# **ABSTRACT**

We have applied different image processing techniques for recognizing handwritten mathematical equations. Converting handwritten equations from an image is a challenging machine learning problem. The crucial step in the recognition of mathematical equations is the operand and operator recognition. In this paper, we discuss how we implemented these various image processing techniques in order to recognize handwritten characters.

Our best results are obtained using multi-class SVM as a classifier, the system was able to achieve a recognition of 76.29% per character for equations with spaces and recognition of 68.64% per character for equations without spaces as the highest accuracy. We have trained our classifier using images from MNIST dataset, which contains 60,000 characters and datasets created by the researchers. We have limited our research to 22 characters only including the digits 0 to 9 and symbols (a, b, c, x, y, z, +, -, /, =, (, )).

.

# **ACKNOWLEDGEMENT**

Foremost, the team would like to extend their sincere gratitude for giving valuable and precious contributions towards the completion of the research.

To the team’s content adviser, Prof. Marianne S. Yao, for the continuous support for this research, for her enthusiasm, motivation, immense knowledge and also prayers. Her valuable feedback, encouragements and for checking our computations were of immense help throughout the duration of the research.

To the team’s panelists, Prof. Albert Geroncio Y. Rivera, Prof. Chuchi S. Montenegro, and Prof. Maria Lourdes S. Curativo, for their perceptive criticisms and insightful comments that kept us working on this research in much better way.

To the capstone project coordinator, Prof. Melody Angelique C. Rivera, whose reminders and constant motivation encouraged us to meet the deadlines and for being patient and always there for us to answer our inquiries and for assisting the team in order to meet the terms with the project requirements.

To the college dean, Prof. Dave E. Marcial, whose encouragements made it possible to achieve the goal of finishing the research.

To all the friends and colleagues who helped out in filling in and writing our data sets, without which this research would be incomplete.

And most importantly, to the team’s family, who not only assisted us financially but also extended all their support morally and emotionally.

# **LIST OF FIGURES**

[Figure 1: Block Diagram 17](file:///C:\Users\HEECHUL\Documents\CHORBS\Capstone_Final%2006-12-2017.docx#_Toc500182191)

[Figure 2: A 6-level greyscale image (right) and its histogram (left) 19](#_Toc500182192)

[Figure 3: Sample original image on the left and the result of Otsu algorithm on the right 21](#_Toc500182193)

[Figure 4: Surrounding connected components (b) and borders (c) 22](#_Toc500182194)

[Figure 5: The connected characters (left) and its vertical histogram (right) 23](#_Toc500182195)

[Figure 6: Original connected image (left) and image after drop fall (right) 24](#_Toc500182196)

[Figure 7: An example where the elements are positioned based on their X and Y features.   
SVM will then classify the elements whether they belong to class red or class blue. 25](#_Toc500182197)

[Figure 8: Elements are being classified what class they belong to using SVM 25](#_Toc500182198)

[Figure 9: Examples of hyperplanes wherein the value of w is modified [19] 26](#_Toc500182199)

[Figure 10: Examples of hyperplanes wherein the value of b is modified [19] 27](#_Toc500182200)

[Figure 11: An example where there are more than two classes to be classified by the SVM 28](#_Toc500182201)

[Figure 12: Focus to only one class first to be determined and generalize the rest so   
elements can be easily separated 28](#_Toc500182202)

[Figure 13: Class red is classified 29](#_Toc500182203)

[Figure 14: CSV format of the data from the 28x28 pixel image 36](#_Toc500182204)

[Figure 15: LIBSVM format of the data from the CSV file 36](#_Toc500182205)

[Figure 16 Overshadowed equation (left) and the result in thresholding (right) 68](#_Toc500182206)

# **LIST OF EQUATIONS**

[Equation 1: Probability that a pixel belongs to a certain class 19](#_Toc508147953)

[Equation 2: Means for the two classes 20](#_Toc508147954)

[Equation 3: Class Variances for the two classes 20](#_Toc508147955)

[Equation 4: Between-class variance 20](#_Toc508147956)

[Equation 5: Equation for maximizing between-class variance 20](#_Toc508147957)

[Equation 6: A hyperplane is represented with this equation 25](#_Toc508147958)

**LIST OF CHARTS**

[Chart 1: This chart shows the percentage accuracies of each model for Experiment 1 59](#_Toc500116507)

[Chart 2: This chart shows the percentage accuracies of each model for Experiment 2 59](#_Toc500116508)

[Chart 3: This chart shows the percentage accuracies of each model for Experiment 3 60](#_Toc500116509)

[Chart 4: This chart shows the percentage accuracies of each model   
for equations with and without spaces 60](#_Toc500116510)

[Chart 5: This chart shows the overall accuracies of each model 61](#_Toc500116511)

[Chart 6: This chart shows the operand and operator determining rate by the researcher’s   
solution using the model’s recognition for Experiment 1 61](#_Toc500116512)

[Chart 7: This chart shows the operand and operator determining rate by the researcher’s   
solution using the model’s recognition for Experiment 2 62](#_Toc500116513)

[Chart 8: This chart shows the operand and operator determining rate by the researcher’s   
solution using the model’s recognition for Experiment 3 63](#_Toc500116514)

[Chart 9: This chart shows the summary of determining rate by the researcher’s solution   
using the models’ recognition 63](#_Toc500116515)

[Chart 10: This chart shows the overall determining rate by the researcher’s solution using   
the models’ recognition 64](#_Toc500116516)

**LIST OF TABLES**

[Table 1: Model Recognition and Operand and Operator Determining   
Rate of Experiment 1 using Model 1 41](#_Toc500115612)

[Table 2: Model Recognition and Operand and Operator Determining   
Rate of Experiment 2 using Model 1 42](#_Toc500115613)

[Table 3: Model Recognition and Operand and Operator Determining   
Rate of Experiment 3 using Model 1 43](#_Toc500115614)

[Table 4: Model Recognition and Operand and Operator Determining   
Rate of Experiment 1 using Model 2 44](#_Toc500115615)

[Table 5: Model Recognition and Operand and Operator Determining   
Rate of Experiment 2 using Model 2 45](#_Toc500115616)

[Table 6: Model Recognition and Operand and Operator Determining   
Rate of Experiment 3 using Model 2 46](#_Toc500115617)

[Table 7: Model Recognition and Operand and Operator Determining   
Rate of Experiment 1 using Model 3 47](#_Toc500115618)

[Table 8: Model Recognition and Operand and Operator Determining   
Rate of Experiment 2 using Model 3 48](#_Toc500115619)

[Table 9: Model Recognition and Operand and Operator Determining   
Rate of Experiment 3 using Model 3 49](#_Toc500115620)

[Table 10: Model Recognition and Operand and Operator Determining   
Rate of Experiment 1 using Model 4 50](#_Toc500115621)

[Table 11: Model Recognition and Operand and Operator Determining   
Rate of Experiment 2 using Model 4 51](#_Toc500115622)

[Table 12: Model Recognition and Operand and Operator Determining   
Rate of Experiment 3 using Model 4 52](#_Toc500115623)

[Table 13: Model Recognition and Operand and Operator Determining   
Rate of Experiment 1 using Model 5 53](#_Toc500115624)

[Table 14: Model Recognition and Operand and Operator Determining   
Rate of Experiment 2 using Model 5 54](#_Toc500115625)

[Table 15: Model Recognition and Operand and Operator Determining   
Rate of Experiment 3 using Model 5 55](#_Toc500115626)

[Table 16: Table of accuracy for each models by experiments for   
equations with spaces and without spaces 56](#_Toc500115627)

[Table 17: Table of accuracies for each models by accuracies of   
equations with spaces and without spaces 57](#_Toc500115628)

[Table 18: Table of accuracies for determining of operands and   
operators by experiments for equations with and without spaces 58](#_Toc500115629)

[Table 19: Table of accuracies for determining of operands and   
operators by accuracies of equations with spaces and without spaces 59](#_Toc500115630)

**TABLE OF CONTENTS**

[**ABSTRACT** 1](#_Toc508146207)

[**ACKNOWLEDGEMENT** 2](#_Toc508146208)

[**LIST OF FIGURES** 3](#_Toc508146209)

[**LIST OF EQUATIONS** 4](#_Toc508146210)

[**LIST OF CHARTS** 5](#_Toc508146211)

[**LIST OF TABLES** 6](#_Toc508146212)

[**TABLE OF CONTENTS** 8](#_Toc508146213)

[**Chapter 1 : INTRODUCTION AND BACKGROUND OF THE STUDY** 10](#_Toc508146214)

[**1.1** **Introduction** 10](#_Toc508146215)

[**1.2** **Conceptual Background** 11](#_Toc508146216)

[**1.3** **Literature Review** 11](#_Toc508146217)

[**1.3.1** **Binarization** 12](#_Toc508146218)

[**1.3.2** **Image Segmentation** 12](#_Toc508146219)

[**1.3.3** **Feature Extraction** 13](#_Toc508146220)

[**1.3.4** **Classification and Recognition** 13](#_Toc508146221)

[**1.4** **Problem Description and Statement** 13](#_Toc508146222)

[**1.5** **Objectives** 14](#_Toc508146223)

[**1.6** **Scope and Limitation** 14](#_Toc508146224)

[**Scope** 14](#_Toc508146225)

[**Limitation** 14](#_Toc508146226)

[**1.7** **Significance of the Study** 15](#_Toc508146227)

[**1.8** **Definition of Terms** 15](#_Toc508146228)

[**Chapter 2 : THEORY AND METHODS** 16](#_Toc508146229)

[**2.1 Conceptual Framework / Technical Background** 16](#_Toc508146230)

[**2.1.1 Grayscale** 17](#_Toc508146231)

[**2.1.2 Otsu Method** 17](#_Toc508146232)

[**2.1.3 Topological Analysis by Border Following** 20](#_Toc508146233)

[**2.1.4 Peak Finding** 21](#_Toc508146234)

[**2.1.5 Drop Fall Segmentation** 22](#_Toc508146235)

[**2.1.6 Support Vector Machine** 23](#_Toc508146236)

[**2.1.6 Neuroph** 28](#_Toc508146237)

[**2.1.7 MNIST Database** 29](#_Toc508146238)

[**2.2 PROJECT TIMELINE** 30](#_Toc508146239)

[**Chapter 3 : ALGORITHM IMPLEMENTATION** 32](#_Toc508146240)

[**3.1 Image Preprocessing** 32](#_Toc508146241)

[**3.1.1 Grayscale** 32](#_Toc508146242)

[**3.1.2 Binarize** 32](#_Toc508146243)

[**3.2 Segmentation** 32](#_Toc508146244)

[**3.2.1 Topological Analysis by Border Following Algorithm** 32](#_Toc508146245)

[**3.2.2 Peak Finding** 33](#_Toc508146246)

[**3.2.3 Drop Fall** 34](#_Toc508146247)

[**3.3 Image Information Acquisition** 34](#_Toc508146248)

[**3.4 Training** 35](#_Toc508146249)

[**3.5 Testing** 35](#_Toc508146250)

[**3.6 Determining Operands and Operators** 36](#_Toc508146251)

[**CHAPTER 4: TESTING, RESULTS, AND OBSERVATIONS** 39](#_Toc508146252)

[**4.1 TESTING** 39](#_Toc508146253)

[**4.2 RESULTS** 42](#_Toc508146254)

[**4.3 Accuracy Calculations** 58](#_Toc508146255)

[**4.3 OBSERVATIONS** 67](#_Toc508146256)

[**CHAPTER 5: CONCLUSION AND RECOMMENDATION** 67](#_Toc508146257)

[**5.1 CONCLUSION** 67](#_Toc508146258)

[**5.2 RECOMMENDATION** 68](#_Toc508146259)

[**BIBLIOGRAPHY** 69](#_Toc508146260)

[**APPENDIX A** 71](#_Toc508146261)

[**APPENDIX B** 71](#_Toc508146262)

**OVO : APPLYING IMAGE PROCESSING TECHNIQUES ON SIMPLE HANDWRITTEN MATHEMATICAL EQUATION**

# **: INTRODUCTION AND BACKGROUND OF THE STUDY**

## **Introduction**

Handwritten character recognition is becoming more important in the modern world since it helps human not only make their jobs easier, but also it solves more complex problems. However, recognizing lines of handwritten text is a difficult task.

