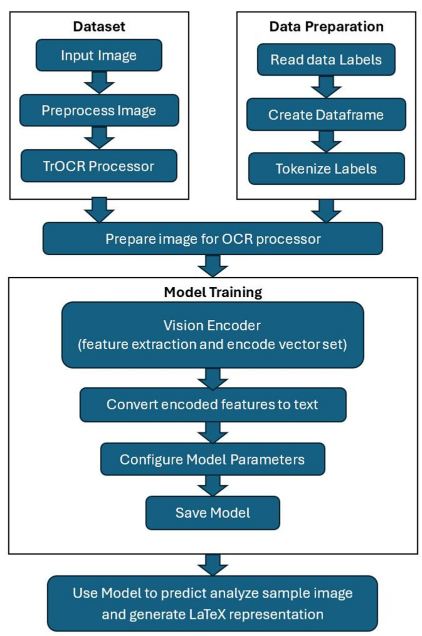

Figure 3: Methodology Architecture

3.1 DATASET

To capture the diverse handwriting styles and patterns present in the African region and to address the lack of diverse datasets representing African handwriting styles, primary data consisting of over 1,000 handwritten mathematical expressions and equations were collected from secondary school students in Nigeria providing a rich source of handwritten mathematical content. The process aimed to capture a wide range of mathematical symbols, operators, and structures to ensure a comprehensive dataset.

This dataset aims to provide a representative sample of handwriting from the African demographic, contributing to the development of an inclusive and accurate recognition system addressing the problem of limited and biased datasets. Figure 4 shows a snapshot of some of the expressions. Each mathematical expression or equation was saved as an image file, and a corresponding label was assigned to represent the textual (LaTeX) representation of the handwritten content. A pandas DataFrame was created to store the handwritten equations and their corresponding labels (textual representations). This DataFrame also included filenames for easy mapping between the images and their labels. Table 1 shows a portion of this dataframe.

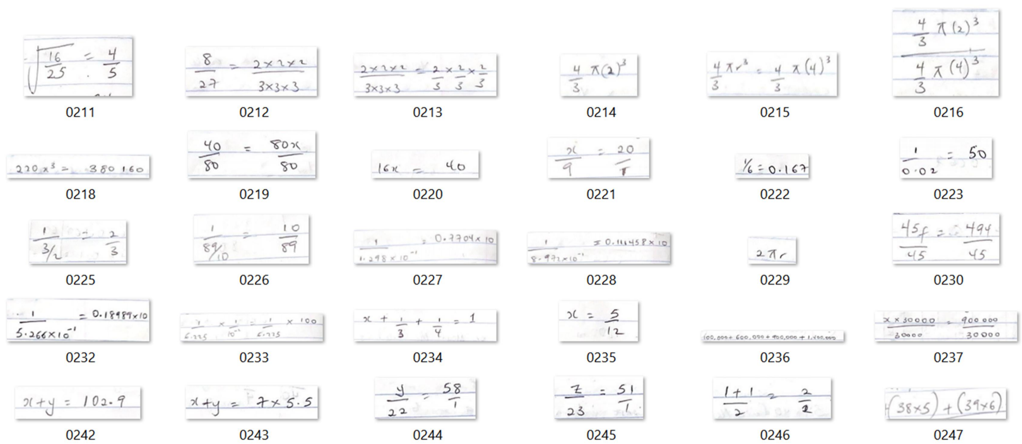

Figure 4: A snapshot of part of the image dataset

TABLE1: A PORTION OF CORRESPONDING LATEX LABELS OF EACH OFFLINE IMAGE IN THE DATASET

S/N	text
1	x = 9 \times 2
2	x = 18
3	\frac{3}{4} = \frac{a}{20}
4	4a = 60
5	4 \times a = 3 \times 20
6	\frac{4a}{4} = \frac{60}{4}
7	a = 15
8	\frac{d- 7}{2} = 5
9	d-7 = 2 \times 5
10	d - 7 = 10
11	d = 10 + 7
12	h = \frac{18 + 5h}{7}

3.2 DATA PREPARATION

The DataFrame was split into training and testing sets, 80% for training and 20% for testing. This step ensures that the model is evaluated on unseen data during the testing phase. The training set was used to train the recognition model, allowing it to learn patterns and relationships between handwritten symbols and their textual representations. Meanwhile, the testing set was held back for evaluating the model's performance on unseen data, simulating real-world usage.

