ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 10, Issue 10, October 2023

Graph-An-Image: An Application for Offline Mathematical Expression Recognition and Graphing using Vision Transformer, CNN and OpenCV

[1] Vidya N*

[1] M.Sc., Data Science, NMKRV College for Women, Bengaluru, Karnataka, India
Corresponding Author Email: [1] vidya.n.nmkrv@gmail.com

Abstract*— This paper studies the topic of detecting handwritten mathematical expressions using two deep learning models - namely the convolutional neural network (CNN) and the vision transformer (ViT). We implement and train these models on a merged dataset comprising EMNIST dataset, HASYv2 dataset and a Mathematical Symbol dataset. We also compare them for the specific use case of graphing mathematical expressions in a single variable. It was found that the CNN performs better for multiple classification evaluation metrics such as accuracy, precision, recall, F1 score, training time, final model size, Cohen's kappa score and Matthew's correlation coefficient. Hence, it was chosen for building the application. The user facing application uses not only the aforementioned model for recognition and detection of the equations in the input images, it also displays a user interface for capturing user input using streamlit, performs morphological operations, noise reduction, and segmentation of the input images using Python code, and displays the graphical form of the equation using SymPy.

Index Terms*— CNN, Handwritten mathematical expression detection, Vision Transformer.

I. INTRODUCTION

The detection of handwritten mathematical expressions has numerous practical applications, one of which is providing immediate feedback to students through automatic grading of tests and assignments. During the COVID-19 pandemic, the world saw a surge in online activities, one of which was online schooling. This greatly increased the demand for tools that allowed overburdened teachers to reduce their cognitive load.

Another application of this idea is in assistive technologies that allow individuals with impairments to provide input in the form of handwriting instead of a traditional keyboard. This idea works with the abilities the above individuals have and can also help ease many communication difficulties they may have.

Additionally, this is highly useful in converting handwritten notes to text - whether they may be postcards, antique books, cheques, or notes. Not only does this store the data optimally, it also makes it simpler to digitally index, archive and retrieve this data by reducing manual work.

II. BACKGROUND

Hand written mathematical equation recognition is one of the most relevant problems that needs to be solved in the present day. This problem has interested many research students and still continues to be a tough problem to solve.

In 2017, Darmatasia and Mohamad Ivan Fanany used CNN with SVM for feature extraction and classification [23]. Data from NIST SD 192nd edition was used both for training and testing. The proposed method which combines CNN and linear SVM using L1 loss function and L2 regularisation achieved a recognition rate better than only CNN. The recognition rate achieved by the proposed method are 98.85% on numeral characters, 93.05% on uppercase characters, 86.21% on lowercase characters, and 91.37% on the merger of numeral and uppercase characters. While the original CNN achieves an accuracy rate of 98.30% on numeral characters, 92.33% on uppercase characters, 83.54% on lowercase characters, and 88.32% on the merger of numeral and uppercase characters.

Joseph James and his team in April 2019, implemented Convolutional neural networks along with XGBOOST model to improve handwritten recognition systems [22]. Here XGBOOST was used as an accurate prediction tool and CNN was doing the feature prediction. The XGBOOST model is evaluated for loss function and regularisation. When XGBOOST was used along with CNN the total computation time and the accuracy rate increased. NIST was the dataset used.

NIST is a special database containing 810,000 separate character images of lower case, uppercase and digits of English language. This research consisted of two parts, a CNN with custom architecture with prepared dataset. Accuracy of prediction was calculated with the same. The second part was the extracted features from CNN were input to XGBOOST classification and the accuracy of classification was calculated. The dataset was divided into 25 % for testing and 75 % for training. It was found that CNN without XGBOOST has an accuracy of 93.42% and CNN with XGBOOST gives accuracy 97.18.

The recognition of handwritten mathematical expressions in offline settings is a prominent research domain within the

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 10, Issue 10, October 2023

broader field of mathematical expression recognition. The offline handwritten mathematical expression recognition (HMER) task is commonly perceived as more challenging when compared to online HMER. This is primarily attributed to the absence of timing information and the presence of more heterogeneity in writing styles.