In order to bridge the gap between technology and the traditional pen and paper approach, image processing was used. Image processing is a method that is used to convert an image into digital form and perform some operations on it in order to get an enhanced image or to extract some useful information from it [1]. Using image processing, one can extract and get the useful data that will be needed.

Human beings obtain the knowledge to identify letters, numbers, math symbols, or which are the operators or the operands in a handwritten mathematical equation even with or without the proper spacing. However, making a machine do these types of task especially with the lack of proper spacing can be a very complicated problem considering spaces is a part of the basis to identify which is the operator or the operand.

So, can a machine determine the operators and the operands of a handwritten mathematical expression? Can it determine which is which even with lack of proper spacing? These questions have led the researchers to come up with the idea of determining which are the operator and the operand in a handwritten mathematical expression and applying image processing techniques.

## **Conceptual Background**

For the past years, many researchers are already having interest on the field of Image Processing. One study uses image processing techniques where it recognizes machine printed linear equations. [2]

As technology advances, researchers delved deeper and widen the coverage of Image Processing and now having studies on Handwriting Recognition. This includes recognition of written numbers, letters, symbols, etc.

In Handwriting Recognition, it can be “offline” or “online”. It will be considered offline if the handwriting is written on paper and is often scanned. This means the individual characters contained in the scanned image will need to be extracted. On the other hand, online is when the handwriting is written using a special digitizer or PDA, where a sensor picks up the pen-tip movements as well as pen-up/pen-down switching.

Our study focuses on offline recognition of handwritten equation. It converts the handwritten equation from an image into digitized form and be able to determine the equation’s operators and operands.

## **Literature Review**

Image processing and image segmentation plays an important role in handwritten character, digit, and symbol recognition. A study [3], applied the steps in image processing particularly the pre-processing stage which involves all of the operations to achieve a clean character image, so that it can be used for the next stage. Before segmentation, a sequence of simple, common preprocessing is applied to make the image feasible to the recognition algorithms and to reduce complexity.

### **Binarization**

In the study [4], discussed that image binarization is an important pre-processing step of image processing and analysis because it affects the segmentation stage as well as the final character recognition. According to [5], binarization process converts gray scale or colored image to black and white image and pointed that the process may be applied to enhance visibility and structural information of the character from an image. In a study [6], they applied image binarization for preprocessing task. The study also pointed that “image binarization is useful to character analysis systems and for character recognition”. It automatically converts the document images in a bi-level form in such way that the foreground information is represented by black pixels and the background by white ones.

### **Image Segmentation**

Image segmentation refers to the partition of an image into a set of regions that cover it. The goal in many tasks is for the regions to represent meaningful areas of the image.

In the study [7], pointed that the segmentation phase is the most important process. They used segmentation to separate the individual characters from an image. One study that applies image segmentation wherein their study enhances the resolution of an image [8], discusses that “image segmentation is an essential task in the fields of image processing and computer vision”. The study uses image segmentation for partitioning the digital images and is used to locate the boundaries into a finite number of meaningful regions and easier to analyze.

### **Feature Extraction**

Feature extraction in a type of dimensionality reduction that efficiently represents interesting parts of an image as a compact feature vector.

In a study [9], uses feature extraction for easier classifying of data from an image. They pointed that the main purpose of feature extraction is to remove the redundancy from data and select the important features from the image for efficient recognition of pattern. Moreover, they discussed that feature extraction is used based on the shape approximation.

### **Classification and Recognition**

After feature extraction a recognition stage is used to identify the corresponding character.

The study PCA based Mathematical Equation Solver: Digital Image, uses Principal Component Analysis (PCA) based handwritten mathematical digits and symbols recognition algorithm. “Recognition of mathematical symbols is one of the most difficult recognition tasks given its complexity, type and forms of equations, large variation in the way people write equations. In increasing the number of basis and increasing the size of the dataset, one might achieve close to perfect recognition of a mathematical equation using PCA.” [10]

A study that applied SVM (Support Vector Machine) [11], uses SVM for classifying and recognizing characters from the image.

## **Problem Description and Statement**

There are existing researches today that can recognize handwritten and printed mathematical equations. However, these researches only considered equations that have the right spacing between operands and operators. So with this, we came up with the idea of distinguishing the operand and operators from the handwritten equation even without spaces.

## **Objectives**

The objective of the study is to determine the operands and operators from an image of a handwritten mathematical equation even without spaces.

## **Scope and Limitation**

The scope and limitations of this project include the following:

### **Scope**

• The math equation will be inputted by taking a picture using a camera from the paper containing the equation.

• The equations with or without spaces will be processed as long as it is valid.

### **Limitation**

• The paper used for writing the equation must be blank. Lined or ruled papers are not acceptable. The paper can have any color as long as the image of the equation taken is clear.

• The equation must be valid and must be written in one line.

• An operand should be a counting number (e.g. 1, 50, 50000), a variable using a single letter (e.g. x, y, z, a, b, c), or a combination of both number and variable (e.g. 20a, 14x).

• The letters that can be recognized are a, b, c, x, y, z since these are the most common letters that are used as variables in an equation.

• The operator symbol should be ‘+’ for addition, ‘-’ for subtraction, ‘x’ for multiplication, ‘/’ for division and “=” for equal sign.

## **Significance of the Study**

The study is significant in a way that it would serve as a reference for further studies in the field of image processing.

## **Definition of Terms**

**Feature Extraction** – a type of dimensionality reduction that efficiently represents interesting parts of an image as a compact feature vector.

**Image Classification** – to categorize all pixels in a digital image into one several land cover classes.

**Support Vector Machines** – are group of supervised learning methods that can be applied to classification or regression.

**Hole Border** – The inner border of a binary image. (Refer to Fig. 4)

# **: THEORY AND METHODS**

## **2.1 Conceptual Framework / Technical Background**

Figure 1: Block Diagram

Input image

TRAINING

Creating of data sets for operators and letters

IMAGE PREPROCESSING

Grayscale

Merging the created data set (operators and letters) and the data set from MNIST

Binarize

Train the merged data set using Multiclass SVM

Segmentation

Cleaning

Output of trained SVM

RECOGNITION

Recognition of handwritten digits, operators, and letters using the output of trained SVM

Determination of operands and operators

Outputs the list of operand and operators

Store into a linked list with its digital form

Fig. 1 shows 3 major processes: image pre-processing, data training, and image recognition. The image that has been entered will undergo image pre-processing and the result from the training part is going to be used for the recognition of the pre-processed image.

### **2.1.1 Grayscale**

To greyscale an image is an essential step as a precursor to image binarization. Greyscale is a range of shades of gray without apparent color. A greyscale image is a collection of pixels which each hold a value between 0 (white) and 255 (black) [12]. Gray RGB color code has equal red, green and blue values: R = G = B.

### **2.1.2 Otsu Method**

Otsu method is a type of global thresholding in which it depend only in the grey value of an image or a greyscale image that can be used for image pre-processing [12]. The researchers will be using the Otsu Method to separate the light and dark regions of the greyscale image and to extract an object from its background by assigning intensity value for each pixel such that each pixel is either classified as an object point or a background point.

In order to achieve this, we need to set a threshold value. Otsu’s method finds the threshold value that we are looking for. To find the threshold value, Otsu’s method iterates through all the possible threshold values and calculates a measure of spread for the pixel levels each side of the threshold, i.e. the pixel that either falls in the foreground or background. Otsu method is using the image histogram data as input and finds a pixel value (so called threshold level) that is able to separate the image into foreground and background [13].

For example, the algorithm will be demonstrated using the simple 6x6 image shown below. The histogram for Fig.2 is shown next to it.

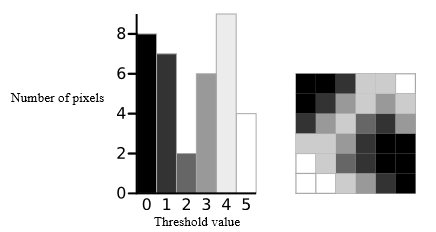
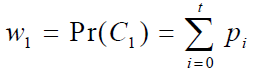
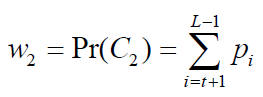


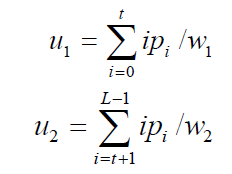
Figure 2: A 6-level greyscale image (right) and its histogram (left)

Assuming an image is represented in L gray levels [0,1,…L-1]. The number of pixels at level ***i*** is denoted by *ni,,* and the total number of pixels is denoted by *N* = *n1 + n2 + … + nL.* Now the goal is to separate this distribution into two distinct classes *C1*with gray levels [0,1,…*t*] and *C2*with gray levels [*t*+1,…L-1] by the threshold *t*. The gray level probability distributions for the two classes are

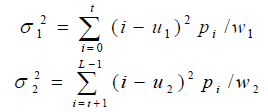
Equation 1: Probability that a pixel belongs to a certain class

Also, the means for the two classes are



Equation 2: Means for the two classes

The class variances are



Equation 3: Class Variances for the two classes

Otsu was focusing on a method to determine the “goodness” of the separation. Otsu defined the between-class variance of the thresholded image as

C:\Users\HEECHUL\Desktop\bet-class.png

Equation 4: Between-class variance

Otsu method chooses the optimal threshold *t* by maximizing the between-class variance, which is equivalent to minimizing the within-class variance, since the total variance (the sum of within-class variance and the between-class variance) is constant for different partitions.

C:\Users\HEECHUL\Desktop\max.png

Equation 5: Equation for maximizing between-class variance

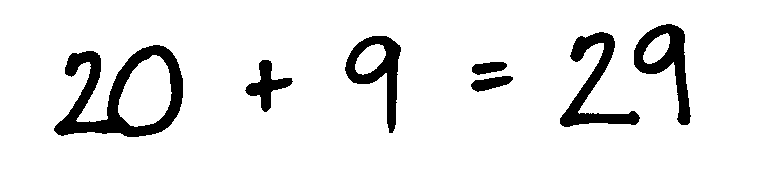
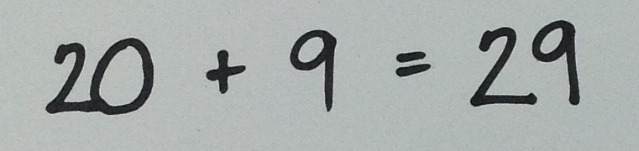


Figure 3: Sample original image on the left and the result of Otsu algorithm on the right

### **2.1.3 Topological Analysis by Border Following**

Topological Analysis by Border Following is an algorithm that determines the borders of a binary image. This algorithm is used to extract the topological structure of a given binary image. The information to be extracted is the relation among the two types of borders in a binary image which are the outer borders and hole borders. This can be done by first, putting a unique mark on each border rather than to adopt same marking procedure for every border (border labeling) and second, to add a procedure for obtaining the parent border of the current followed border [14].

In the algorithm, the black pixels are denoted as 1–pixels while the white pixels are denoted as the 0–pixels. Moreover, it is noted that the hole border as well as the outer border is defined as a set of 1 – pixels (not 0 – pixels). Furthermore, the algorithm adopt the coordinate system where the row number (denoted as *i*) increases from top to bottom and the column number (denoted as *j*) increases from left to right. A picture having the density value at a pixel ( *i , j* ) is denoted by F = {}. There are several steps in the algorithm that are performed in each pixel to identify the border such that.

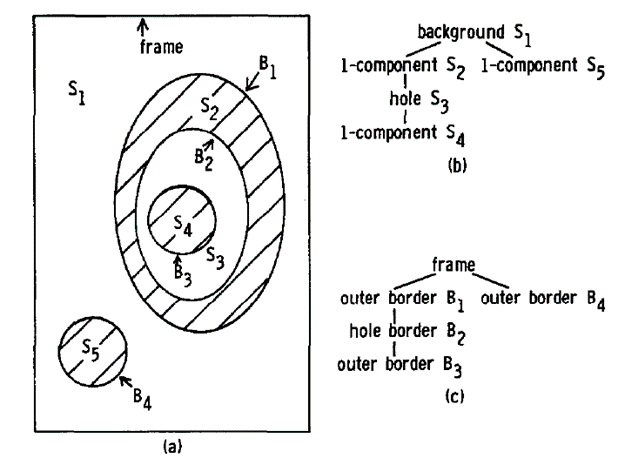


Figure 4: Surrounding connected components (b) and borders (c)

Fig.4 shows the relation among connected components and the borders. (a) S1 component is the background component that contains black pixels as well as S3 which is a hole. The S2, S4 and S5 are components that contain white pixels. The B1, B3 and B4 are the outer borders and B2 is the hole border [14].