The image files containing the handwritten mathematical expressions underwent a series of processing steps to prepare them as input to the recognition model. This included loading the images using PIL, converting from RGB format to grayscale, using the TROCR processor to apply an internal normalization and resizing and then returning tensor-formatted pixel values. The latex formulas (labels) were preprocessed by stripping whitespaces from the raw text, tokenizing the text using WordLevel tokenizer, handling special tokens ([CLS], [SEP], [PAD], etc.) and replacing padding tokens with -100 for loss calculation.

3.3 Model Training

The pre-processed data underwent feature extraction to transform the handwritten mathematical equations into a suitable format for model training and inference. The labels associated with each image, representing the textual form of the handwritten equations, were encoded using a tokenizer. Tokenization is the process of converting text into a format that can be understood by the model. The tokenizer split the labels into individual tokens (words or symbols) and assigned a unique numerical representation to each token. This numerical representation served as the input to the recognition model, allowing it to interpret and predict the textual form of the handwritten equations.

To expedite the development process and leverage the power of transfer learning, pre-trained models were utilized. Specifically, the Transformers OCR (TROCR) models, which have been trained on large datasets of handwritten text, were employed as a starting point. These models have already learned general patterns and features of handwritten text, making them well-suited for recognizing mathematical symbols and equations. The pre-trained models were fine-tuned using the collected dataset to adapt to the specific task at hand. A vision encoder-decoder model was initialized and configured based on the selected TROCR model. This encoder-decoder architecture is well-suited for tasks involving image-to-text translation, such as handwritten equation recognition.

The feature-extracted data was used to train the configured vision encoder-decoder model using AdamW optimizer and the model's inbuilt loss function. The encoder-decoder architecture is particularly well-suited for sequence-to-sequence tasks, such as translating images of handwritten equations into their textual representations. The encoder processes the input image and extracts relevant features, while the decoder generates the output sequence (textual representation) based on the encoded features. During the training process, the model learned to map the handwritten equation images to their corresponding textual representations.

Additionally, techniques such as Hyperparameter tuning, number of layers, hidden units, and activation functions, were employed to ensure the model's robustness and optimize its performance. The model training utilized an AdamW optimizer with a learning rate of 5e-5, running for 50 epochs with a batch size of 16, and implements beam search with 4 beams for generation with maximum sequence length of 100 tokens and fixed vocabulary size of 600 tokens.

3.4 Model Evaluation metrics

After training, the model's performance was evaluated using Character Error Rate (CER) and Bilingual Evaluation Understudy (BLEU) score. These metrics provided insights into the model's ability to accurately recognize and represent handwritten mathematical symbols and equations. Character Error Rate (CER) measures the percentage of characters that are incorrect, inserted, or deleted compared to the ground truth (correct) text.

It provides a simple and intuitive way to assess the accuracy of a text recognition system at the character level. A lower CER indicates better performance, with a CER of 0 representing perfect recognition (no errors). The formula for calculating CER is as seen in equation 1:

$$\mathrm{CER} = \frac{(\mathrm{S} + \mathrm{D} + \mathrm{I})}{\mathrm{N}} \qquad \text{Eq 1}$$

Where:

S is the number of substitutions (incorrect characters)
D is the number of deletions (missing characters)
I is the number of insertions (extra characters)
N is the total number of characters in the ground truth text

Bilingual Evaluation Understudy (BLEU) Score is a metric used for evaluating the quality of machine translation. However, it can also be applied to text recognition tasks, particularly when dealing with longer sentences or passages. BLEU score considers not only character-level accuracy but also n-gram precision (n-gram refers to sequences of n consecutive characters). In simpler terms, it checks how well the recognized text matches the ground truth in terms of both individual characters and short sequences of characters.

The simplified formula for BLEU is shown in equation 2:

$$\mathrm{BP} \times \exp(\Sigma_{\mathrm{n}=1}^{\mathrm{N}} \ \mathrm{W_n}.\log(\mathrm{P_n})) \qquad \text{Eq 2}$$

Where:

BP is the brevity penalty.
N is the maximum n-gram length considered (usually up to 4).
W_n is the weight assigned to n-grams of length n. Typically, all weights are set to $1/N$.
P_n is the precision for n-grams of length n.

3.5 Training Environment

This research work was carried out using Google Colab, which provides users the ability to write and execute Python code in a collaborative environment similar to Jupyter Notebook. It is a very effective tool for building and presenting interactive data science projects. For this research, the Scikit-learn, pandas, transformers library were used and the codes were written using Python programming language. Figure 5 presents the Python code snippet representing the training loop for the model. This loop iterates over 50 epochs, where each epoch involves training the model on the training dataset.