A work in 2023 by Ujjwal & Anuj, presents a novel approach utilising an encoder-decoder model that incorporates paired adversarial learning [23]. The encoder extracts semantic-invariant features from both handwritten mathematical expression images and their printed mathematical expression counterparts. The utilisation of semantic-invariant features in conjunction with a DenseNet encoder and transformer decoder has facilitated an enhancement in the rate of expression compared to prior research endeavours. When assessed using the CROHME dataset, the team has achieved an approximate 4% enhancement in the performance of the newest CROHME 2019 test set findings.

The user's text revolves around several key concepts in the field of computer vision and machine learning. These concepts include Optical Character Recognition (OCR), Human Micro-Expression Recognition (HMER), deep learning, adversarial learning and transformers, pattern recognition, and expression recognition.

In 2020, S. Vijayprasath undertook a feature extraction method for analysis of hand written characters [3]. A predefined set of 108 samples with 24 alphabets which were handwritten were taken as the training set. The process was binarization and pre-processing for the segmentation stage. Then a neural network system was used as a classifier utilizing the average number of epochs and recognition was carried out.

At the same time Bineesh Jose and K P Pushpalatha also conducted handwritten character recognition research [4]. They provided an overview of different extraction and classification techniques using Malayalam scripts using Deep Learning approaches. They also provide a comparative study of the algorithm and the accuracy as a guideline for other researchers.

Also, in 2020, Khandokar and team from Malaysia showed interest in the Handwritten language recognition [5]. They implemented CNN on a dataset NIST to get 92.91 percent accuracy.This dataset has 1000 images and was used to train the model.

In the year 2021, a group of researchers undertook an offline handwritten text recognition project [25]. This topic has been of interest for many years and still continues to create interest. Here the team introduced the handling of the problem and analysed latest advances and potential direction for future research in this field.

In 2021, H Parikshith and team conducted character recognition of Kannada language using convolution neural networks and transfer learning [2]. They achieved a validation accuracy of 86.92 percent after using VGG16 the accuracy increased to 93.06 percent.

III. MOTIVATION

The main objective of the project is to make the visualisation of a handwritten mathematical expression in a seamless and quick manner, using an intuitive interface. We compare a Convolutional Neural Network (CNN) and a Vision Transformer (ViT) trained on a merged and sampled dataset.

The Extended MNIST dataset is a collection of handwritten characters and digits [19]. It is an extension of the MNIST dataset that contains only the digits (0-9). It contains the 10 digits and the 26 upper and lowercase alphabets.

The HASYv2 dataset (HAndwritten SYmbol v2) contains 32x32 resolution images of 369 classes, including but not limited to the ones in the previous dataset [20].

The Handwritten Math Symbols dataset contains 16 classes namely, the digits (0-9) and some operators (+,-,x,/,=, decimal point) in images of various resolutions.

The merged dataset comprises all of the above datasets. The 13 characters used in this study are the digits 0-9, +, -, and X. Rarer characters were removed based on the anticipated impact on the scope of the application. Each of the 13 classes contains around 650 images in total in the train, test and validation sets. All images were also resized to 28x28 resolution and read as grayscale.

IV. METHODS

The general flow of the application is as follows:
An image is taken as input from the end user.
The image is processed.
The image is split into its constituent parts.
Features are extracted from the image.
The image is recognised.
The output text is generated for each of the split images.
An equation is generated using the output text
The equation is visualised using a graph.

The handwritten mathematical expression detector consists of three parts: the frontend/UI, the detector model and the graphing unit.

The frontend takes care of point 1 above. The Python code takes care of point 2, 3 and 7. The detector model handles point 4, 5, and 6 and the graphing unit takes care of point 8.

For the frontend, we use an application framework called streamlit that allows us implement the logical aspects of the application with Python code, and also use the same to handle the user interface aspects of the application.