If a binary image that is stored in a form of the borders and the relation among borders are extracted by this algorithm, some simple image processing can be done without restoring the original image. Thus, the algorithm offers an effective way of storing binary image data that can be used for image segmentation [14].

### **2.1.4 Peak Finding**

Peak finding is the process of finding one of the elements where it peaks [22]. This method will be used for determining how many digits or characters are connected. To perform peak finding, the maximum black pixels should be determined. In order to recognize the maximum black pixels, the black pixels in every column of the image were counted in order to create a vertical histogram. Once the vertical histogram was created, it will be used to look for all the maximum black pixels in each column. A black pixel x(n) is considered as a peak when it is greater than both the elements on its sides (x(n-1) < x(n) > x(n+1). For example: (Refer to Fig. 5)

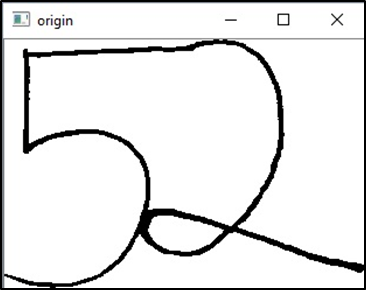
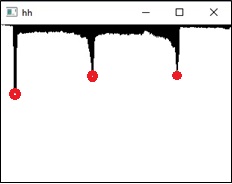
 

Figure 5: The connected characters (left) and its vertical histogram (right)

Based on the number of peaks in Fig. 5, you can determine that there are two connected characters. If the number of peaks is greater than or equal to 2, it is then assumed that the characters are connected. Otherwise, the image has only one character.

### **2.1.5 Drop Fall Segmentation**

Drop fall is a segmenting algorithm that is often used in character segmentation most especially in segmenting handwritten characters that are connected [23]. This algorithm mimics the motions of a falling raindrop that falls from above the characters, rolls along the contour of the characters and cuts through the contour when it cannot fall further. The raindrop follows a set of movement rules to determine the segmentation trace.

The starting point can be any position between the connected numbers and above the connection (which is the area to be cut). It assumes that an appropriate starting drop will have white pixels on either side of it. Also, the starting drop should be close to the top of the image. It will scan the image row by row, starting at the top and works its way down until it finds a white pixel and black pixel nearby. If all pixels around the current point are occupied with white pixels, it will move downward therefore cutting the contour. For example:

Figure 6: Original connected image (left) and image after drop fall (right)

After choosing an appropriate start point, the algorithm starts. The algorithm follows a set of rules that elevates one pixel at a time until the bottom of the image is reached.

### **2.1.6 Support Vector Machine**

Machine learning is about learning structure from data. It explores the study and construction of algorithms that can learn from and make predictions on data [15]. In machine learning, there are three broad categories namely supervised learning, unsupervised learning, and reinforcement learning. In our case, supervised learning will be used. It uses labeled training data to infer a function and these training data consists a set of training examples. Each example is a pair consisting of an input object (typically a vector) and a desired output value [16].

Support vector machine is a supervised learning method that can be applied to classification [17]. It analyzes the training data and produces an inferred function which can be used for mapping new examples.

SVM acts as a classifier where its purpose is to decide what class the new data belongs. Here is an example on how SVM works. Refer to Figure 7.

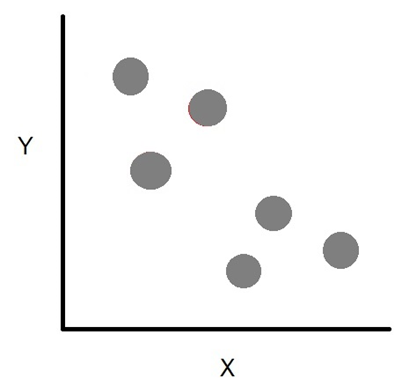


Figure 7: An example where the elements are positioned based on their X and Y features. SVM will then classify the elements whether they belong to class red or class blue.

With the SVM algorithm being applied, it was able to know which elements belong to class red or class blue by using the hyperplane, which separates the classes.

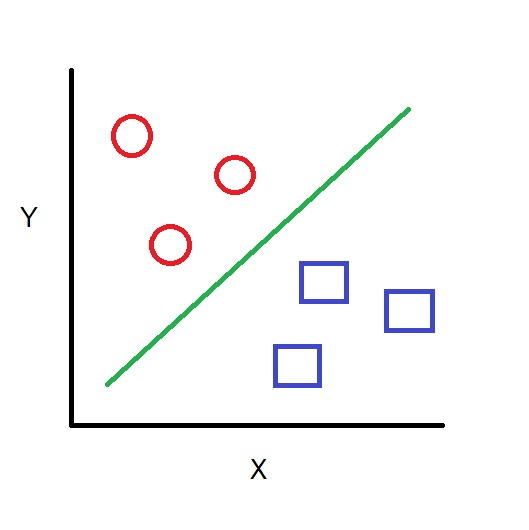


Figure 8: Elements are being classified what class they belong to using SVM

Equation 6: A hyperplane is represented with this equation

A hyperplane is the borderline of the classes which separates them and its position can be changed using Equation 6. In that equation, **w** is for the weight vector, **x** for the input vector and **b** is for the bias. With the value of w being modified, the orientation of the hyperplane is changed (rotated). With the value of b being modified, the displacement of the hyperplane is changed (shifted).

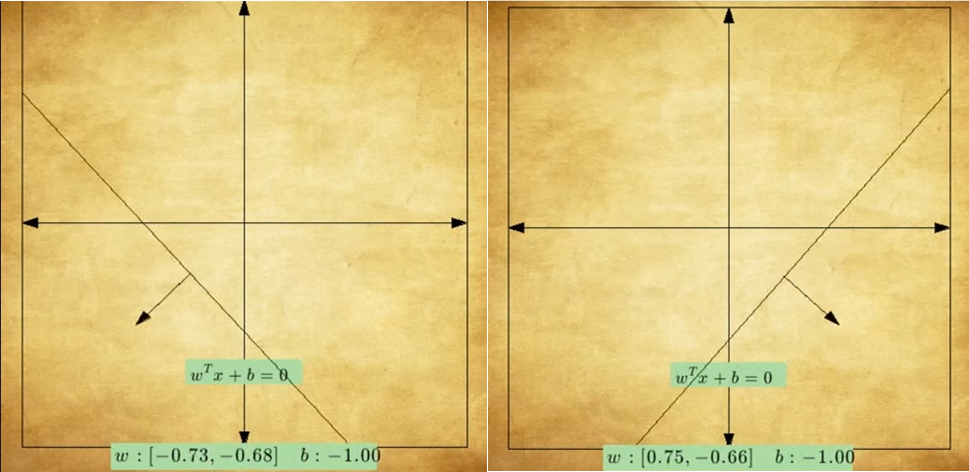


Figure 9: Examples of hyperplanes wherein the value of w is modified [19]

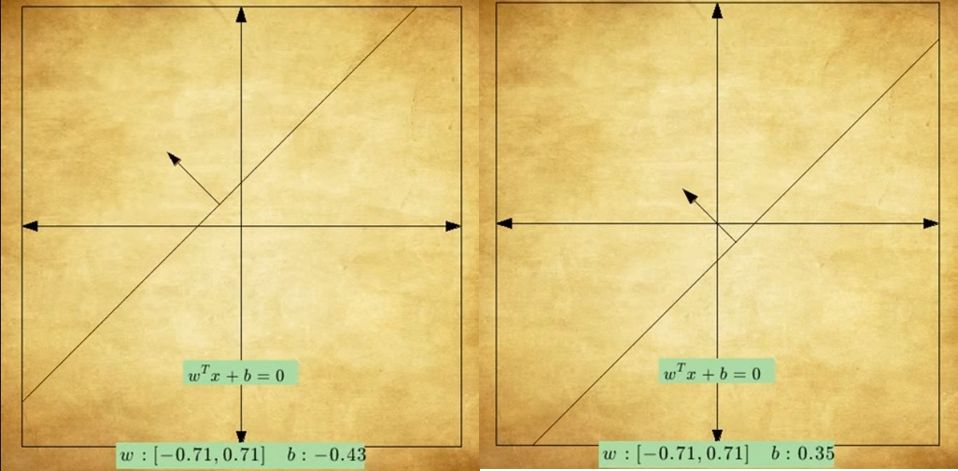


Figure 10: Examples of hyperplanes wherein the value of b is modified [19]

Once the hyperplane is positioned in the right place, the data can then be determined if it belongs to class red or class blue (Figure 8). We can then assume that class red is class -1 and class blue is +1. Therefore, class red is w ∙ x + b < 0 and class blue is w ∙ x + b > 0.

After the machine has been trained, it is time to classify a new element x. If x is greater than 0, then it belongs to class blue. Otherwise, class red.

Based on the explanation given, only two classes are being classified. In some cases, there are more than two classes to be classified (Figure 11). This situation applies in the project’s case. We will be using the one vs. all approach of the SVM or also called as the Multiclass SVM. What it does is it focuses only on one class and generalizes the rest (Figure 12). So any element that does not have the feature of red, is considered not red or it belongs to class -1.

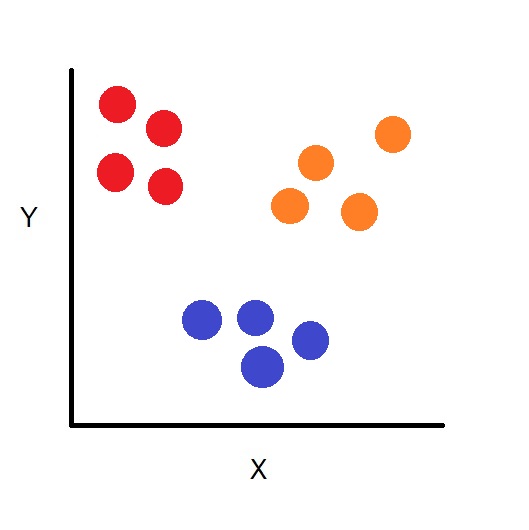


Figure 11: An example where there are more than two classes to be classified by the SVM

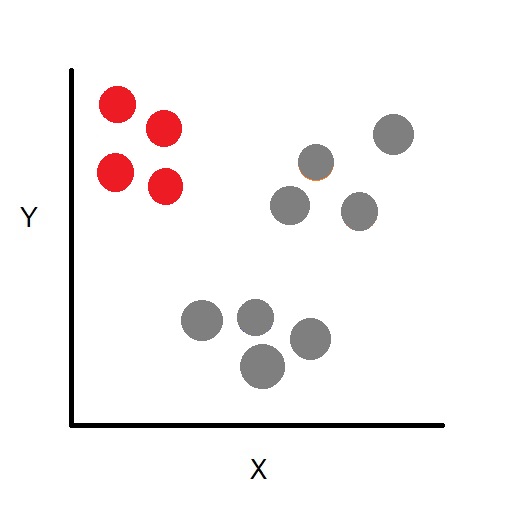


Figure 12: Focus to only one class first to be determined and generalize the rest so elements can be easily separated

The same concept of the hyperplane is being done. As the hyperplane is w ∙ x + b = 0, those elements greater than 0 belongs to one class and those elements lesser than 0 belongs to another class (Figure 13). In the said example, since the blue and orange class does not have the features of red, then they are being generalized.

