Final Report – CSCI 507

Colin Short

December 11, 2023

Introduction

When doing scientific or engineering work, it is often a hassle to convert handwritten math to LaTeX. Furthermore, math is often handwritten when being actively developed but must be translated to a typeset form (often LaTeX) to be effectively presented. This process is inefficient, boring, and an ineffective use of human effort. Therefore, it should be automated to save time and effort for academics, students, and professionals who work with math and must present their work. This paper details a solution to this problem that uses optical character recognition to read and interpret images of handwritten as the corresponding LaTeX. Specifically, the implementation in this paper accepts images of handwritten math as input and outputs the LaTeXrepresentation of each character in the image.

Previous Work

Optical character recognition is a large and relatively old field. The earliest systems were developed in the 1940s and the first ever system capable of reading handwritten numbers was debuted in 1965 by IBM at the 1964-1965 New York World's Fair [1]. Memon et al.[1] conducted a systematic literature review in 2020 and identified existing classification methods used in the field of optical character recognition. These methods fall into five major categories: artificial neural networks, kernel methods, statistical methods, template matching techniques, and structural pattern recognition [1]. They found that artificial neural networks are one of the best classification methods for recognition tasks, and convolutional neural networks have become a widely used and successful variant of artificial neural networks. Before the popularization of artificial neural networks, kernel methods such as support vector machines were among the most robust classification methods. Many researchers still stand by support vector machines as a strong classification method for handwritten characters [1]. Statistical classifiers can be either parametric or non-parametric. Parametric classifiers have a fixed number of parameters, are generally quick to train, and do not require a large dataset. Non-parametric classifiers are typically more flexible but their number of parameters grows with the size of the dataset. Two of the most popular statistical methods are k nearest neighbors, a non-parametric classifier, and hidden Markov models, a parametric classifier [1]. The non-machine learning categories of classification methods are template matching techniques and structural pattern recognition. Template matching techniques typically use

a template image that slides over an image and the similarity is obtained via various metrics like the sum of squared differences or cross-correlation. Regions of the image that are highly similar are then matched to the template image. Structural pattern recognition classifies objects based on a relationship between pattern primitives typically organized into strings or trees to represent the structure of an object (in this case: images)[1]. In general, there are many different approaches to optical character recognition. Additionally, the condition and quality of the inputs to optical character recognition systems may necessitate additional image processing and feature extraction. For these, situations many computer vision techniques are used to accomplish these tasks such as connected components, image filtering, image morphology, edge detection[5], etc. The scope and maturity of this field have resulted in a variety of previous approaches to this class of problems.

Solution

The solution implemented for this problem uses a Convolutional Neural Network to classify handwritten math symbols. The input images are processed to separate the image into images of the individual symbols represented in the input image. The individual images of the symbols are then classified and then the aggregate of the symbol classification is the output for the input image.
This solution consists of four primary components:

1. **Data:**
   The dataset used in this project is a Kaggle dataset[4] that was modified from version 2.0 of the CHROME dataset[3]. This dataset consists of over 100,00 images of handwritten basic Greek letters, English alphanumeric symbols, math operators, set operators, predefined math functions (e.g., sin, log, lim), and other math symbols. Each sample image is a 45x45 pixel .jpg file. They are labeled by placing images into a directory whose name is the corresponding label. For example, all images that are labeled '\pi' are placed in the 'pi' folder.

2. **Data Processing:**
   In this step, all of the data is read and categorized into a data frame where each row is a sample and each column is a pixel value. The last column of each row is a numeric label. This data frame is then loaded into a pickle file. Additionally, a .csv is produced that stores the numeric labels and their corresponding LaTeXvalues.

3. **Convolutional Neural Network:**
   Samples are classified using a convolutional neural network built using TensorFlow and Keras. Its structure is summarized in Figure 1.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 45, 45, 256)	6656
max_pooling2d (MaxPooling2D)	(None, 22, 22, 256)	0
conv2d_1 (Conv2D)	(None, 22, 22, 128)	295040
max_pooling2d_1 (MaxPooling2D)	(None, 11, 11, 128)	0
dropout (Dropout)	(None, 11, 11, 128)	0
flatten (Flatten)	(None, 15488)	0
dense (Dense)	(None, 128)	1982592
dense_1 (Dense)	(None, 82)	10578

Figure 1: Convolutional neural network summary.

The first 2D convolution layer has 256 filters and the second has 128. Also, the first uses 5x5 convolutions while the second uses 3x3 convolutions. These 2D convolution layers and the dense layer all use the rectified linear unit activation function. This model was trained over 5 epochs. The training data is divided into a 70/30 train/test split.