For the detector model selection, two such models were evaluated - the CNN and the vision transformer (ViT). Both models were trained on the given dataset and some evaluation metrics were collated and compared.

The graphing unit uses the Python library SymPy - which is a tool for symbolic mathematics calculations.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 10, Issue 10, October 2023

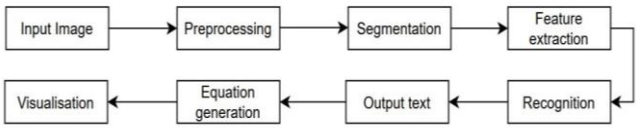

Figure 1: Proposed application flow

Evaluation metrics used

In this section, we discuss some evaluation metrics used to compare the two models.

Accuracy summarizes the performance of a classification model as the ratio of the total number of correct predictions to the total number of predictions.

Precision is the classification metric that measures the actual correctness of the positively classified instances. It is the fraction of the actual positives to the total positives.

Recall is the classification metric that deals with how many real positives were identified correctly. It is the fraction of relevant positives to all positive examples in the dataset.

F1 score measures the model's accuracy. It is the harmonic mean of precision and recall. Accuracy is a useful metric only when you have an equal distribution of classes in your classification. While the classes in our dataset are more or less balanced, some do have more instances than the others. The F1 score balances precision and recall on the positive class while accuracy looks at correctly classified observations both positive and negative.

The precision-recall trade-off represents the fact that precision is inversely proportional to recall. The F1 score gives equal weight to both precision and recall.

Training time refers to the time taken for the model to train.

Model final size refers to how much space it takes up on disk when the model is saved.

Cohen's kappa score is generally used to compare reliability between raters. In the case of checking classification reliability, one rater is taken as the ground truth and the other is taken as the predicted values.

The Matthews correlation coefficient is used in machine learning as a measure of the quality of binary and multiclass classifications. MCC takes into account all four values in the confusion matrix, and a high value (close to 1) means that both classes are predicted well, even if one class is disproportionately under- (or over-) represented. A coefficient of +1 represents a perfect prediction, 0 an average random prediction and -1 an inverse prediction.

CNN Implementation

The CNN was implemented with Keras which is an interface for TensorFlow. The Keras ImageDataGenerator was used to generate images in batches. This function also allows real-time data augmentation which in this case was used to convert images to the correct size in grayscale mode.

```
model = Sequential()

model.add(Conv2D(filters = 32, kernel_size = (5,5),padding = 'Same',
                 activation ='relu', input_shape = (28,28,1)))
model.add(Conv2D(filters = 32, kernel_size = (5,5),padding = 'Same',
                 activation ='relu'))
model.add(MaxPool2D(pool_size=(2,2)))
model.add(Dropout(0.25))


model.add(Conv2D(filters = 64, kernel_size = (3,3),padding = 'Same',
                 activation ='relu'))
model.add(Conv2D(filters = 64, kernel_size = (3,3),padding = 'Same',
                 activation ='relu'))
model.add(MaxPool2D(pool_size=(2,2), strides=(2,2)))
model.add(Dropout(0.25))


model.add(Flatten())
model.add(Dense(256, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation = "softmax"))
```

Figure 2: Python specification for CNN model.

Fig. 2 describes the layers of the CNN in Python code. The Sequential class converts a group of layers into a model, while the Conv2D layer applies the convolutions to the input, and the Dense layer corresponds to a layer in which the neurons are fully connected to the neurons in the next layer.

The Dropout layer is a special layer that helps prevent overfitting by randomly setting input to the layer to be zero. The rest of the inputs are set to *1/(1-given_rate_of_dropout)* such that the sum over all the inputs remains the same.

Overfitting is tuning a model too far on one specific set of inputs, such that the generalisability of the model is lost.

We also use an optimiser in the model, which is a function that reduces the loss by adjusting weights and helps improve a model's performance.

Another concept to consider while implementing a CNN is the learning rate. It indicates how large or small of a step is taken by the optimiser in the direction of the gradient. A higher learning rate could lead to overshooting the optimum and a very small learning rate could lead to very slow convergence on the optimal solution.