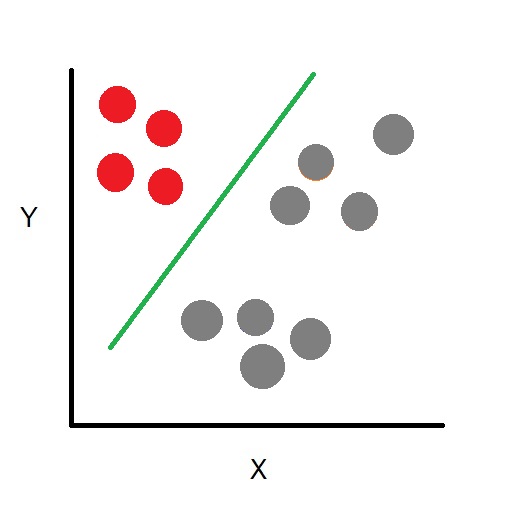


Figure 13: Class red is classified

As we have classified the red class, we call this as classifier 1. The same process is being done with class blue (classifier 2) and class orange (classifier 3). Now that the machine has been trained, it is time to classify a new element x. What the machine does is it will make a decision. The three classifiers which are trained to classify a specific class are run to determine if a new element x belong to the class. Whichever classifier that is the most confident one, we can then say that element x belongs to that class.

### **2.1.6 Neuroph**

Neuroph is lightweight Java neural network framework to develop common neural network architectures. It contains well designed, open source Java library with small number of basic classes which correspond to basic Neural Network concepts. Also has GUI (Graphical User Interface) neural network editor to quickly create Java neural network components. Neuroph simplifies the development of neural networks by providing Java neural network library and GUI tool that supports creating, training and saving neural networks [20].

This is used for creating the data sets for letters (a, b, c, x, y, z), operators (+, - , x, /) and symbols (=, ( )). The researchers will collect samples each to be used as training set.

### **2.1.7 MNIST Database**

MNIST Database is a dataset of handwritten digits, that has training set of 60,000 examples, and a test set of 10,000 examples. The database is a subset of a larger set of available from NIST. The digits in the database have been size-normalized (28x28 pixels) and centered in a fixed-size image [21].

The data is stored in a very simple file format designed for storing vectors and multidimensional matrices. All the integers in the files are stored in the MSB first (high endian) format used by most non-Intel processors.

There are two files that will be used in the project. The first file contains training set images that consist of 60,000 handwritten digit images from 0 to 9. The second file contains the training set labels that consist of 60,000 label values from 0 to 9 indicating which digit the image represents. In addition to this, the datasets (a, b, c, x, y, z, ( ), =, +, -, x, /) that is created by the researchers will be added to the MNIST dataset. These files will be used to train the SVM classifier.

## **2.2 PROJECT TIMELINE**

First Increment (preparing the data for recognition)

Input data: the image taken of the handwritten equation

Input Data

Grayscale: converts the image into grayscale.

Data Pre-processing

Binarize: converts the grayscaled image into a binarized image

Grayscale

Binarize

Segmentation: the binarized image of the handwritten equation will be “chopped” thus giving us an individual images of the numbers, letters, and operators from the equation

Data Processing

Segmentation

Cleaning: removes the noise that is included in the individual images

Cleaning

**First increment input:** an image of a handwritten equation

**First increment output:** individual images of the numbers, letters, and operators which is already binarized and noise-free

Second Increment (training the data to recognize handwritten digits, operators, and letters)

Training

Since there is no existing data set for handwritten operators and letters, it has to be created

Creating of data sets for operators and letters

After the data set for operators and letters has been created, it is merged with the existing data set for handwritten digits from MNIST

Train the merged data set using Multiclass SVM

Merging the created data set (operators and letters) and the data set from MNIST

With the merged data sets consisting of digits, needed letters, and operators, it’s time to train the data

**Second increment input:**

* handwritten operators and needed letters for creating of data sets for it
* data set from MNIST

**Second increment output:** trained data for recognizing handwritten digits, operators and letters

Third Increment (Determining operand and operators)

The output from the 1st increment (the data to be recognized) will be recognized using the output from the 2nd increment

Recognition of handwritten digits, operators, and letters

Once the data has been recognized, its digital form can be determined and it will be stored in a stack for preparation of determining the operands and operators

Store into a linked list with its digital form

Using stack, operands and operators will be known

Determination of operands and operators

After the machine has learned, it is now time for testing.

Testing

**Third increment input:** output from 1st and 2nd increment

**Third increment output:** determined operands and operators

# **: ALGORITHM IMPLEMENTATION**

## **3.1 Image Preprocessing**

### **3.1.1 Grayscale**

After attaining the image to be used as test data in the study, it will be then processed into greyscale. Greyscaling is the task of converting the image into a different shades of gray with the darkest possible shade is black and the lightest possible shade is white. An image must be greyscaled in order to achieve better results in binarization. The result image of this will then be binarized.

### **3.1.2 Binarize**

Binarization is the process of converting the pixel image into a binary image. So each pixel values must either be black or white only. Otsu method, an algorithm that converts a greyscale image into a binarized one, is applied in order to separate the image into background or foreground. In result to this, the stroke of the handwriting is in contrast to its background. After the image of an equation has been binarized, it is now ready to be segmented.

## **3.2 Segmentation**

Before the segmentation process starts, the stroke of the handwriting should be white and its background should be black as it is a requirement for the next process.

### **3.2.1 Topological Analysis by Border Following Algorithm**

OpenCv has a function called findContours which is a useful tool for shape analysis, object detection and recognition. The function only extracts the outer boundary of each contour. Finding contours is basically finding the white pixels from a binary image which in this case is the stroke of the handwritten equation. The function retrieves contours from the binary image using the Topological Analysis by Border Following algorithm. The algorithm is used to extract the topological structure of a binary image. After this process, the characters of the equation will now be individually segmented.

### **3.2.2 Peak Finding**

There will be instances where there will be connected characters which are considered as one character by the preceding process. To be able to know whether there are connected characters, peak finding helps in solving this kind of situation. Peak finding is the process of finding the peaks of an element. This method will be used for determining if the characters are connected by performing the following:

1. Find Peaks using histogram (horizontal histogram and vertical histogram)
2. Counting the number of pixels in each column to create a horizontal/vertical histogram. The histogram shows the distribution of pixels in each column.
3. Using the histogram, find the peaks by determining the uphill and downhill. The number of peaks will determine if there are connected characters or none.
4. Find top peaks of the segmented contour
5. The image acquired from the contour segmentation will be determined as a connected character or not by determining the uphill and downhill of the contour. A peak can be considered as a peak if it has an uphill and downhill.
6. The algorithm checks every column of the pixels if the next column’s highest black pixel is greater than the current pixel. If so, it is then determined as an uphill. However, if the next column is less than the current pixel then it is determined as downhill.
7. Find bottom peaks of the segmented contour.
8. The acquired image from segmentation of contours will be recognized as a connected character or not by determining downhill and uphill of the contour.
9. Finding the bottom peaks is basically the opposite of finding the top peaks.

### **3.2.3 Drop Fall**

When there are connected characters found, Drop Fall Algorithm is used to segment the connected characters. The drop starts from the bottom of the image and advances one black pixel at a time until it reaches the top of the image. The algorithm follows a set of rules to perform the splitting of connected characters. The algorithm considers five adjacent pixels: the pixel to the left, pixel to the right and the three pixels above the pointer (upper right, upper left, up). The pointer moves one pixel at a time until the pointer reaches a position where it is surrounded by white pixels. Finally, if the pointer is surrounded by white pixels, the pointer will move upward and thereby cutting the contour.

Moreover, if the splitting of the connected character resulted in a way that the number of contour present in the image is less than or greater than two contours, then the drop fall will be performed from the top to the bottom of the image.

# **3.3 Image Information Acquisition**

After the segmentation process, the image will then be converted into a CSV file. Every pixel value of the 28x28pixel image is saved in one row of a CSV file. The 2nd column until the 785th has the image information but not all of it are significant so only the needed pixel values are being picked along with the column number and is being saved as a LIBSVM format which will then be later on fed into the classifier for prediction.

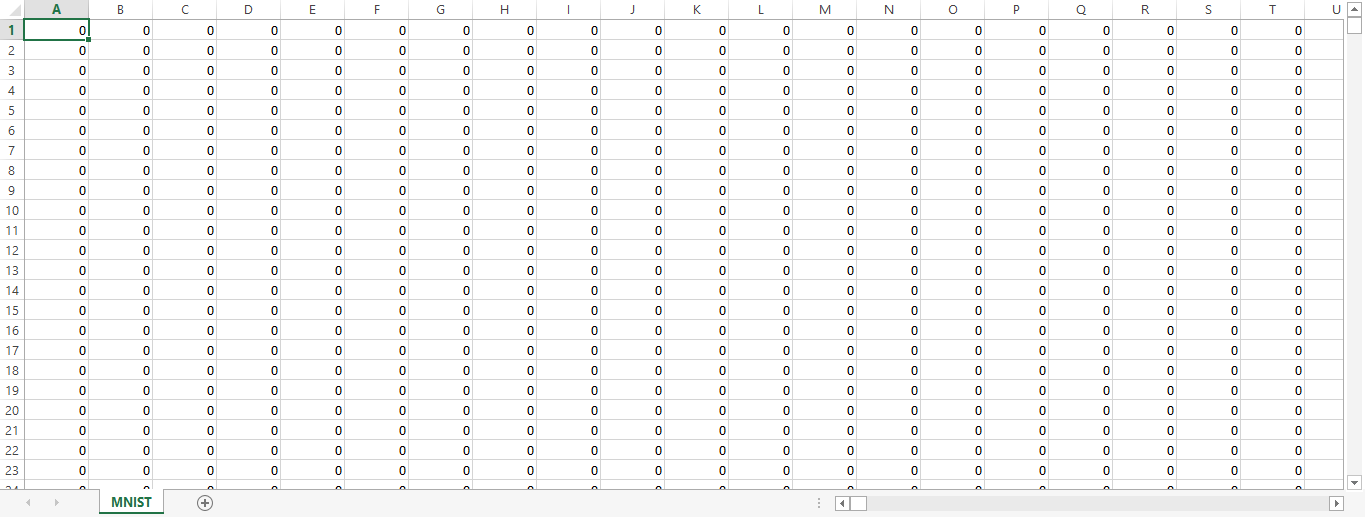


Figure 14: CSV format of the data from the 28x28 pixel image

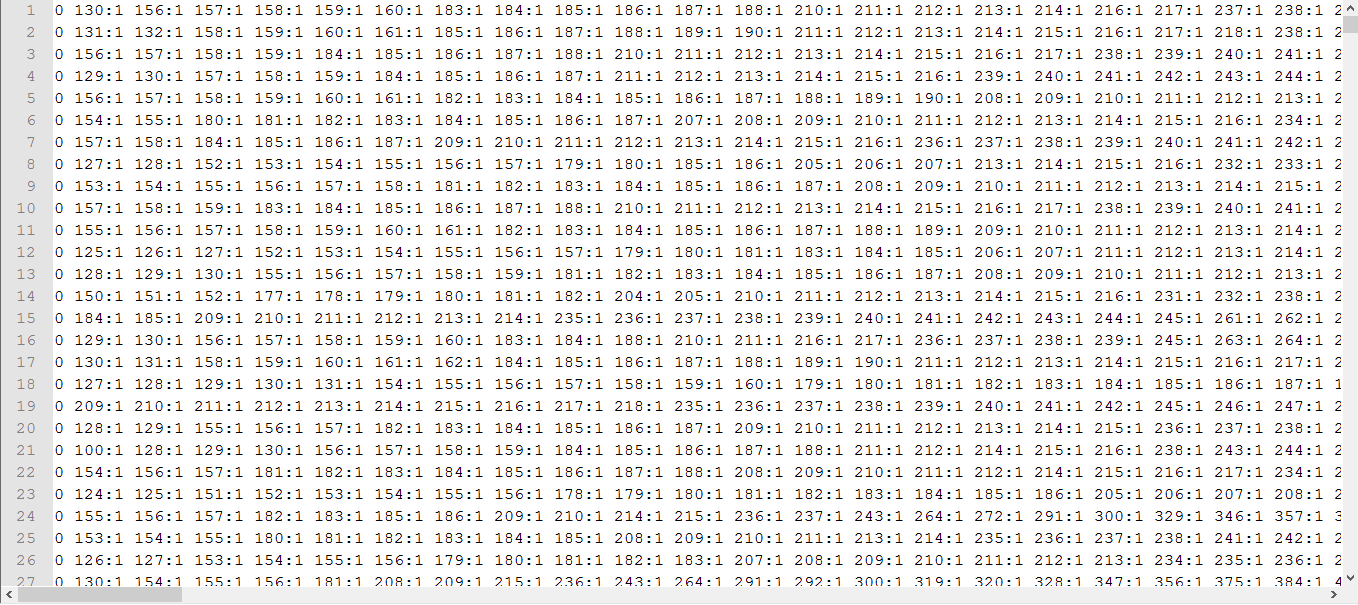


Figure 15: LIBSVM format of the data from the CSV file

# **3.4 Training**

Before the prediction phase, training has to be done first. LIBSVM is used and RBF is selected as the kernel function. The training phase outputs a model file and this model file will be used for predicting the segmented images from the handwritten equation (see Appendix B).

