International Journal of Computer Trends and Technology
ISSN: 2231-2803 / https://doi.org/10.14445/22312803/IJCTT-V68I4P138
Volume 68 Issue 4, 248-252, April 2020

Original Article

Handwritten Mathematical Equations Conversion to LaTeX Equivalent

Pooja Chaudhari[1], Bhagirath Prajapati[2], Priyanka Puvar[3]

[1]*Undergraduate Student, Dept. of Computer Engineering, A. D. Patel Institute of Technology, Anand, Gujarat, India*
[2,3] *Associate Prof., Dept. of Computer Engineering, A. D. Patel Institute of Technology, Anand, Gujarat, India*

Received Date: 22 March 2020
Revised Date: 04 May 2020
Accepted Date: 05 May 2020

Abstract *- Recently, in academics, the usage of digital documents has been increasing so that it has been necessary to digitise most of the documents. With the increasing usage of digital documents, the demand for converting hand-written mathematical equations into digital form has been increased.*

Nowadays, LaTeX is popular for academic, scientific or technical articles. LaTeX is a tool for document formatting. LaTeX facilitates writing mathematical expressions by remembering the syntax, but it is difficult to remember syntax all the time.

This problem can be solved with the concept of Deep Learning. Using which a model can be trained with a relative dataset, then the trained model is used to detect mathematical expression, which can further be converted into LaTeX syntax as required by the user.

Keywords *- Mathematical expression, LaTeX, Neural networks, Deep Learning.*

I. INTRODUCTION

A mathematical expression can be transformed into digital format in various ways like using LaTeX or C strings, MATLAB or mathematical editor such as embedded in MS word. But these are time-consuming even for a single expression. So that mathematical materials are very difficult and time-consuming to be digitised [1]

With the increasing demand for digital documents, there should be a way available for automatically recognising mathematical symbols from equations inputs by the user and then converting them into LaTex equivalent. Here in this article, we propose the solution of converting handwritten mathematical equations into LaTex format online while preparing documents into digital form by using the concept of Deep Learning and Neural networks.

The mathematical expressions are recognized by using the given steps:

- Recognizing a single mathematical symbol using a Neural network.
- Recognizing the relation between all symbols by using their traces on the interface
- Converting the symbols into LaTeX and arranging them to form complete mathematical expressions.

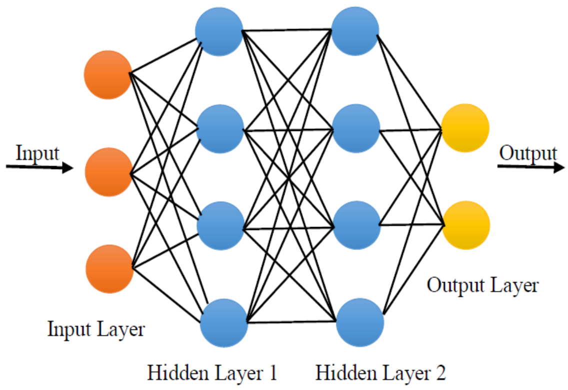

Fig.1 A sample neural network

Deep learning models focus on the right features by themselves with very little human intervention. Neural networks are used by Deep Learning to simulate human-like decision making.[6]

As shown in the above Fig.1, a neural network consists of small units called neurons(shown in the figure using circles).[5] A neural network consists of multiple layers having multiple neurons. A neuron in terms of neural networks is a mathematical function that classifies the data as per the architecture of the neural network model.[7]

II. IMPLEMENTATION

The aim of this work is to make an interface that can be used to recognise handwritten mathematical expressions and provide their LaTeX equivalent to the users, which can be used in LaTeX documents. This problem is approached by using Neural networks in order to get higher accuracy.

The user gives input of handwritten mathematical expressions from the user interface. The input is sent to the trained model, which is used for detecting symbols in the mathematical expression. Then these symbols are combined and converted to form a mathematical expression in LaTeX syntax.[4]

A. Dataset Description

We have used the mathematical symbols dataset provided by CROHME (Competition on Recognition of Online Handwritten Mathematical Expressions). This dataset is a collection of mathematical handwritten symbols that can be used to form mathematical expressions. [3]

The dataset consists of ink ml files that contain the traces of the mathematical symbols gathered by taking input from multiple users across the world. This dataset is used to train the neural network model used for recognizing mathematical expressions.