There are multiple different types of optimisers that can be used:

Momentum: Helps speed up training by adding a fraction of the previous update to the current update.

RMSprop (Root Mean Square Propagation): Adapts the learning rates of different parameters by scaling them inversely proportional to the recent magnitudes of their gradients.

Adam (Adaptive Moment Estimation): Uses both momentum and RMSProp of the gradient and provides a greater benefit than either.

Adagrad: Adaptive learning rate based on historical gradient.

We are using RMSProp optimiser for the CNN model.

With regards to the loss function used, we are using categorical crossentropy loss. Often used specifically in multiclass classification tasks, categorical crossentropy loss indicates the difference between the predicted class

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 10, Issue 10, October 2023

probabilities and the true class probabilities.

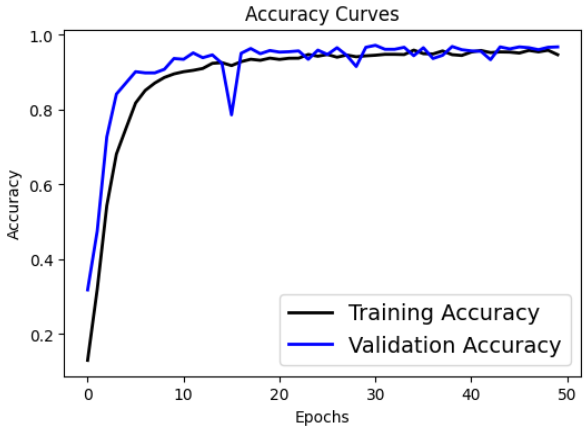

Figure 3: Accuracy curve for CNN.

It also selectively penalises wrong predictions based on their confidence. High confidence wrong predictions are constrained more than lower confidence wrong predictions.

The final CNN related concept covered here is the MaxPool2D layer as seen in Fig. 2 above.

Max-pooling is an operation that selects the maximum value from a given region of the input matrix specified. The max-pooling filter slides over a certain number of columns in the matrix and this is given by the stride configured.

Max-pooling confers several advantages such as reducing computational load and reducing the risk of overfitting by pooling multiple features together.

CNN Results

The accuracy curve in Fig. 3 shows a graph consistent with a good fit of the data.

The loss curve in Fig. 4 also shows a good fit of the CNN model with the problem.

The other metrics, namely, precision, recall, F1 score, specificity and misclassifications are calculated for the CNN and shown in Table 1. Similarly, the training time and model final size were observed and Cohen's kappa score and Matthew's correlation coefficient were calculated for the CNN and added to Table 2.

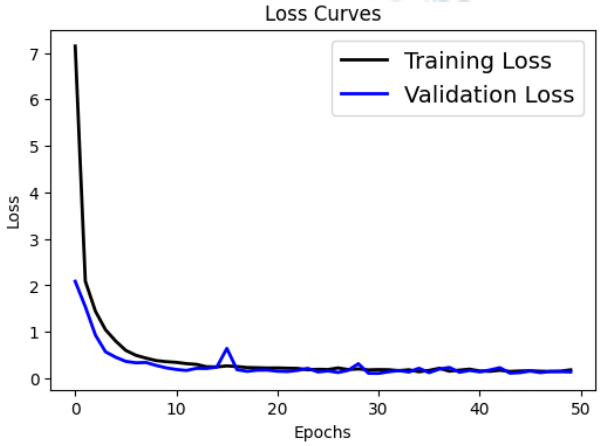

Figure 4: Loss curve for CNN.