# **3.5 Testing**

Testing will follow after the acquisition of the image information of the segmented images from the handwritten equation. The image information along with the model from the training will be fed to a function for prediction. It will then output its predictions on what the segmented image from the handwritten equation is. Based on the results, it will display the operand apart from the operators.

# **3.6 Determining Operands and Operators**

The researchers used linked list to store data. There are three linked list to be used. The first one is a linked list for the predicted characters from the equation. The second one is a linked list for the operands. The last one is a linked list for the operators. Every time the classifier has predicted a character in an equation, it saves it into a node. This will be on loop until all predictions from the equation are in the linked list. The first node will be pointed by a pointer and another pointer points to the characters next to it, both being checked what character they are. Below is a list of different scenarios on how to determine operands and operators.

**Scenario 1:** if a digit is followed by a digit

Solution: Both digits will be saved into the linked list for operands

**Scenario 2:** if a digit is followed by an alphabet

Solution: There will be two cases for this scenario. First, the alphabet following the digit is also an operand (eg. 2a, 2x). Second, the alphabet following the digit is an operator (eg. 2 x 3). In cases where the alphabet is “x”, it will be confusing if it is an operand or an operator. The solution used is to check the alphabet. If it is an “a” or “b” or “c” or “y” or “z” (eg. 5z, 9c), both the digit and the alphabet will be saved into the linked list for operands. However, if the alphabet is an “x”, the next character is checked. If it is a number or a letter (eg. 2 x a, 2 x 4), the digit will be saved into the linked list for operands and the alphabet will be saved into the linked list for operators. However, if it is an operator (eg. 7a +), the digit and the alphabet will be saved into the linked list for operands.

**Scenario 3:** if a digit is followed by an operator

Solution: the digit will be saved into the linked list for operands and the operator will be saved into the linked list for operators

**Scenario 4:** if a digit is followed by an open or close parenthesis

Solution: the digit will be saved into the linked list for operand and the open or close parenthesis will be ignored

**Scenario 5:** if an alphabet is followed by an operator

Solution: the alphabet will be saved into the linked list for operands and the operator will be saved into the linked list for operators

**Scenario 6:** if an alphabet is followed by an open or close parenthesis

Solution: the alphabet will be saved into the linked list for operand and the open or close parenthesis will be ignored

**Scenario 7:** if an operator is followed by a digit

Solution: the operator will be saved into the linked list for operators and the digit will be saved into the linked list for operands

**Scenario 8:** if an operator is followed by an alphabet

Solution: the operator will be saved into the linked list for operators and the alphabet will be saved into the linked list for operands

**Scenario 9:** if a parenthesis is followed by a digit

Solution: the parenthesis will be ignored and the digit will be saved into the linked list for operands

**Scenario 10:** if a parenthesis is followed by an alphabet

Solution: the parenthesis will be ignored and the digit will be saved into the linked list for operands

# **CHAPTER 4: TESTING, RESULTS, AND OBSERVATIONS**

# **4.1 TESTING**

There are 5 models created for testing, which contains collected data from the researchers and from the MNIST database. The data used are the following:

From the researchers (see Appendix A):

* Digits ( 0 to 9 )
* Letters ( a, b, c, x, y, z )
* Operators ( +, -, /, = )
* Symbols ( (, ) )

From MNIST Database (see Appendix A):

* Digits ( 0 to 9 )

MODEL 1

500 each digits from MNIST (0 to 9)

1000 each letters, operators, and symbols from researchers (a, b, c, x, y, z, +, -, /, =, (, ))

Total: (500 x 10) + (1000 x 12) = 17000 training data

MODEL 2

1000 each digits from MNIST (0 to 9)

1000 each letters, operators, and symbols from researchers (a, b, c, x, y, z, +, -, /, =, (, ))

Total: (1000 x 10) + (1000 x 12) = 22000 training data

MODEL 3

2000 each digits from MNIST (0 to 9)

1000 each letters, operators, and symbols from researchers (a, b, c, x, y, z, +, -, /, =, (, ))

Total: (2000 x 10) + (1000 x 12) = 32000 training data

MODEL 4

500 each digits from MNIST (0 to 9)

500 each digits from researchers (0 to 9)

1000 each letters, operators, and symbols from researchers (a, b, c, x, y, z, +, -, /, =, (, ))

Total: (500 x 10) + (500 x 10) + (1000 x 12) = 22000 training data

MODEL 5

1000 each digits from researchers (0 to 9)

1000 each letters, operators, and symbols from researchers (a, b, c, x, y, z, +, -, /, =, (, ))

Total: (1000 x 10) + (1000 x 12) = 22000 training data

Provided with the number of data to be trained in each model listed above, the data is fed to the libsvm. With the 5 models (see Appendix B), there will be three scenarios conducted for experimentation which are listed below:

* Experiment 1 [12 equations for digits as operands]
  + [A] 5 equations consists of digits, operators, symbols with spaces
  + [A] 1 equation consist of digits, operators, symbols with spaces with poor lighting
  + [B] 5 equations consists of digits, operators, symbols without spaces
  + [B] 1 equation consists of digits, operators, symbols without spaces

with poor lighting

* Experiment 2 [12 equations for digits and letters as operands]
  + [A] 5 equations consists of digits, letters, operators, symbols with spaces
  + [A] 1 equation consists of digits, letters, operators, symbols with spaces

with poor lighting

* + [B] 5 equations consists of digits, letters, operators, symbols without spaces
  + [B] 1 equation consists of digits, letters, operators, symbols without spaces

with poor lighting

* Experiment 3 [10 equations for digits and letters as operands in a colored background]
  + [A] 5 equations consists of digits, letters, operators, symbols with spaces written in a non-white background
  + [A] 1 equation consists of digits, letters, operators, symbols with spaces written in a non-white background with poor lighting
  + [B] 5 equations consists of digits, letters, operators, symbols without spaces written in a non-white background
  + [B] 1 equation consists of digits, letters, operators, symbols without spaces written in a non-white background with poor lighting

During testing, each equations undergo preprocessing, segmentation, resizing, saving of the needed values, and converting it into a file such that it can be fed for prediction with the model to be used. After the classifier has predicted, operands and operators can then be determined. The accuracy of the classifier’s prediction greatly affects the determination of operands and operators from the handwritten mathematical equation. In attaining the accuracy, the number of correctly classified operators and operands was divided by the total number of operands and operators present in the equation.

# **4.2 RESULTS**

Listed below are the experiments done by the researchers.

EXPERIMENT 1 – numbers from 0 – 9, operators, & symbols

EXPERIMENT 2 – numbers from 0 – 9, letters, operators, & symbols

EXPERIMENT 3 – numbers from 0 – 9, letters, operators, & symbols in a non-white background

A – Equations with spaces

B – Equations without spaces

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 1: Numbers from 0 – 9 , Operators & Symbols** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **1 2 3 + 4 5 6 = 5 7 9** | **a z 3 + a a 6 4 a 3y** | **27.27%** | **3, 64a, 3y** | **+** | **20%** |
| **7 8 9 – 2 1 = 7 8 9** | **3 z y – z 1 9 8 6 a** | **20%** | **3z, y, 1986a** | **-** | **20%** |
| **3 0 x 4 3** | **3 0 x a 3** | **80%** | **30, 3** | **x** | **66.6%** |
| **6 5 / 8 7** | **6 a / z y** | **40%** | **6a, z, y** | **/** | **33.3%** |
| **( 9 1 + 8 6 ) – 3 7** | **( a 8 + a 6 ) - 3 y** | **40%** | **8, 6, 3y** | **+, -** | **40%** |
| **With poor lighting** | **( 9 1 + 8 6 ) – 3 7** | **( y 8 + - - 3** | **40%** | **8, 3** | **+ -, -** | **20%** |
| **(B)**  **Without spaces** | **65 x 7 8** | **a a x z a** | **20%** | **a** | **None** | **0%** |
| **20 + 9 = 2 9** | **a a 8 y 9 z a** | **0%** | **8y, 9z, a** | **None** | **0%** |
| **1 3 4 – 60 = 74** | **8 3 y – 6 a 2 y 4** | **44.44%** | **83y, 6a, 2y, 4** | **-** | **20%** |
| **10 / ( 74 + 8 )** | **8 a / ( z a 8 a )** | **33.33%** | **8a, 8a** | **/** | **20%** |
| **13 + 78 = 9 1** | **8 a + y a 9 y 8** | **12.5%** | **8a, 9y, 8** | **+** | **20%** |
| **With poor lighting** | **13 + 78 = 9 1** | **a 8 a a a – 8 a a** | **0%** | **None** | **-** | **0%** |

**MODEL 1**

Table 1: Model Recognition and Operand and Operator Determining Rate of Experiment 1 using Model 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 2: Numbers from 0 – 9, Letters, Operators & Symbols** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **4 5 a + 6 3 b** | **4 a a + 6 y b** | **71.43%** | **4a, a, 6y, b** | **+** | **33.3%** |
| **7 0 c – 1 6 = 2 8 x** | **y a 6 – 1 6 9 2 a x** | **50%** | **6, 1692a, x** | **-** | **20%** |
| **9 4 / ( 2 5 + 7 y ) + 8 6 z** | **0 a / ( a a + y y ) + a 6 z** | **57.14%** | **0a, 6z** | **/, +, +** | **42.9%** |
| **3 c – 5 a x 10** | **a c – 5 a x 8 0** | **75%** | **5a, 80** | **-, x** | **60%** |
| **2 z x 9 a = 5 b** | **z z x y a = a b** | **62.5%** | **a, a, b** | **=** | **20%** |
| **With poor lighting** | **2 z x 9 a = 5 b** | **z – x a** | **37.5%** | **z, x, a** | **-** | **0%** |
| **(B)**  **Without spaces** | **26 a + 9 4b** | **a 6 a + y a b** | **57.14%** | **6a, b** | **+** | **33.3%** |
| **6c / ( 3a – 4 )** | **6 c / ( a a –a )** | **77.78%** | **6c, a** | **/, -** | **60%** |
| **2y x 7x = 20** | **a y x 3 9 z 0** | **37.5%** | **39z, 0** | **x** | **20%** |
| **2 5z – 1 8 c x 40** | **a a a – a a c x y 0** | **40%** | **a, c, 0** | **-, x** | **40%** |
| **94 a x 6 2b** | **y a a x 6 a b** | **57.14%** | **a, 6a, b** | **x** | **33.3%** |
| **With poor lighting** | **94 a x 6 2b** | **y a x 6 a a x 6 a** | **42.85%** | **6a, a, 6a** | **x, x** | **33.3%** |