4. **Image Processing, Feature Extraction, and Classification:**
To be fit for classification by the neural network, the input images must be processed, the symbols must be extracted, and the symbols must be classified. This process happens as follows:

   (a) The input images are converted to grayscale.

   (b) A 3x3 Gaussian blur is applied to the images.

   (c) Thresholding is applied to the images using Otsu's method and then the binary images are inverted.

   (d) Two iterations of morphological closing with a 3x3 kernel are applied to the binary images. This fills in most of the gaps in the strokes that compose each symbol.

   (e) All connected components are identified and the components with small or large areas beyond a set threshold are filtered out.

   (f) The connected components that are sufficiently close to each other either horizontally or vertically are identified and combined. This addresses the issue of symbols that consist of more than one connected component (e.g. 'i').

(g) The connected component images are resized via cropping and padding to be 45x45 pixels with the symbol in the middle of the image.

(h) A 5x5 Gaussian blur is applied to each connected component image.

(i) Thresholding is applied with a threshold value of 127 to each connected component image using Otsu's method and then the binary image is inverted.

(j) The strokes that compose the symbol in each connected component image are thinned to a thickness of one pixel via Zhang-Suen thinning. This is done because each of the images in the training data set contains symbols with a stroke thickness of 1 pixel.

(k) Each connected component image is classified by the convolutional neural network and the classifications for each symbol are concatenated into the final classification for the input image.

Figure 2 shows the overall structure of this solution and Table 1 outlines the computer vision techniques used in this solution.

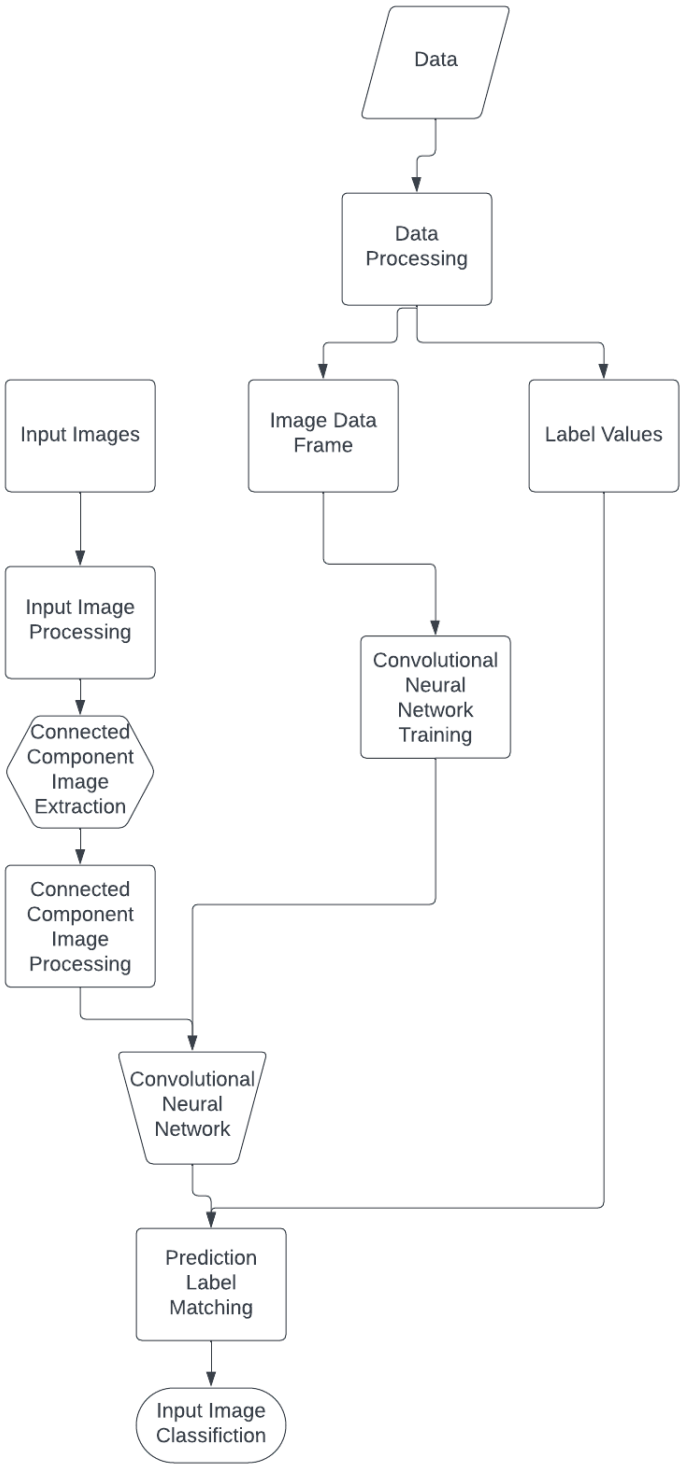

Figure 2: Block diagram of the full system.