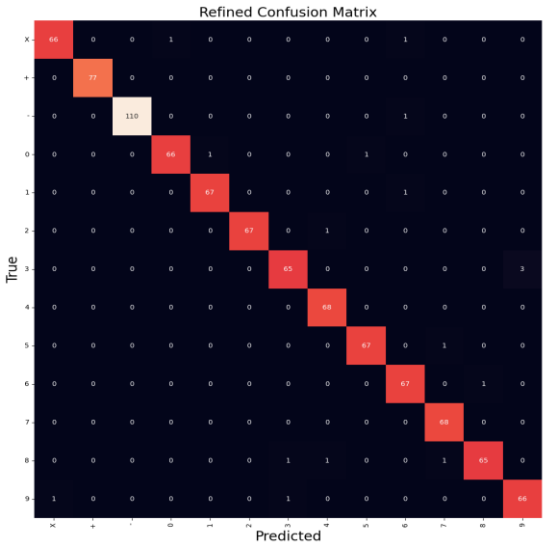

Figure 5: Overall confusion matrix for CNN.

Fig. 5 shows the overall confusion matrix plotted for all of the 13 classes for the CNN.

ViT Implementation

The steps in the process of training a vision transformer are:

Splitting an image into patches of fixed size

Flattening the image patches

Creating lower-dimensional linear embeddings from these flattened image patches

Including positional embeddings.

Feeding the sequence as an input to a state-of-the-art transformer encoder

Pre-training the ViT model with image labels, which is then fully supervised on a big dataset

Fine-tuning the downstream dataset for image classification.

We fine-tuned the pre-trained ViT (ViT-B/16) on the same dataset as the CNN.

We are using the Adam optimiser for the ViT model.

ViT Results

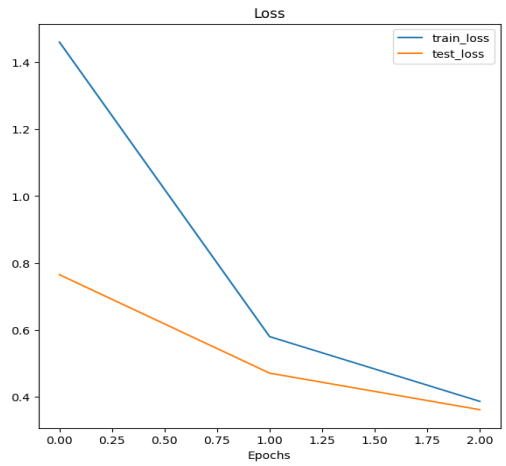

Figure 6: Loss curve for ViT.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 10, Issue 10, October 2023

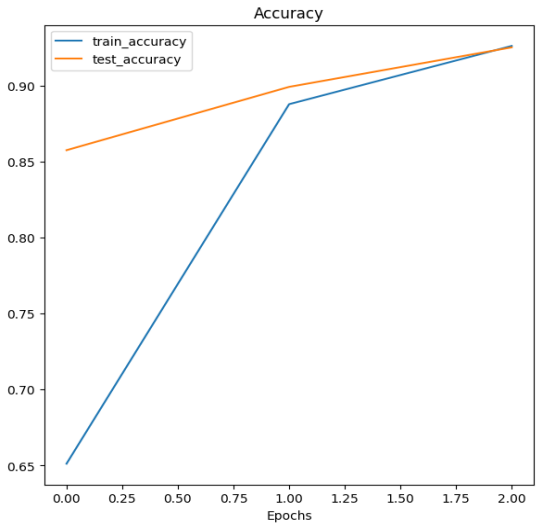

Figure 7: Accuracy curve for the ViT.

The accuracy curve in Fig. 7 seems to show low overfitting of the data to the model. Due to training time, there were issues with running this model for more epochs to get a better fit. The loss curve in Fig. 6 also seems to show a good fit of the model to the problem. Due to training time, there were issues with running this model for more epochs to get a better fit.

The other metrics, namely, precision, recall, F1 score, specificity and misclassifications are calculated for the ViT and shown in Table 1. Similarly, the training time and model final size were observed and Cohen's kappa score and Matthew's correlation coefficient were calculated for the ViT as well and added to Table 2.

Fig. 8 shows the overall confusion matrix plotted for all of the 13 classes for the ViT.

Comparison of CNN and ViT results

Table I and II detail the metrics for the CNN and the ViT in comparison to one another.