Table 2: Model Recognition and Operand and Operator Determining Rate of Experiment 2 using Model 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 3: Numbers from 0 – 9, Letters, Operators & Symbols (Non-white background)** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **3 c – 5 a x 1 0** | **3 c – a a x 1 0** | **87.5%** | **3c, 10** | **-, x** | **80%** |
| **2 z x 9 a = 5 b** | **z z x y a 2 5 b** | **62.5%** | **25b** | **None** | **0%** |
| **4 5 a + 6 3 b** | **4 a a 8 6 3 b** | **71.43%** | **4a, 863b** | **None** | **0%** |
| **( 9 1 + 8 6 ) – 3 7** | **1 a 8 4 a 6 1 – 3 8** | **30%** | **1a, 84a, 61, 38** | **-** | **20%** |
| **3 0 x 4 3** | **3 a x a 3** | **60%** | **3a, 3** | **x** | **33.3%** |
| **With poor lighting** | **3 0 x 4 3** | **a a** | **0%** | **None** | **None** | **0%** |
| **(B)**  **Without spaces** | **94 a x 6 2b** | **a a a x 6 z b** | **57.14%** | **a, 6z, b** | **x** | **33.3%** |
| **2x x 7x = 20** | **z x x z x 9 z a** | **37.5%** | **z, 9z, a** | **x, x** | **20%** |
| **65 x 7 8** | **6 5 x a z** | **60%** | **65, a, z** | **x** | **66.6%** |
| **4 a x 6 2b** | **a a x 6 z a** | **50%** | **6z, a** | **x** | **33.3%** |
| **20 + 9 = 2 9** | **z 0 8 a 2 z a** | **14.29%** | **08a, 2z, a** | **None** | **0%** |
| **With poor lighting** | **20 + 9 = 2 9** | **a – 8 1 2** | **0%** | **a, 812** | **-** | **0%** |

Table 3: Model Recognition and Operand and Operator Determining Rate of Experiment 3 using Model 1

**MODEL 2**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 1: Numbers from 0 – 9 , Operators & Symbols** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **1 2 3 + 4 5 6 = 5 7 9** | **a z 3 + a 5 6 7 a 3 y** | **36.36%** | **a, 567a, 3y** | **+** | **20%** |
| **7 8 9 – 2 1 = 7 8 9** | **3 z y – z 1 7 8 6 8** | **20%** | **3z, y, 17868** | **-** | **20%** |
| **3 0 x 4 3** | **3 0 x a 3** | **80%** | **30, 3** | **x** | **66.6%** |
| **6 5 / 8 7** | **6 5 / 8 y** | **80%** | **65, 8y** | **/** | **66.6%** |
| **( 9 1 + 8 6 ) – 3 7** | **( a 8 + a 6 ) – 3 y** | **60%** | **8, 6, 3y** | **+, -** | **40%** |
| **With poor lighting** | **( 9 1 + 8 6 ) – 3 7** | **( y 2 + - - 3** | **40%** | **2, 3** | **+-, -** | **20%** |
| **(B)**  **Without spaces** | **65 x 7 8** | **6 5 x z a** | **60%** | **65, z, a** | **x** | **66.6%** |
| **20 + 9 = 2 9** | **a a 8 y 2 z a** | **0%** | **8y, 2z, a** | **None** | **0%** |
| **1 3 4 – 60 = 74** | **8 3 y – 6 a 2 y 4** | **44.44%** | **83y, 6a, 2y, 4** | **-** | **20%** |
| **10 / ( 74 + 8 )** | **8 0 / ( z 4 8 a )** | **55.56%** | **80, 48a** | **/** | **20%** |
| **13 + 78 = 9 1** | **8 a 8 y a 7 y 8** | **0%** | **8a, 8y, 7y, 8** | **None** | **0%** |
| **With poor lighting** | **13 + 78 = 9 1** | **6 8 a a a - 8 a a** | **0%** | **68a, 8a, a** | **-** | **0%** |

Table 4: Model Recognition and Operand and Operator Determining Rate of Experiment 1 using Model 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 2: Numbers from 0 – 9, Letters, Operators & Symbols** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **4 5 a + 6 3 b** | **4 a a + 6 y b** | **71.43%** | **4a, a, 6y, b** | **+** | **33.3%** |
| **7 0 c – 1 6 = 2 8 x** | **y a 6 – 1 6 7 2 a x** | **50%** | **6, 1672a, x** | **-** | **20%** |
| **9 4 / ( 2 5 + 7 y ) + 8 6 z** | **0 a / ( a 5 + y y ) + 8 6 z** | **71.43%** | **0a, 5, 86z** | **/, +, +** | **57.1%** |
| **3 c – 5 a x 10** | **3 c – 5 a x 8 0** | **87.5%** | **3c, 5a, 80** | **-, x** | **80%** |
| **2 z x 9 a = 5 b** | **z z x y a = 5 b** | **75%** | **a, 5 b** | **=** | **40%** |
| **With poor lighting** | **2 z x 9 a = 5 b** | **z – x a** | **37.5%** | **z, x, a** | **-** | **0%** |
| **(B)**  **Without spaces** | **26 a + 9 4b** | **a 6 a + y 4 b** | **71.43%** | **6a, 4b** | **+** | **33.3%** |
| **6c / ( 3a – 4 )** | **6 c / ( a a – a )** | **77.78%** | **6c, a** | **/, -** | **60%** |
| **2y x 7x = 20** | **a y x 3 7 20** | **50%** | **3720** | **x** | **20%** |
| **2 5z – 1 8 c x 40** | **a a a – 8 8 c x y 0** | **50%** | **a, 88c, 0** | **-, x** | **40%** |
| **94 a x 6 2b** | **y 8 a x 6 a b** | **57.14%** | **8a, 6a, b** | **x** | **33.3%** |
| **With poor lighting** | **94 a x 6 2b** | **y a x 6 a a x 6 a** | **42.85%** | **6a, 6a** | **x, x** | **33.3%** |

Table 5: Model Recognition and Operand and Operator Determining Rate of Experiment 2 using Model 2

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 3: Numbers from 0 – 9, Letters, Operators & Symbols (Non-white background)** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **3 c – 5 a x 1 0** | **3 c – a a x 1 0** | **87.5%** | **3c, 10** | **-, x** | **80%** |
| **2 z x 9 a = 5 b** | **2 z x y a 2 5 b** | **75%** | **2z, 25b** | **x** | **40%** |
| **4 5 a + 6 3 b** | **4 a a 8 6 3 b** | **71.43%** | **4a, 863b** | **None** | **0%** |
| **( 9 1 + 8 6 ) – 3 7** | **1 a 8 6 8 6 1 – 3 8** | **40%** | **1a, 86861,38** | **-** | **20%** |
| **3 0 x 4 3** | **3 a x a 3** | **60%** | **3a, 3** | **x** | **33.3%** |
| **With poor lighting** | **3 0 x 4 3** | **a a** | **0%** | **None** | **None** | **0%** |
| **(B)**  **Without spaces** | **94 a x 6 2b** | **a a a x 6 z b** | **57.14%** | **a, 6z, b** | **x** | **33.3%** |
| **2x x 7x = 20** | **z x x 2 x 9 2 0** | **62.5%** | **z, 2, 920** | **x, x, x** | **20%** |
| **65 x 7 8** | **6 5 x 3 z** | **60%** | **65, 3z** | **x** | **66.6%** |
| **4 a x 6 2b** | **a a x 6 z a** | **50%** | **6z, a** | **x** | **33.3%** |
| **20 + 9 = 2 9** | **z 0 6 a 2 2 a** | **28.57%** | **06a, 22a** | **None** | **0%** |
| **With poor lighting** | **20 + 9 = 2 9** | **a – 8 1 2** | **0%** | **a, 812** | **-** | **0%** |

Table 6: Model Recognition and Operand and Operator Determining Rate of Experiment 3 using Model 2

**MODEL 3**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 1: Numbers from 0 – 9 , Operators & Symbols** | | | | | | |
|  | **Input** | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **1 2 3 + 4 5 6 = 5 7 9** | **8 2 3 + a 5 6 2 a 3 y** | **45.45%** | **823, 562a, 3y** | **+** | **20%** |
| **7 8 9 – 2 1 = 7 8 9** | **3 z y – z 1 2 8 6 8** | **20%** | **3z, y, 12868** | **-** | **20%** |
| **3 0 x 4 3** | **30 x a 3** | **80%** | **30, 3** | **x** | **66.6%** |
| **6 5 / 8 7** | **6 5 / 8 y** | **80%** | **65, 8y** | **/** | **66.6%** |
| **( 9 1 + 8 6 ) – 3 7** | **( a 8 + a 6 ) – 3 y** | **60%** | **8, 6, 3y** | **+,-** | **40%** |
| **With poor lighting** | **( 9 1 + 8 6 ) – 3 7** | **( y 1 + - - 3** | **50%** | **1, 3** | **+-, -** | **20%** |
| **(B)**  **Without spaces** | **65 x 7 8** | **6 5 x 2 a** | **60%** | **65, 2a** | **x** | **66.6%** |
| **20 + 9 = 2 9** | **2 a 8 y 2 z a** | **14.29%** | **2a, 8y, 2z, a** | **None** | **0%** |
| **1 3 4 – 60 = 74** | **8 3 y – 6 0 2 y 4** | **55.56%** | **83y, 602y, 4** | **-** | **20%** |
| **10 / ( 74 + 8 )** | **8 0 / ( z 4 8 8 )** | **66.67%** | **80, 488** | **/** | **20%** |
| **13 + 78 = 9 1** | **8 3 8 y a 2 y 8** | **12.5%** | **838y, 2y, 8** | **None** | **0%** |
| **With poor lighting** | **13 + 78 = 9 1** | **6 8 a 0 0 – 8 0 0** | **0%** | **6800, 800** | **-** | **0%** |

Table 7: Model Recognition and Operand and Operator Determining Rate of Experiment 1 using Model 3

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 2: Numbers from 0 – 9, Letters, Operators & Symbols** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **4 5 a + 6 3 b** | **4 a a + 6 y b** | **71.43%** | **4a, a, 6y, b** | **+** | **33.3%** |
| **7 0 c – 1 6 = 2 8 x** | **y a 6 – 1 6 2 2 8 x** | **60%** | **6, 16228x** | **-** | **20%** |
| **9 4 / ( 2 5 + 7 y ) + 8 6 z** | **0 a / ( 2 5 + y y ) + 8 6 z** | **78.57%** | **0a, 25, 86z** | **/, +, +** | **71.43%** |
| **3 c – 5 a x 10** | **3 c – 5 a x 8 0** | **87.5%** | **3c, 5a, 80** | **-, x** | **80%** |
| **2 z x 9 a = 5 b** | **2 z x y a 2 5 b** | **75%** | **2z, 25b** | **x** | **40%** |
| **With poor lighting** | **2 z x 9 a = 5 b** | **2 – 8 8** | **12.5%** | **2, 88** | **-** | **0%** |
| **(B)**  **Without spaces** | **26 a + 9 4b** | **a 6 a + y 4 b** | **71.43%** | **6a, 4b** | **+** | **33.3%** |
| **6c / ( 3a – 4 )** | **6 c / ( a a – a )** | **77.78%** | **6c, a** | **/, -** | **60%** |
| **2y x 7x = 20** | **a y x 3 2 2 0** | **50%** | **3220** | **x** | **20%** |
| **2 5z – 1 8 c x 40** | **2 a 2 – 8 8 c x 5 0** | **60%** | **2a, 2, 88c, 50** | **-, x** | **40%** |
| **94 a x 6 2b** | **y 9 a x 6 a b** | **57.14%** | **9a, 6a, b** | **x** | **33.3%** |
| **With poor lighting** | **94 a x 6 2b** | **y a x 6 a 8 x 6 2** | **42.85%** | **68, 62** | **x, x** | **33.3%** |

Table 8: Model Recognition and Operand and Operator Determining Rate of Experiment 2 using Model 3

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 3: Numbers from 0 – 9, Letters, Operators & Symbols (Non-white background)** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **3 c – 5 a x 1 0** | **3 0 – a a x 1 0** | **75%** | **30, 10** | **-, x** | **60%** |
| **2 z x 9 a = 5 b** | **2 z x 8 a 2 5 b** | **75%** | **2z, 8a, 25b** | **x** | **40%** |
| **4 5 a + 6 3 b** | **9 a a 8 6 3 b** | **57.14%** | **9a, 863b** | **None** | **0%** |
| **( 9 1 + 8 6 ) – 3 7** | **1 a 8 6 8 6 1 – 3 8** | **40%** | **1a, 86861, 38** | **-** | **20%** |
| **3 0 x 4 3** | **3 0 x a 3** | **80%** | **30, 3** | **x** | **66.6%** |
| **With poor lighting** | **3 0 x 4 3** | **a a** | **0%** | **None** | **None** | **0%** |
| **(B)**  **Without spaces** | **94 a x 6 2b** | **a a a x 6 2 b** | **71.43%** | **a, 62b** | **X** | **66.6%** |
| **2x x 7x = 20** | **z x x 2 x 2 2 0** | **62.5%** | **z, 2, 220** | **x, x, x** | **20%** |
| **65 x 7 8** | **6 5 x 2 z** | **60%** | **65, 2z** | **X** | **66.6%** |
| **4 a x 6 2b** | **a a x 6 z 8** | **50%** | **6z, 8** | **X** | **33.3%** |
| **20 + 9 = 2 9** | **2 0 6 82 2 a** | **42.86%** | **206822a** | **None** | **0%** |
| **With poor lighting** | **20 + 9 = 2 9** | **a – 8 1 2** | **0%** | **a, 812** | **-** | **0%** |