This dataset also provides a ground truth file which has a list of all the ink ml files mapped to the symbols they are representing. Our main focus would be on the ink ml files, and the ground truth file is used for mapping traces to their symbols which are used to recognize a mathematical symbol. The ink ml file contains traces of mathematical symbols as shown below:

```
<ink xmlns="http://www.w3.org/2003/InkML">
<traceFormat>
<channel name="X" type="decimal"/>
<channel name="Y" type="decimal"/>
</traceFormat><annotation type="UI">ISICAL_2014_F18_E1_24</annotation>
<trace id="0">
272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 2
90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 9
272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 272 90, 2
90, 272 90, 272 90, 272 90, 272 90, 274 90, 274 90, 274 90, 275 92, 275 92, 275 9
277 93, 278 95, 278 95, 278 95, 278 95, 278 95, 279 96, 279 96, 279 96, 282 97, 2
</trace>
<traceGroup>
        <traceGroup xml:id="0">
                <traceView traceDataRef="0"/>
        </traceGroup>
</traceGroup>
</ink>
```

Fig. 2 Sample ink ml file of CROHME dataset

These ink ml files of the dataset, as shown in Fig. 2, contains traces of different mathematical symbols which are in the form of x and y coordinates of the total trace of the symbol from start to end.

One ink ml file contains a trace Format tag which has the axes X and Y for the traces. Then it has traces in the format of x, y which are traces for the given symbol. There might be multiple trace tags in an ink ml file according to different mathematical symbols which require multiple traces.

There are a total of 19224 such ink ml files consisting of traces for multiple mathematical symbols. The input is taken from the canvas on the web application also gets traces of the symbols and uses it for getting output in the form of a LaTeX equation.

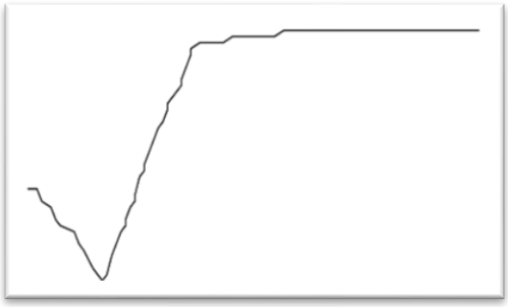

Fig.3 Symbol formed from traces of ink ml file

B. Data Pre-Processing

The training data used in the project is the CROHME dataset which contains ink ml files for different mathematical symbols. These ink ml files contain a set of traces for the specific symbols.

Before training the model, each ink ml file present in the dataset is mapped with its ground truth (the actual symbol it is representing) which is present in the ground truth file given with the dataset. This mapping is stored and used for recognizing mathematical symbols.

The input given by the user is taken in the form of traces of different symbols in the expression, which are then converted into a specific format which is then used to predict the symbol using a trained neural network model.[2]

C. NN Model

The NN model for recognition of mathematical symbols and ultimately mathematical expressions is constructed using TensorFlow API for python. The model defined uses the Neural Net module of TensorFlow. This model contains multiple layers, which help in the recognition of mathematical symbols.

Layer 1- It is the convolutional layer that extracts distinctive features from the given input matrix, which is a normalized image of the symbol. This layer creates a feature map for the image. It uses ReLU (rectified linear unit) as the activation function.

Layer 2- The second layer is a max-pooling layer. The sole purpose of max-pooling is to reduce the spatial size of the image, thereby reducing the number of trainable parameters.

Layer 3- The third layer is again the convolutional layer which extracts more features, thus improving the model's accuracy and using ReLU (rectified linear unit) as the activation function. After using max-pooling to reduce the spatial size, this convolutional layer further extracts more definite and distinct features of the image.

Layer 4- This is again a max-pooling layer to further narrow down the image size for more specific and correct prediction.

Layer 5- This is the flattening layer that converts all elements feature map matrices into individual neurons that serve as input to the next layer.

Layer 6- This is the dropout layer. The main function of this layer is to remove unwanted features or neurons that can increase training time and make the model bulky.

In the end, the neurons are activated using SoftMax. The equation for the SoftMax function is given below.

$$\left\{ \sigma(z)_{\{j\}} = \frac{\left\{ e^{\{z_{\{f\}}\}} \right\}}{\left\{ \sum_{\{k=1\}}^{\{k\}} e^{\{z_{\{k\}}\}} \right\}} \text{ for } j = 1, \ldots, k \right\}$$

SoftMax is a logistic classification function that is used for multiclass classification. It calculates probabilities for each target class overall target classes. The target class can be later easily identified using these calculated probabilities for given inputs. SoftMax provides us with a range of 0 to 1 with the sum of all probabilities equal to 1. The class with the highest probability is the target class.