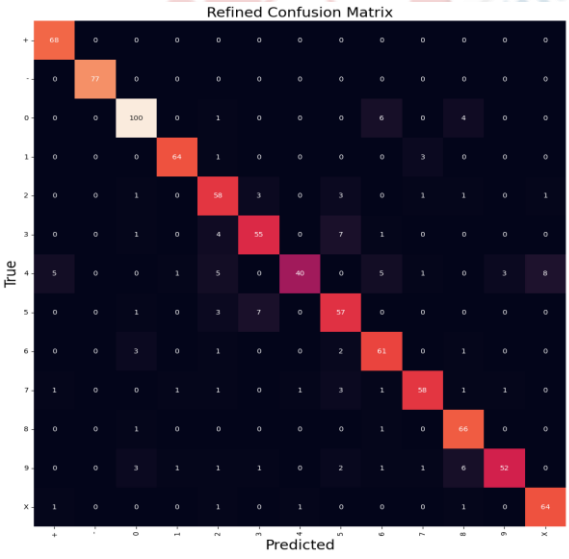

Figure 8: Overall confusion matrix for the ViT.

Table I: Evaluation metric comparison table - 1

	accuracy	misclassified	precision	recall	specificity	F1	
0	0.978	0.022	0.909	0.901	0.988	0.905	vit
0	0.997	0.003	0.985	0.971	0.999	0.978	cnn
1	0.993	0.007	0.955	0.941	0.997	0.948	vit
1	0.998	0.002	0.985	0.985	0.999	0.985	cnn
2	0.97	0.03	0.763	0.853	0.979	0.806	vit
2	0.999	0.001	1	0.985	1	0.993	cnn
3	0.974	0.026	0.833	0.809	0.987	0.821	vit
3	0.995	0.005	0.97	0.956	0.998	0.963	cnn
4	0.968	0.032	0.952	0.588	0.998	0.727	vit
4	0.998	0.002	0.971	1	0.998	0.986	cnn
5	0.97	0.03	0.77	0.838	0.98	0.803	vit
5	0.998	0.002	0.985	0.985	0.999	0.985	cnn
6	0.976	0.024	0.803	0.897	0.983	0.847	vit
6	0.996	0.004	0.957	0.985	0.997	0.971	cnn
7	0.983	0.017	0.906	0.853	0.993	0.879	vit
7	0.998	0.002	0.971	1	0.998	0.986	cnn
8	0.983	0.017	0.825	0.971	0.984	0.892	vit
8	0.996	0.004	0.985	0.956	0.999	0.97	cnn
9	0.979	0.021	0.929	0.765	0.995	0.839	vit
9	0.995	0.005	0.957	0.971	0.997	0.964	cnn
-	1	0	1	1	1	1	vit
-	0.999	0.001	1	0.991	1	0.995	cnn
+	0.993	0.007	0.907	1	0.992	0.951	vit
+	1	0	1	1	1	1	cnn
X	0.986	0.014	0.877	0.941	0.99	0.908	vit
X	0.997	0.003	0.985	0.971	0.999	0.978	cnn

We can see from Table I that the CNN slightly outperforms the vision transformer for this use case.

Table II: Evaluation metric comparison table - 2

	training time (mins)	model size(MB)	kappa	MCC
CNN	10	3.4	0.98028	0.98029
ViT	360	327.4	0.86546	0.86611

For these metrics as well, the CNN outperforms the ViT for this use case as seen in Table II.

Final application flow

Since the CNN outperforms the ViT as above, we use the CNN in the following flow.

In the application, an image of a mathematical expression is taken from the end user. The image is split into its constituent parts (digits and operators and characters) and resized in such a manner than aspect ratio is maintained in order not to lose image information. Each resized image is sent to the model for prediction.

A string equation is generated from each of the predicted values and used to create the graph as below using SymPy.

The application then displays the graph to the end user in the user interface.

This flow is described in Fig. 9.