Table 9: Model Recognition and Operand and Operator Determining Rate of Experiment 3 using Model 3

**MODEL 4**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 1: Numbers from 0 – 9 , Operators & Symbols** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **1 2 3 + 4 5 6 = 5 7 9** | **a 2 3 + 4 5 6 9 5 7 9** | **81.82%** | **23, 4569579** | **+** | **20%** |
| **7 8 9 – 2 1 = 7 8 9** | **3 z 9 – 2 1 7 8 6 8** | **40%** | **3z, 217868** | **-** | **20%** |
| **3 0 x 4 3** | **3 0 x 4 3** | **100%** | **30, 43** | **x** | **100%** |
| **6 5 / 8 7** | **6 5 / 8 9** | **80%** | **65, 89** | **/** | **66.6%** |
| **( 9 1 + 8 6 ) – 3 7** | **( 9 1 + 8 6 ) – 3 7** | **100%** | **91, 86, 37** | **+,-** | **100%** |
| **With poor lighting** | **( 9 1 + 8 6 ) – 3 7** | **( 9 1 + - - 3** | **60%** | **91, 3** | **+, -, -** | **60%** |
| **(B)**  **Without spaces** | **65 x 7 8** | **6 5 x 7 8** | **100%** | **65, 78** | **x** | **100%** |
| **20 + 9 = 2 9** | **a a 8 9 2 z 9** | **28.57%** | **892z, 9** | **None** | **0%** |
| **1 3 4 – 60 = 74** | **1 3 4 – 6 0 2 7 4** | **88.89%** | **134, 60274** | **-** | **40%** |
| **10 / ( 74 + 8 )** | **1 0 / ( 7 4 8 8 )** | **88.89%** | **10, 7488** | **/** | **40%** |
| **13 + 78 = 9 1** | **1 3 8 7 8 2 9 1** | **75%** | **13878291** | **None** | **0%** |
| **With poor lighting** | **13 + 78 = 9 1** | **a 8 a a a – 8 a a** | **0%** | **8a, 8a, a** | **-** | **0%** |

Table 10: Model Recognition and Operand and Operator Determining Rate of Experiment 1 using Model 4

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 2: Numbers from 0 – 9, Letters, Operators & Symbols** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **4 5 a + 6 3 b** | **4 5 a + 6 3 b** | **100%** | **45a, 63b** | **+** | **100%** |
| **7 0 c – 1 6 = 2 8 x** | **7 0 6 – 1 6 9 2 8 x** | **80%** | **706, 16928x** | **-** | **20%** |
| **9 4 / ( 2 5 + 7 y ) + 8 6 z** | **0 a / ( 2 5 + 7 y ) +8 6 z** | **85.71%** | **0a, 25, 7y, 86z** | **/, +, +** | **85.71%** |
| **3 c – 5 a x 10** | **3 c – 5 a x 1 0** | **100%** | **3c, 5a, 10** | **-, x** | **100%** |
| **2 z x 9 a = 5 b** | **2 z x 9 a = 5 b** | **100%** | **2z, 9a, 5b** | **x, =** | **100%** |
| **With poor lighting** | **2 z x 9 a = 5 b** | **2 - x a** | **37.5%** | **2, x, a** | **-** | **0%** |
| **(B)**  **Without spaces** | **26 a + 9 4b** | **a 6 a + 9 4 b** | **85.71%** | **6a, 94b** | **+** | **66.6%** |
| **6c / ( 3a – 4 )** | **6 c / ( 3 a – 4 )** | **100%** | **6c, 3a, 4** | **/, -** | **100%** |
| **2y x 7x = 20** | **a y x 3 9 2 0** | **50%** | **3920** | **x** | **20%** |
| **2 5z – 1 8 c x 40** | **2 a a – 8 8 c x 4 0** | **70%** | **2a, a, 88c, 40** | **-, x** | **60%** |
| **9 4 a x 6 2b** | **9 4 a x 6 2 b** | **100%** | **94a, 62b** | **x** | **100%** |
| **With poor lighting** | **9 4 a x 6 2b** | **9 a x 6 a a x 6 a** | **57.14%** | **9a, 6a, a, 6a** | **x, x** | **33.3%** |

Table 11: Model Recognition and Operand and Operator Determining Rate of Experiment 2 using Model 4

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 3: Numbers from 0 – 9, Letters, Operators & Symbols (Non-white background)** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **3 c – 5 a x 1 0** | **3 0 – a a x 1 0** | **75%** | **30, 10** | **-, x** | **60%** |
| **2 z x 9 a = 5 b** | **2 z x 9 a 2 5 b** | **87.5%** | **2z, 9a, 25b** | **x** | **60%** |
| **4 5 a + 6 3 b** | **4 5 a 8 6 3 b** | **85.71%** | **45a, 863b** | **None** | **33.3%** |
| **( 9 1 + 8 6 ) – 3 7** | **1 9 1 4 8 6 1 – 3 8** | **60%** | **1914861, 38** | **-** | **20%** |
| **3 0 x 4 3** | **3 0 x 4 3** | **100%** | **30, 43** | **x** | **100%** |
| **With poor lighting** | **3 0 x 4 3** | **a a** | **0%** | **None** | **None** | **0%** |
| **(B)**  **Without spaces** | **94 a x 6 2b** | **9 4 a x 6 2 b** | **100%** | **94a, 62b** | **x** | **100%** |
| **2x x 7x = 20** | **2 x x 7 x 9 2 0** | **87.5%** | **2x, 7, 920** | **x, x** | **40%** |
| **65 x 7 8** | **6 5 x 7 z** | **80%** | **65, 7z** | **x** | **66.6%** |
| **4 a x 6 2b** | **4 a x 6 z 8** | **66.67%** | **4a, 6z, 8** | **x** | **66.6%** |
| **20 + 9 = 2 9** | **2 0 4 9 2 2 8** | **57.14%** | **2049228** | **None** | **0%** |
| **With poor lighting** | **20 + 9 = 2 9** | **a – 8 1 2** | **0%** | **a, 812** | **-** | **0%** |

Table 12: Model Recognition and Operand and Operator Determining Rate of Experiment 3 using Model 4

**MODEL 5**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 1: Numbers from 0 – 9 , Operators & Symbols** | | | | | | |
|  | **Input** | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **1 2 3 + 4 5 6 = 5 7 9** | **a z 3 + 4 5 6 x 5 7 9** | **72.73%** | **3, 456, 579** | **+, x** | **60%** |
| **7 8 9 – 2 1 = 7 8 9** | **3 z 9 – 2 + x z 6 8** | **30%** | **3z, 9, 2, 68** | **-, +** | **20%** |
| **3 0 x 4 3** | **3 0 x 4 3** | **100%** | **30, 43** | **x** | **100%** |
| **6 5 / 8 7** | **6 5 / 8 7** | **100%** | **65, 87** | **/** | **100%** |
| **( 9 1 + 8 6 ) – 3 7** | **( 9 1 + 8 6 ) – 3 7** | **100%** | **91, 86, 37** | **+, -** | **100%** |
| **With poor lighting** | **( 9 1 + 8 6 ) – 3 7** | **( a 8 + a 6 ) - 3 y** | **60%** | **8, 6, 3y** | **+, -** | **40%** |
| **(B)**  **Without spaces** | **65 x 7 8** | **6 5 x 7 8** | **100%** | **65, 78** | **x** | **100%** |
| **20 + 9 = 2 9** | **a 3 a 9 a z 9** | **28.57%** | **3a, 9a, 9** | **None** | **0%** |
| **1 3 4 – 60 = 74** | **1 3 4 – 6 0 a 7 4** | **88.89%** | **134, 60a, 74** | **-** | **60%** |
| **10 / ( 74 + 8 )** | **1 0 / ( 7 4 a 8 )** | **88.89%** | **10, 74a, 8** | **/** | **60%** |
| **13 + 78 = 9 1** | **1 3 + 7 8 = 9 1** | **100%** | **13, 78, 91** | **+, =** | **100%** |
| **With poor lighting** | **13 + 78 = 9 1** | **a + a a a - + a a** | **0%** | **a, a, a, a** | **+, -+** | **20%** |

Table 13: Model Recognition and Operand and Operator Determining Rate of Experiment 1 using Model 5

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 2: Numbers from 0 – 9, Letters, Operators & Symbols** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **4 5 a + 6 3 b** | **4 5 a + 6 3 b** | **100%** | **45a, 63b** | **+** | **100%** |
| **7 0 c – 1 6 = 2 8 x** | **7 0 6 - + 6 = 2 8 x** | **80%** | **706, 6, 28x** | **-+, =** | **60%** |
| **9 4 / ( 2 5 + 7 y ) + 8 6 z** | **a a / ( 2 5 + 7 y ) + 8 6 z** | **85.71%** | **25, 7y, 86z** | **/, +, +** | **85.7%** |
| **3 c – 5 a x 10** | **3 c – 5 a x 1 0** | **100%** | **3c, 5a, 10** | **-, x** | **100%** |
| **2 z x 9 a = 5 b** | **2 z x 9 a = 5 b** | **100%** | **2z, 9a, 5b** | **x ,=** | **100%** |
| **With poor lighting** | **2 z x 9 a = 5 b** | **2 - x a** | **37.5%** | **2, x, a** | **-** | **0%** |
| **(B)**  **Without spaces** | **26 a + 9 4b** | **a 6 a + 9 4 b** | **85.71%** | **6a, 94b** | **+** | **66.6%** |
| **6c / ( 3a – 4 )** | **6 c / ( 3 a – 4 )** | **100%** | **6c, 3a, 4** | **/,-** | **100%** |
| **2y x 7x = 20** | **a y x a = 2 0** | **62.5%** | **20** | **=** | **40%** |
| **2 5z – 1 8 c x 40** | **2 a a – a 8 c x 4 0** | **70%** | **2a, a, 8c, 40** | **-, x** | **40%** |
| **94 a x 6 2b** | **9 4 a x 6 2 b** | **100%** | **94a, 62b** | **x** | **100%** |
| **With poor lighting** | **94 a x 6 2b** | **9 a x b a a x 6 a** | **42.85%** | **9a, a, 6a** | **x, x** | **33.3%** |

Table 14: Model Recognition and Operand and Operator Determining Rate of Experiment 2 using Model 5

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Experiment 3: Numbers from 0 – 9, Letters, Operators & Symbols (Non-white background)** | | | | | | |
| **Input** | | **Model Recognition Results** | | **Operators and Operands Results** | | |
| **(A)**  **With spaces** | **Original Equation** | **Recognition per character** | **Recognition Rate** | **Operands** | **Operators** | **Determining Rate** |
| **3 c – 5 a x 1 0** | **3 6 – a a x + 0** | **62.5%** | **36, x, 0** | **-, +** | **40%** |
| **2 z x 9 a = 5 b** | **2 z x 9 a a 5 b** | **87.5%** | **2z, 9a, 5b** | **x** | **80%** |
| **4 5 a + 6 3 b** | **4 5 a a 6 3 b** | **85.71%** | **45a, 63b** | **None** | **66.6%** |
| **( 9 1 + 8 6 ) – 3 7** | **+ 9 1 a 8 6 + - 3 +** | **60%** | **91a, 86, 3** | **+, +-, +** | **40%** |
| **3 0 x 4 3** | **3 0 x 4 3** | **100%** | **30, 43** | **x** | **100%** |
| **With poor lighting** | **3 0 x 4 3** | **a a** | **0%** | **None** | **None** | **0%** |
| **(B)**  **Without spaces** | **94 a x 6 2b** | **9 4 a x 6 2 b** | **100%** | **94a, 62b** | **x** | **100%** |
| **2x x 7x = 20** | **2 x x 7 x a z 0** | **75%** | **2x, 7, 0** | **x, x** | **40%** |
| **65 x 7 8** | **6 5 x 7 z** | **80%** | **65, 7z** | **x** | **66.6%** |
| **4 a x 6 2b** | **4 a x 6 z a** | **66.67%** | **4a, 6z, a** | **x** | **66.6%** |
| **20 + 9 = 2 9** | **2 0 a a a 2 a** | **42.86%** | **20a, 2a** | **None** | **0%** |
| **With poor lighting** | **20 + 9 = 2 9** | **a - a + a** | **0%** | **a, a, a** | **-, x** | **0%** |