Key functionalities within the loop

- Data Loading: The train_dataloader efficiently loads and batches training data for each iteration.
- Device Allocation: Input data is transferred to the appropriate device (GPU if available, else, CPU) for faster processing.
- Model Training: The model's forward pass is performed on the data batch, calculating outputs and loss.
- Backpropagation: Gradients of the loss are computed with respect to model parameters using loss.backward().
- Parameter Update: The optimizer updates model parameters based on the gradients using optimizer.step().
- Loss Monitoring: The training loss is tracked and reported periodically (e.g., every report_step batches).

- **Loss History:** Average loss for each epoch is stored in a list for analysis.
- **Model Saving:** After training completion, the trained model is saved for future use or evaluation using model.save_pretrained(). This training loop is essential for developing the model's ability to recognize and represent handwritten mathematical symbols effectively, aligning with the research objectives.

```
# Training loop
for epoch in range(num_epoch):
    # Train the model
    model.train()
    train_loss = 0.0
    for i, batch in enumerate(train_dataloader):
        for k,v in batch.items():
            batch[k] = v.to(device)

        outputs = model(**batch)
        loss = outputs.loss
        loss.backward()
        optimizer.step()
        optimizer.zero_grad()

        train_loss += loss.item()
        if i % report_step == 0:
            print(f"Loss: {loss.item()}")

    loss = train_loss / len(train_dataloader)
    loss_vals.append(loss)
    print(f"Loss after epoch {epoch}:", loss)

# Save the trained model
model.save_pretrained(f"version_{version}/final")
```

Figure 5: Model Training Loop

4. RESULTS AND DISCUSSION

The inclusion of primary data from Nigerian secondary school students addresses the lack of diverse and representative datasets, ensuring that the developed model can accommodate the unique handwriting styles and patterns present in the African region, fostering inclusivity and accessibility in education, scientific collaboration, and assistive technologies. Thus, figure 4 depicts a portion of this image database which displays several examples of handwritten equations. These examples demonstrate the diversity of mathematical expressions captured in the dataset, ranging from simple algebraic expressions to more complex equations involving fractions, exponents, and other mathematical symbols.

A Python function allows the previewing of the images and their corresponding LaTeX label. This function reads a random image from the collected dataset, retrieves its corresponding LaTeX label, and returns both the image and its label. This functionality is crucial for training and evaluating the handwritten mathematical equation recognition model, as it allows for the systematic presentation of image-label pairs to the model during the learning process, this image serves as evidence of the successful collection of a diverse and representative dataset.

This is essential for developing an effective and accurate recognition model for handwritten mathematical symbols and equations. Figure 6 depicts the model's training loss over epochs for the handwritten equation recognition system. On the X-axis, we have the Epoch (complete pass through training data), while on the Y-axis, the Model Loss (that is the difference between predictions and ground truth) are plotted. This loss plot is vital for understanding the model's learning process and potential for accurate recognition. The model starts with a high loss value around 3.0 at epoch 0, indicating poor initial performance. We notice that there is a very steep drop in loss in the first few epochs, showing the model is quickly learning important patterns. After the initial dramatic improvement, the loss continues to decrease but at a gentler slope.

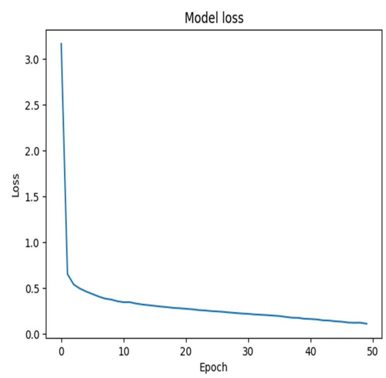

Figure 6: Loss over Epoch graph

The curve then flattens considerably in later epochs (after about epoch 30), suggesting the model has reached a stable state. This learning curve demonstrates healthy model training behavior because, the consistent downward trend shows the model is successfully learning from the data. The smooth curve without spikes or erratic behavior indicates stable training. The asymptotic behavior toward the end suggests the model has effectively converged. The final loss value is quite low, indicating good model fit.

```
  0%|          | 0/14 [00:00<?, ?it/s]
Loss: 0.18172286450862885
Loss after epoch 45: 0.16994825218405044
  0%|          | 0/14 [00:00<?, ?it/s]
Loss: 0.18849831819534302
Loss after epoch 46: 0.1708611473441124
  0%|          | 0/14 [00:00<?, ?it/s]
Loss: 0.15610270202159882
Loss after epoch 47: 0.155237255351884
  0%|          | 0/14 [00:00<?, ?it/s]
Loss: 0.161479651927948
Loss after epoch 48: 0.15237559697457723
  0%|          | 0/14 [00:00<?, ?it/s]
Loss: 0.11750093102455139
```