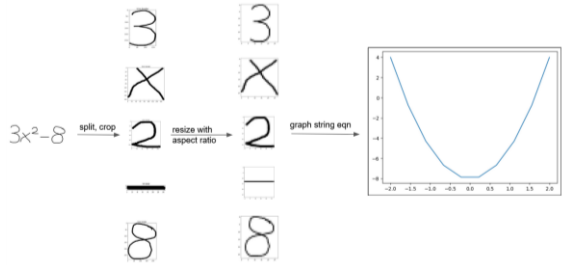

Figure 9: Application flow for the image.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 10, Issue 10, October 2023

V. DISCUSSION

For this particular use case, we see that the CNN performs better than the vision transformer for the evaluation metrics calculated and compared.

CNNs are computationally efficient making them more practical for real-time and heavily resource-contrained environments. In this case we also find that the training time and model size are favourable for such deployments.

However, in the literature, we see that the vision transformer performs better than CNNs. Some reasons for the worse performance may include:

The dataset on which it was fine-tuned was too narrow or too different from the original dataset on which it was pre-trained, leading to the model forgetting the general features previously learnt.

The vision transformer may not be the best model for this application as the images have fewer global characteristics which it could use for prediction.

The high training duration could be due to hardware limitation, unoptimised hyperparameters, large batch sizes or inefficient code.

VI. CONCLUSION

Handwritten mathematical expression detection is a lucrative field with important work being done regularly. We have compared CNNs and Vision Transformers for this task using a merged combination of datasets and found for our task with our system configurations, the CNN performed better.

Some unresolved issues that can be addressed in future work:

Ambiguous symbol handling: When handwritten, many characters may look similar. There should be an improved method of detecting the characters also informed by the context of the character within the expression.

Use of more classes within the dataset: Having more training data could improve outcomes.

Usage of the whole mathematical expression for training: This may be a better use case for the vision transformer as it would make use of the global.

Better image preprocessing pipeline: Improvements here can lead to better outcomes in the classification tasks implemented using the models.

The application deployment strategy: Persistent deployment in the form of a web app or mobile app and a better storage and retrieval pipeline for the images can also go a long way in making the application more usable.

REFERENCES

[1] Mr. Pratik U. Patil, and Mr. Zaid. “Handwritten Mathematical Expression Recognition And Grading System.” International Journal of Engineering Applied Sciences and Technology, 2021 Vol. 6, no. 3,ISSN No. 2455-2143 (July 2021).

[2] Parikshith, H, S M Naga Rajath, D Shwetha, C M Sindhu, and P Ravi. “Handwritten Character Recognition of Kannada Language Using Convolutional Neural Networks and Transfer Learning.” IOP Conference Series: Materials Science and Engineering 1110, no. 1 (March 1, 2021): 012003. https://doi.org/10.1088/1757-899x/1110/1/012003.

[3] Vijayprasath, S. “A Simple Feature Extraction Method for Analysis of Hand Written Characters.” Journal of Physics: Conference Series 1717, no. 1 (January 1, 2021): 012066. https://doi.org/10.1088/1742-6596/1717/1/012066.

[4] Jose, Bineesh, and K. P. Pushpalatha. “Intelligent Handwritten Character Recognition For Malayalam Scripts Using Deep Learning Approach.” IOP Conference Series: Materials Science and Engineering 1085, no. 1 (February 1, 2021): 012022. https://doi.org/10.1088/1757-899x/1085/1/012022.

[5] Khandokar, I, M Hasan, F Ernawan, S Islam, and M N Kabir. “Handwritten Character Recognition Using Convolutional Neural Network.” Journal of Physics: Conference Series 1918, no. 4 (June 1, 2021): 042152. https://doi.org/10.1088/1742-6596/1918/4/042152.

[6] Garain , Mr. Utpal . “Https://Link.Springer.Com/Chapter/10.1007/978-1-84628-726-8_11.” In OCR of Printed Mathematical Expressions, Https://link.springer.com/chapter/10.1007/978-1-84628-726-8_11., n.d.

[7] Tope, Prathamesh. Recognition of Handwritten Mathematical Expression and Using Machine Learning Approach. E-ISSN: 2395-0056. Vol. 08 Issue: 12. International Research Journal of Engineering and Technology (IRJET) , 2021.