Computer Vision Technique	Use in Project
Image Filtering	Gaussian blurring to smooth out edges of images.
Thresholding	Binarize images and reduce noise.
Morphological Closing	Fill most gaps between the strokes of symbols in the input images.
Connected Components	Extract individual symbols from the input images.
Zhang-Sueng Thinning	Extract the skeleton of symbols.
Deep Learning	Classify images of symbols.

Table 1: Computer vision techniques and their uses in this project.

Zhang-Suen Thinning

The Zhang-Suen thinning algorithm is an important part of the image processing for this system. It is an iterative, point-by-point transformation based on the value of a small set of the neighboring points of any given point in the image. A 3x3 window is used, and so each point's value is calculated based on its eight immediate neighbors[7]. A visualization of this 3x3 window is shown in Figure 3. Each iteration of the algorithm

P_9 $(i-1, j-1)$	P_2 $(i-1, j)$	P_3 $(i-1, j+1)$
P_8 $(i, j-1)$	P_1 (i, j)	P_4 $(i, j+1)$
P_7 $(i+1, j-1)$	P_6 $(i+1, j)$	P_5 $(i+1, j+1)$

Figure 3: The nine pixels designated by a 3x3 window[7].

is divided into two sub-iterations. In the first iteration, the point of interest P_1 is deleted (set to 0) if the following conditions are met:

(a) $2 \leq B(P_1) \leq 6$

(b) $A(P_1) = 1$

(c) $P_2 \times P_4 \times P_6 = 0$

(d) $P_4 \times P_6 \times P_8 = 0$

where $A(P_1)$ is the number of 0s followed by 1s or vice versa in the ordered set of the eight neighbors of P_1, and $B(P_1)$ is the number of nonzero neighbors of P_1 [7]. In the second iteration, only the third and fourth conditions are changed as follows:

(a) $P_2 \times P_4 \times P_8 = 0$

(b) $P_2 \times P_6 \times P_8 = 0$

The fourth and third conditions define how each iteration removes boundary and corner points. The first iteration only removes the bottom-right boundary points and the top-left corner points which do not belong in the skeleton of the region. Similarly, the second iteration only removes the top-left boundary points and the bottom-right corner points which do not belong in the skeleton of the region. Condition one in the first iteration preserves the endpoints of the skeleton line of the region and the second condition prevents the deletion of the points that make up the skeleton line [7]. The final result is a skeleton of the region which has a stroke thickness of one pixel.

Results

The convolutional neural network has a high performance despite being limited by the computational power available for this project. It reached 94.38% accuracy in the training step and 96.63% accuracy in the testing step. The graphs of the accuracy and the loss function over the five epochs are given by Figure 4.