Figure 7: Training loss after 49 epoch

Figure 7 shows the decreasing training loss demonstrates that the model is effectively learning from the training data, improving its ability to make accurate predictions or translations. The Training Loss starts at a relatively high value of 19.38 and gradually decreases to 0.12 after 49 epochs, indicating the model is learning and improving performance.

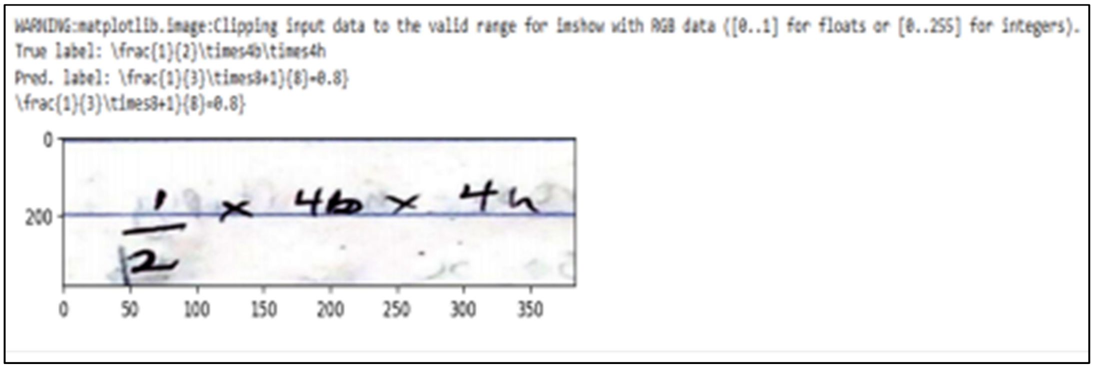

Figure 8: Sample Model Output

Figure 8 shows a sample model output. It shows a sample handwritten equation image displayed along with its true label (line 2) and predicted label (line 3), enabling visual inspection and analysis of the model's performance on a specific instance. This comparison allows for evaluating the model's accuracy and understanding its strengths and limitations in recognizing handwritten mathematical equations. A code snippet is provided, representing the method used to obtain the predicted label from the model. Additionally, This section is crucial in assessing the overall effectiveness of the developed handwritten mathematical equation recognition system and identifying areas for potential improvement. The Model was evaluated using validation CER and BLEU.

Validation CER: The Validation CER (59.6%) indicates that the model made errors in approximately 59.6 out of every 100 characters in the validation dataset. A lower CER indicates better performance, so this result means the model is reasonably accurate in recognizing characters, suggesting room for improvement in character recognition accuracy.

BLEU Score: The BLEU score reported is 0.7111. BLEU scores typically range from 0 (completely wrong) to 1 (perfect match). A higher BLEU score indicates better translation quality, so this result suggests that the model's translations are fairly close to human reference translations.

5. CONCLUSION

This research developed a model for the recognition and representation of offline Handwritten Mathematical expressions and equations as text. The inclusion of primary data from Nigerian secondary school students addresses the lack of diverse and representative datasets, ensuring that the developed model can accommodate the unique handwriting styles and patterns present in the African region, thus fostering inclusivity and accessibility in education, scientific collaboration, and assistive technologies. The model can be applied to any instance of the problem in any global setting.

The ability to recognize and represent handwritten mathematical symbols and equations has significant applications in various fields, including: Education Technology: Automated grading of handwritten math exams, providing immediate feedback to students and reducing grading workload for teachers. Accessibility Tools: Assisting visually impaired individuals by converting handwritten equations into a format they can access through screen readers or Braille displays. Digital Libraries and Archiving: Enabling efficient searching and retrieval of handwritten mathematical documents in digital libraries or historical archives. Scientific Research: Streamlining data entry for researchers who may need to analyze handwritten equations from experiments or historical records.

This research is primarily recommended for scientific researchers and students, who aim to use assistive technologies for effective representation of Mathematical equations, symbols, and expressions in their work. To further improve the model's generalizability, additional extensive and varied Handwritten mathematical equations and expressions need to be added to expand the dataset and incorporated in subsequent work. Future work can also focus on getting better quality images, expanding the dataset size, exploring different network architectures, and validating the model in a Mathematics recognition and representation software. Furthermore, experimenting with various CNN and OCR designs and fine-tuning hyperparameters might enhance overall performance.