Fig. 4 shows the block diagram of the above described neural network. It shows the inputs and outputs to various layers, which finally gives the predicted symbol.

D. Training

The neural network model is trained and optimized with Adam optimizer. Adam optimizer is one of the common optimizers used for stochastic gradient descent (SGD), which optimizes an objective function with suitable smoothness properties. This is used to increase the accuracy of the model for prediction. The model was evaluated by the test data provided by CROHME. The test accuracy using this optimization technique for the 6-layer model described above is 85%.

E. LaTeX expression

The above model predicts the mathematical symbols from the given input expression. But there is a particular syntax for writing mathematical expressions in the LaTeX typesetting system. So, this work first recognizes the symbols present in the input mathematical expression and then converts the mathematical symbols into a complete expression with proper LaTeX syntax. The symbols are then converted and arranged in proper form to get the mathematical expression which can be directly copied and used in the LaTeX documentation wherever needed. Thereby reducing the time used in writing mathematical expressions in LaTeX.

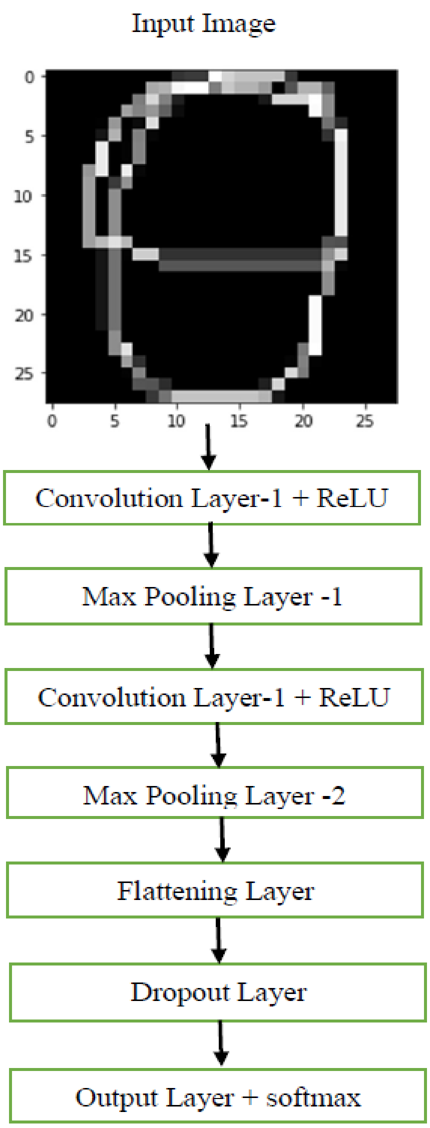

Fig. 4 Layers of Neural Network

F. User Interface

The user interface is a web application that has a canvas, as shown in Fig. 5, on which the user writes mathematical expressions. This canvas stores the traces/positions of the symbols and sends them to the backend for processing. After processing, the output is shown to the user in the form of a LaTeX equation.

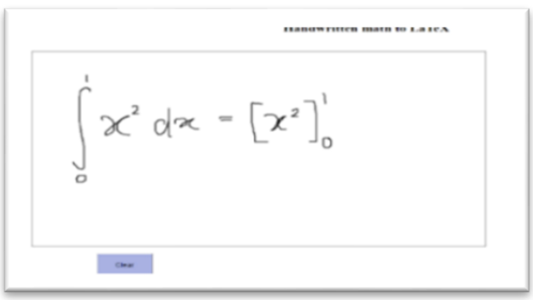

Fig. 5 User Interface (input)

The output is given to the user, as shown in Fig. 6 right next to the input canvas.

Latex : \int_0^{1} x^2 dx = [x^2]

Math output : $\int_0^1 x^2 dx = [x^2]_0^1$

Fig.6 User Interface (output)

III. RESULTS AND ANALYSIS

For recognizing a single symbol, the trace (x and y coordinates of pen/mouse) for the symbol is taken from the user interface provided, as shown in Fig. 7.

Fig. 7 User input – single symbol

The traces are converted into NumPy array format, which can be plotted as an image, as shown in Fig.8. This image is similar to the input given by the user.

This NumPy array is sent to the classifier from the neural network model. The classifier predicts the mathematical symbol. The recognized symbol is then converted into LaTeX equivalent and sent to the user interface as an output, as shown in Fig. 9.