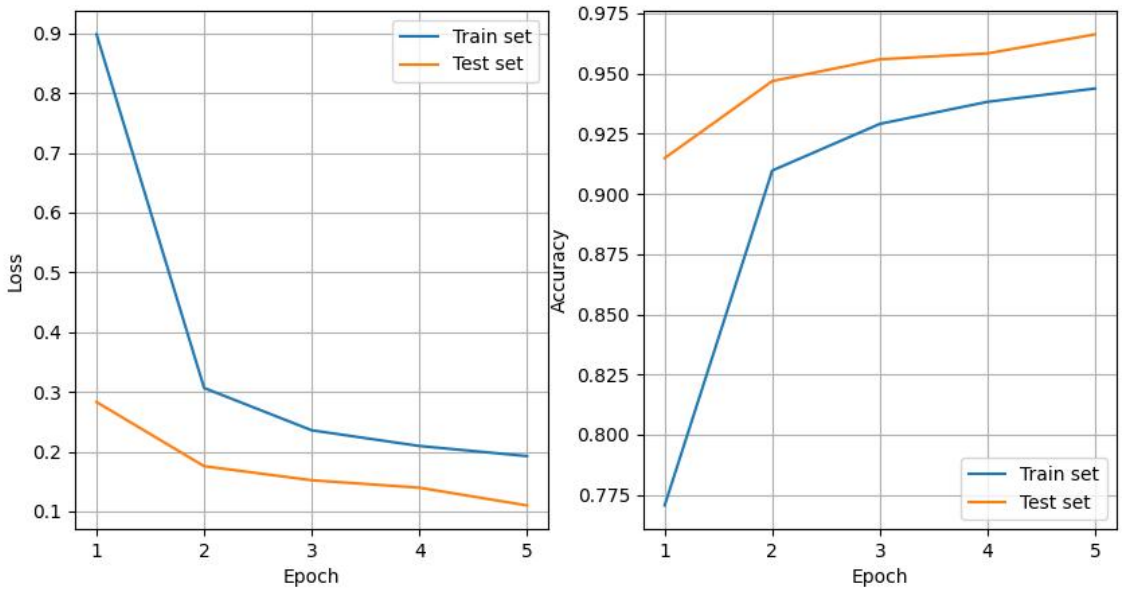

Figure 4: The accuracy and loss of the Convolutional Neural Network over 5 epochs.

The full system was evaluated via a case study using three distinct input images of handwritten equations. The following figure shows the three input images.

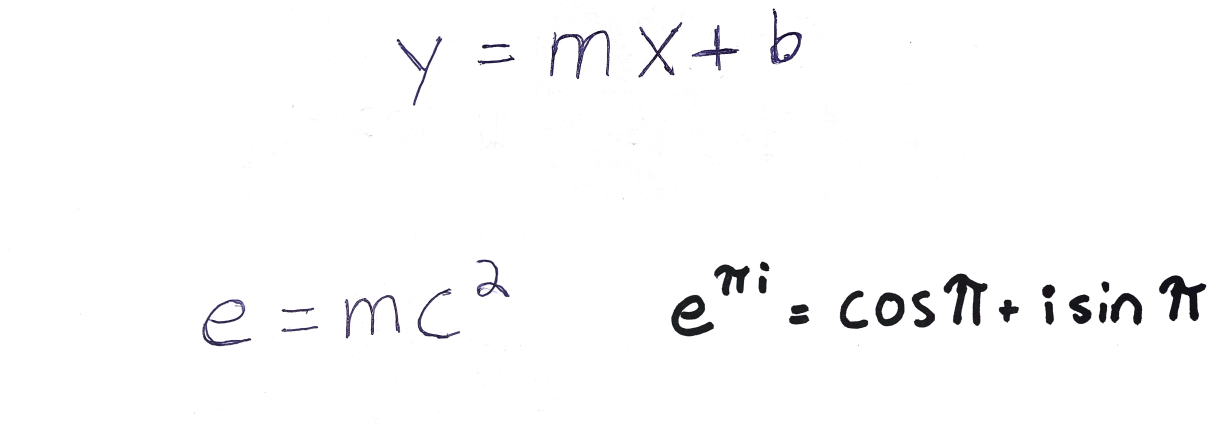

Figure 5: Case study input images.

The result of the image processing on the images is as follows.

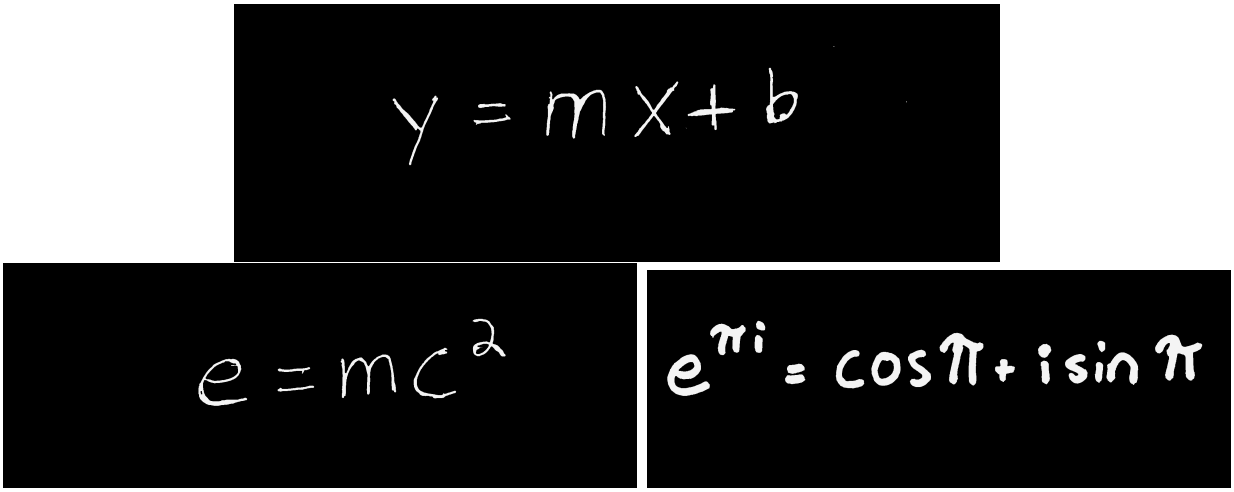

Figure 6: Processed case study input images.