[8] Salma Shofia Rosyda and Tito Waluyo Purboyo ,“A Review of Various Handwriting Recognition Methods”,International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 2 (2018) pp. 1155-1164

[9] Dae Hwan Kim, Jin H. Kim, “Top-down search with bottom-up evidence for recognizing handwritten mathematical expression”, 2010 12th International Conference on Frontiers in Handwriting Recognition

[10] C. Lu and K. Mohan. “Recognition of Online Handwritten Mathematical Expressions” cs231n Project Report Stanford (2015).

[11] Lyzandra D’souza and Maruska Mascarenhas, “Offline Handwritten Mathematical Expression Recognition using Convolutional Neural Network”, 2018 International

[12] Conference on Information, Communication, Engineering and Technology (ICICET) Zeal College of Engineering and Research, Narhe, Pune, India.

[13] Al Hamad H.A, 8-10 Oct. 2013, “Use an efficient neural network to improve the Arabic handwriting recognition” , IEEE International Conference on Signal and Image Processing Applications, Melaka, pp 269 – 274, ISBN: 978-1-4799-0267-5.

[14] Jakjoud, W, Lazrek, A., 7-9 April 2011, “Segmentation method of offline mathematical symbols”, IEEE International Conference on Multimedia Computing and Systems (ICMCS), Ouarzazate, pp. 1 – 7, ISBN: 978-1-61284-730-6.

[15] DOROTHEA BLOSTEIN† and ANN GRBAVEC,1996,“ Recognition of mathematical notation”, Handbook on optical character recognition and document image analysis, World scientific publishing company, 1996

[16] Tom M. Mitchell, 1997, International edit “ Machine Learning” McGraw-Hill Science/Engineering/Math ISBN:0070428077.

[17] N. VenkateswaraRao, Dr. A. Srikrishna, Dr. B. RaveendraBabu, G. Rama Mohan Babu, October 2011, “ An

efficient feature extraction and classification of handwritten digits using neural networks", International Journal of Computer Science, Engineering and Applications (IJCSEA) Vol.1, No.5, DOI : 10.5121/ijcsea.2011.1505

[18] Ahmad-Montaser Awal, Harold Mouchere, Christian Viard-Gaudin, 2009, "Towards Handwritten Mathematical Expressions Recognition", IEEE 10th International Conference on Document Analysis and Recognition, ICDAR 2009, Jul 2009, Barcelone, Spain. pp.1046-1050, 2009

[19] Cohen, G., Afshar, S., Tapson, J., & van Schaik, A. (2017). EMNIST: an extension of MNIST to handwritten letters. Retrieved from http://arxiv.org/abs/1702.05373

[20] Martin Thoma (2017)."The HASYv2 dataset" (https://arxiv.org/abs/1701.08380)

[21] Sagyam Thapa (2020) "Handwritten Math Symbols". Retrieved from Kaggle. (https://www.kaggle.com/datasets/sagyamthapa/handwritten-math-symbols)

[22] Jamess, Joseph & Lakshmi, C & James, Joseph. (2019). An Efficient Offline Hand Written Character Recognition using CNN and Xgboost. 8. 2278-3075.

[23] Darmatasia and Mohamad Ivan Fanany. "Handwriting recognition on form document using convolutional neural network and support vector machines (CNN-SVM)." *2017 5th International Conference on Information and Communication Technology (ICoIC7)* (2017): 1-6.

[24] Thakur, Ujjwal & Sharma, Anuj. (2023). Offline handwritten mathematical recognition using adversarial learning and transformers. International Journal on Document Analysis and Recognition (IJDAR). 10.1007/s10032-023-00451-w.

[25] Wang, Yintong & Xiao, Wenjie & Li, Shuo. (2021). Offline Handwritten Text Recognition Using Deep Learning: A Review. Journal of Physics: Conference Series. 1848. 012015. 10.1088/1742-6596/1848/1/012015.