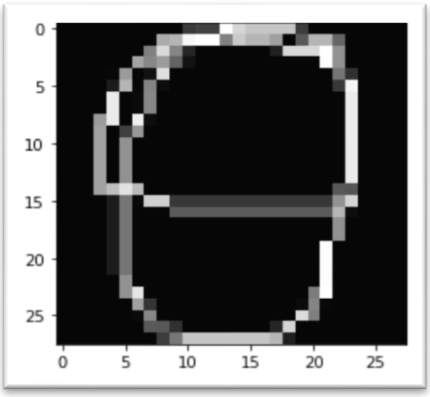

Fig. 8 Image plotted from user input

Latex : \theta

Math Output : θ

Fig. 9 Output for a single symbol

The same method is used to predict mathematical expressions with multiple mathematical symbols. After predicting all the symbols, the predictions and their traces are used to recognise the relationships among the symbols for forming the complete mathematical expression. This mathematical expression is formed and converted into LaTeX format, which is sent to the user interface as an output. Examples of some mathematical expressions inputs and their corresponding outputs are shown in Fig 10 to Fig. 15.

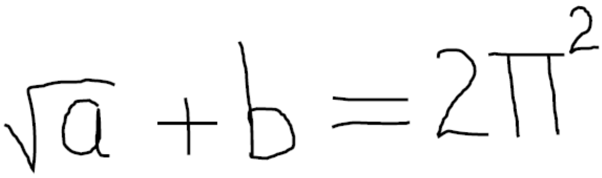

Fig.10 User input – Mathematical expression-1

Latex : \sqrt{a}+b= 2\pi^{2}

Math output : $\sqrt{a} + b = 2\pi^2$

Fig.11 Output mathematical expression- 1

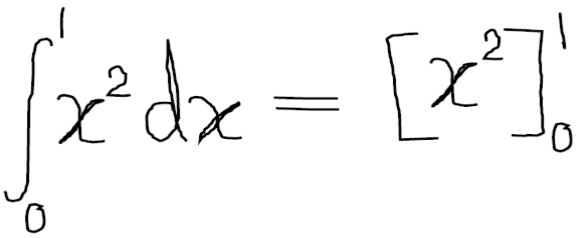

Fig.12 User input – Mathematical expression-2

Latex : \int_0^{1} x^2 dx = [x^2]

Math output : $\int_0^1 x^2 dx = [x^2]_0^1$

Fig. 13 Output mathematical expression- 2

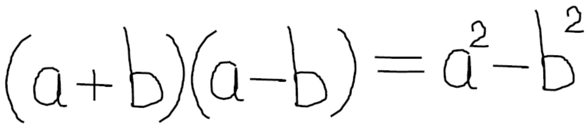

Fig. 14 User input – Mathematical expression- 3

Latex : (a+b)(a-b) = a^2 – b^2

Math output : $(a + b)(a - b) = a^2 - b^2$

Fig. 15 Output mathematical expression- 3

IV. CONCLUSION AND FUTURE SCOPE

Modern-day techniques of deep learning like neural networks can be used to implement this use case of recognizing mathematical expressions and then finding their LaTeX equivalents generating digital documents. With neural networks, the accuracy has increased to a large extent. We used CROHME data and Adam optimizer, which is computationally efficient and requires less memory space. We have provided the user with a web interface on which he/she can write the mathematical expression, which is then taken as input, and its LaTeX equivalent is given as output to the user.

In future, we can extend the work to develop a mobile application through which users can scan input from a camera of the mobile device. The accuracy of the project can be increased by further research on neural networks and using the most efficient way to recognize mathematical symbols.

REFERENCES

[1] Amit Schechter, Norah Borus, William Bakst., Converting Handwritten Mathematical Equations into LaTeX, CS221.

[2] M. Koschinski, H.-J. Winkler, M Lang., Segmentation and Recognition of Symbols Within Handwritten Mathematical Expressions.

[3] H. Mouchère, C. Viard-Gaudin, R. Zanibbi and U. Garain., ICFHR Competition on Recognition of On-Line Handwritten Mathematical Expressions (CROHME 2014), 14th International Conference on Frontiers in Handwriting Recognition, Heraklion, (2014).

[4] Ahmad-Montaser Awal, Harold Mouchère, Christian Viard-Gaudin.,Towards Handwritten Mathematical Expression Recognition.

[5] https://medium.com/@curiousily/tensorflow-for-hackers-part-iv-neural-network-from-scratch-1a4f504dfa8

[6] http://neuralnetworksanddeeplearning.com

[7] https://www.simplilearn.com/neural-networks-tutorial-article