Finally, the results for each image are:

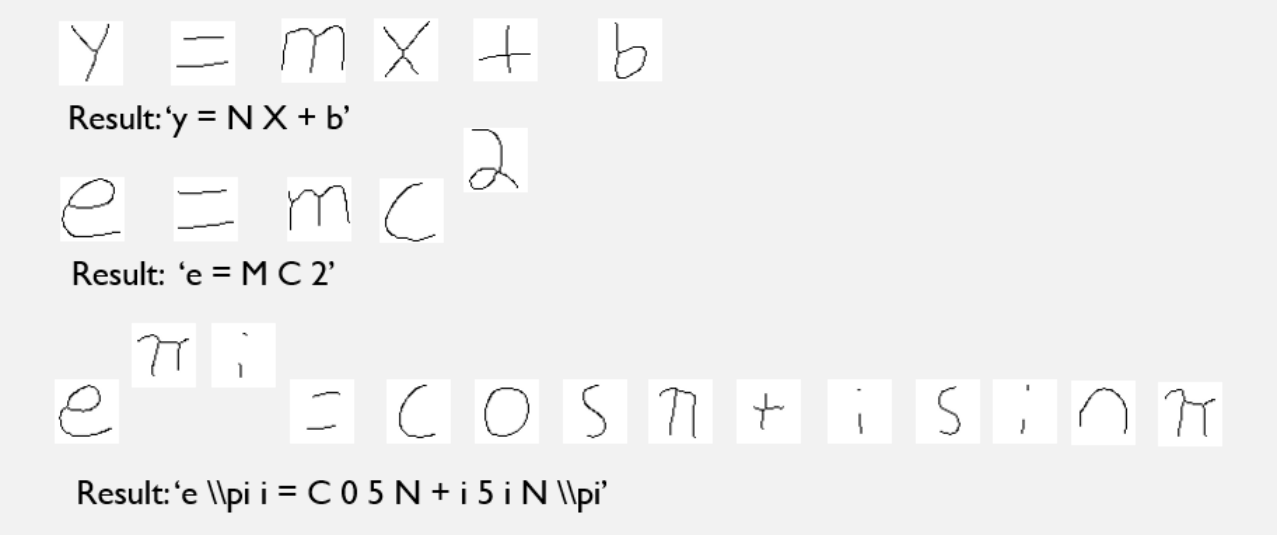

Figure 7: Final results from case study.

Conclusion

The case study shows that the system is not completely accurate in its predictions. Several limitations of this system likely contribute to this. First, only individual characters are classified by this system so functions like sin, log, cos, etc. are not recognized. Second, some characters in the training set look very similar to other characters when handwritten. Some examples shown by the case study are 's' and '5' and 'o' and '0'. To address these two limitations, additional logic or syntax awareness would be needed to extract functions from the input images to be classified and to recognize the contextual differences between similar-looking symbols. Finally, the available computing power for this project limited the size of the convolutional neural network. The structure of the neural network for this project is based on the convolutional neural network used by Mishra et al. [2]. However, the actual neural network structure in this paper would have taken 12.5 days to train while the adapted structure that was used to 5.5 hours to train. Overall, optical character recognition is a very difficult problem space with a large variety of approaches. This difficulty is compounded when recognizing handwriting because of the inconsistency of the inputs.

References

[1] Memon, J., Sami, M., Khan, R. A., and Uddin, M. Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR). *IEEE Access 8* (2020), 142642–142668.

[2] Mishra, A., Ram, A. S., and C, K. Handwritten Text Recognition Using Convolutional Neural Network.

[3] Mouchère, H. CHROME: Competition on Recognition of Online Handwritten Mathematical Expressions, Feb 2014.

[4] Nano, X. Handwritten Math Symbols Dataset, Jan 2017.

[5] Rosebrock, A. OCR: Handwriting recognition with OpenCV, Keras, and TensorFlow, Jun 2023.

[6] Yuan, Y., Liu, X., Dikubab, W., Liu, H., Ji, Z., Wu, Z., and Bai, X. Syntax-Aware Network for Handwritten Mathematical Expression Recognition, 2022.

[7] Zhang, T. Y., and Suen, C. Y. A Fast Parallel Algorithm for Thinning Digital Patterns. *Communications of the ACM 27*, 3 (1984), 236–239.