REFERENCES

Álvaro, F., Sánchez, J.-A., & Benedí, J.-M. (2014). Recognition of on-line handwritten mathematical expressions using 2D stochastic context-free grammars and hidden Markov models. Pattern Recognition Letters, 35. https://doi.org/10.1016/j.patrec.2012.09.023

Bluche, E., Chaudhuri, B., & Yin, H. (2022, June). A survey on handwritten mathematical symbol recognition. [arXiv preprint arXiv:2206.04623].

Chan, K., & Yeung, D.-Y. (2001). Mathematical Expression Recognition: A Survey. International Journal on Document Analysis and Recognition, 3. https://doi.org/10.1007/PL00013549

Chen, X., & Yan, S. (2022, April). Neural enhanced symbol spotting for mathematical equation recognition. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 36, No. 1, pp. 10762-10769).

Fang, D., & Zhang, C. (2020). Multi-Feature Learning by Joint Training for Handwritten Formula Symbol Recognition. IEEE Access, PP, 1. https://doi.org/10.1109/ACCESS.2020.2979346

Graves, A., Wayne, G., & Liwicki, M. (2016). Handwritten mathematical formula recognition using LSTM networks. In 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 001939-001944). IEEE.

Hu, L., & Zanibbi, R. (2016). Line-of-Sight Stroke Graphs and Parzen Shape Context Features for Handwritten Math Formula Representation and Symbol Segmentation. https://doi.org/10.1109/ICFHR.2016.0044

Le Duc, A., & Nakagawa, M. (2017). Training an End-to-End System for Handwritten Mathematical Expression Recognition by Generated Patterns. https://doi.org/10.1109/ICDAR.2017.175

Liu, Y., Lyu, S., Zhang, W., & Jin, Y. (2023). A review of handwritten mathematical symbol recognition. Pattern Recognition, 164, 107680.

Nazemi, A., Tavakolian, N., Fitzpatrick, D., Fernando, C., & Suen, C. (2019). Offline handwritten mathematical symbol recognition utilising deep learning. https://doi.org/10.48550/arXiv.1910.07395.

Nguyen, D. H., Le Duc, A., & Nakagawa, M. (2016). Recognition of Online Handwritten Math Symbols Using Deep Neural Networks. IEICE Transactions on Information and Systems, E99.D. https://doi.org/10.1587/transinf.2016EDP7102

Phan, K., Le Duc, A., Indurkhya, B., & Nakagawa, M. (2018). Augmented incremental recognition of online handwritten mathematical expressions. International Journal on Document Analysis and Recognition (IJDAR), 21. https://doi.org/10.1007/s10032-018-0306-1

Pratik, A., Chaudhuri, B., & Blumenstein, M. (2023). HOT-NER: A high-order transition-based neural network for nested entity recognition in handwritten mathematical expressions. [arXiv preprint arXiv:2303.13822].

Wang, J.; Du, J.; Zhang, J.; and Wang, Z.-R. (2019). Multi modal Attention Network for Handwritten Mathematical Expression Recognition. 2019 International Conference on Document Analysis and Recognition (ICDAR), 1181–1186.

Wigington, C. A., Price, A. P., & Belongie, S. J. (2022). Deep neural networks for scientific document analysis: A survey. [arXiv preprint arXiv:2206.02223].

Wolf, C., Dalheimer, J., Wu, C., & Yahi, H. (2023). Pensieve: A system for real-time feedback on handwritten equations. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (pp. 1-13).

Zanibbi, R., & Blostein, D. (2012). Recognition and retrieval of mathematical expressions. International Journal on Document Analysis and Recognition, 15(4), 331–357. https://doi.org/10.1007/s10032-011-0174-4

Zhang, J., Du, J., Zhang, S., Liu, D., Hu, Y., Hu, J., Wei, S., & Dai, L. (2017). Watch, Attend and Parse: An End-to-end Neural Network Based Approach to Handwritten Mathematical Expression Recognition. Pattern Recognition, 71. https://doi.org/10.1016/j.patcog.2017.06.017

Zhelezniakov D., Zaytsev V., & Radyvonenko O. (2021). Online Handwritten Mathematical Expression Recognition and Applications: A Survey. IEEE Access. http://dx.doi.org/10.1109/ACCESS.2021.3063413