Table 15: Model Recognition and Operand and Operator Determining Rate of Experiment 3 using Model 5

# **4.3 Accuracy Calculations**

Getting the accuracy for each models by experiments for equations with spaces and without spaces

Accuracy = (eq1 + eq2 + eq3 + eq4 + eq5 + eq6) / n

Where:

eq1, eq2, eq3, eq4, eq5, eq6 = the accuracy of each equation in character recognition

n = number of equations being added

Below is a table of accuracy for each models by experiments for equations with spaces and without spaces.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MODELS | Experiment 1 | | Experiment 2 | | Experiment 3 | |
| A | B | A | B | A | B |
| MODEL 1 | 41.21 | 18.38 | 58.93 | 52.07 | 51.91 | 36.49 |
| MODEL 2 | 52.73 | 26.67 | 65.48 | 58.20 | 55.66 | 43.04 |
| MODEL 3 | 55.91 | 34.84 | 64.17 | 59.87 | 54.52 | 47.80 |
| MODEL 4 | 76.97 | 63.56 | 83.87 | 77.14 | 68.04 | 65.22 |
| MODEL 5 | 77.12 | 67.73 | 83.87 | 76.84 | 65.95 | 60.76 |

Table 16: Table of accuracy for each models by experiments for equations with spaces and without spaces

Getting the accuracy for each models by accuracies of equations with spaces and without spaces

AccuracyA = (exp1a + exp2a + exp3a) / n

AccuracyB = (exp1b + exp2b + exp3b) / n

Where:

exp1a, exp2a, exp3a = the accuracy of equation with spaces

exp1a, exp2a, exp3a = the accuracy of equation without spaces

n = number of accuracies being added

Below is a table of accuracies for each models by accuracies of equations with spaces and without spaces.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | MODEL 1 | MODEL 2 | MODEL 3 | MODEL 4 | MODEL 5 |
| A | 50.68 | 57.96 | 58.2 | 76.29 | 75.65 |
| B | 35.65 | 42.64 | 47.50 | 68.64 | 68.44 |

Table 17: Table of accuracies for each models by accuracies of equations with spaces and without spaces

Getting the accuracy for determining of operands and operators by experiments for equations with and without spaces

Accuracy = (eq1 + eq2 + eq3 + eq4 + eq5 + eq6) / n

Where:

eq1, eq2, eq3, eq4, eq5, eq6 = the accuracy of each equation

n = number of equations

Below is a table of accuracies for determining of operands and operators by experiments for equations with and without spaces.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| MODELS | Experiment 1 | | Experiment 2 | | Experiment 3 | |
| A | B | A | B | A | B |
| MODEL 1 | 33.32 | 10 | 29.37 | 36.65 | 22.22 | 25.53 |
| MODEL 2 | 38.87 | 17.77 | 38.4 | 36.65 | 28.88 | 25.53 |
| MODEL 3 | 38.87 | 17.77 | 40.79 | 36.65 | 31.1 | 31.08 |
| MODEL 4 | 61.1 | 30 | 67.62 | 63.32 | 45.55 | 45.53 |
| MODEL 5 | 70 | 56.67 | 74.28 | 63.32 | 54.43 | 45.53 |

Table 18: Table of accuracies for determining of operands and operators by experiments for equations with and without spaces

Getting the accuracies for determining of operands and operators by accuracies of equations with spaces and without spaces

AccuracyA = (exp1a + exp2a + exp3a) / n

AccuracyB = (exp1b + exp2b + exp3b) / n

Where:

exp1a, exp2a, exp3a = the accuracy of equation with spaces

exp1a, exp2a, exp3a = the accuracy of equation without spaces

n = number of accuracies being added

Below is a table of accuracies for determining of operands and operators by accuracies of equations with spaces and without spaces.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | MODEL 1 | MODEL 2 | MODEL 3 | MODEL 4 | MODEL 5 |
| A | 28.3 | 35.38 | 36.92 | 58.09 | 66.24 |
| B | 24.06 | 26.65 | 28.5 | 46.28 | 55.17 |

Table 19: Table of accuracies for determining of operands and operators by accuracies of equations with spaces and without spaces

Chart 1: This chart shows the percentage accuracies of each model for Experiment 1

Chart 2: This chart shows the percentage accuracies of each model for Experiment 2

Chart 3: This chart shows the percentage accuracies of each model for Experiment 3

Chart 4: This chart shows the percentage accuracies of each model for equations with and without spaces

Chart 5: This chart shows the overall accuracies of each model

Chart 6: This chart shows the operand and operator determining rate by the researcher’s solution using the model’s recognition for Experiment 1

Chart 7: This chart shows the operand and operator determining rate by the researcher’s solution using the model’s recognition for Experiment 2

Chart 8: This chart shows the operand and operator determining rate by the researcher’s solution using the model’s recognition for Experiment 3

Chart 9: This chart shows the summary of determining rate by the researcher’s solution using the models’ recognition

Chart 10: This chart shows the overall determining rate by the researcher’s solution using the models’ recognition

# **4.3 OBSERVATIONS**

The researchers have observed that the lighting affects an image. Shadows due to unequal lighting that is present in the image disrupted the thresholding process such as adding noise to the image or worse, overshadowing the equation which leads to poor recognition.

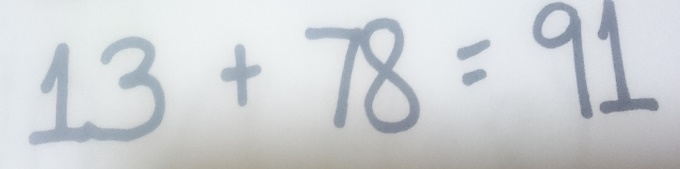
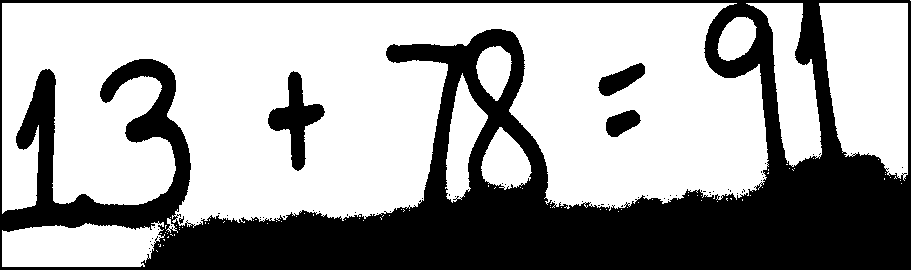
 

Figure 16 Overshadowed equation (left) and the result in thresholding (right)

# **CHAPTER 5: CONCLUSION AND RECOMMENDATION**

# **5.1 CONCLUSION**

Using SVM as a classifier, the system was able to achieve a recognition of 76.29% per character for equations with spaces and recognition of 68.64% per character for equations without spaces as the highest accuracy using Model 4. Moreover, after the equation was recognized by the classifier, the system was able to determine the operands and operators using the researcher’s solution with a 66.24% accuracy in equations with spaces and 55.17% accuracy in equations without spaces as the highest accuracy using the prediction of Model 5.

The experimental results showed that Model 4 has a character recognition rate of 72.47% accuracy as the highest for equations with and without spaces that consists of 500 MNIST digits (0-9), another 500 digits (0-9) from the researchers and 1000 letters, operators and symbols. Furthermore, in determining of operands and operators, Model 5 which consists of 1000 digits (0-9) from the researchers and 1000 letters, operators and symbols has a 60.71% recognition rate as the highest for equations with and without spaces.

Therefore, the researchers concluded that Model 4 is best for character recognition and Model 5 is best for determining of operands and operators for handwritten equations even without spaces by using SVM as a classifier and the algorithms used in the study. Although Model 4 has the highest character recognition rate, it does not perform well in determining operators.

# **5.2 RECOMMENDATION**

It is recommended to use Model 4 in recognizing of characters because of its accuracy rate of 72.47% which is the highest among the other models. However, in determining of operands and operators, Model 5 is recommended with a 72.045% accuracy for character recognition. Model 5 applied with the researcher’s solution, it gives a 60.71% recognition rate of the operands and operators which is the highest among other models. The researcher’s solution is recommended because of the high results of Model 5.

This can be used as reference for further research in this field of study. The researchers also recommend to try other image preprocessing algorithms that can be possibly be more effective in determination of operands and operators.

# **BIBLIOGRAPHY**

1. Introduction to Image Processing. Engineers Garage. Retrieved from [http://www.engineersgarage.com/articles/image-processing-tutorial-applications.](http://www.engineersgarage.com/articles/image-processing-tutorial-applications) 2012.

2. Solving Linear Equations from an Image using Ann. Retrieved from <http://esatjournals.org/Volumes/IJRET/2015V04/I02/IJRET20150402080.pdf>. 2015.

3. Ba-karait, N.O.S. Particle Swarm Optimization (PSO)-Based Clustering Algorithm for Handwritten Digits Recognition. MSc Thesis. Universiti Teknologi Malaysia, 2007.

4. K. Ntirogiannis, B. Gatos, & I. Pratikakis.(2014). Pattern Recognition Letters. Retrieved from [http://users.iit.demokritos.gr/~bgat/ntirogiannis2014.pdf.](http://users.iit.demokritos.gr/%7Ebgat/ntirogiannis2014.pdf)

5. Shah M., Jethava, G.B. (2013). A Literature Review On Hand Written Character Recognition. Retrieved from file:///C:/Users/admin/Downloads/2204.pdf.

6. Stathis, P., Kavallieratou, E., Papamarkos, N., (2008). An Evaluation Technique for Binarization Algorithms. Journal of Universal Computer Science, 14(18).

7. Bhatia, N. (2014). Optical Character Recognition Techniques: A Review. Retrieved from <http://www.ijarcsse.com/docs/papers/Volume_4/5_May2014/V4I5-> 0677.pdf.

8. Mukherjee, A., Kanrar, S., (2010). Enhancement of Image Resolution by Binarization.

International Journal of Computer Applications, 10(10), 0975 – 8887.

9. Haron,H., Nasien, D. & Yuhaniz, S. S. The Study of Handwriting Character Recognition (HCR) and Support Vector Machine (SVM). Retrieved from<http://comp.utm.my/pars/files/2013/04/The-Study-of-Handwriting-Character-> Recognition-HCR-and-Support-Vector-Machine-SVM.pdf.

10. Boor, Saeed Boor & Gulzar, Muhammad Ali. (2014).PCA based Mathematical Equation Solver Digital Image Processing.

11. Shrivastava, S. K., Gharde, S. S., (2010). Support Vector Machine for Handwritten Devanagari Numeral Recognition. Retrieved from [http://www.ijcaonline.org/volume7/number11/pxc3871769.pdf.](http://www.ijcaonline.org/volume7/number11/pxc3871769.pdf)

12. Greensted, Andrew. (2010). Otsu Thresholding. *The Lab Book Pages.* Retrieved from <http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html>

13. Zhong Qu & Li Hang. (2010). *Research on Image Segmentation Based on the Improved Otsu Algorithm.*

14. Suzuki, S. & Abe, K. (1985). Topological Structural Analysis of Digitized Binary Images by Border Following. CVGIP 30 1, pp 32-46.

15. *SVM- Support Vector Machine*. Retrieved from <http://www.support-vector-machines.org/>

16. Weston, Jason. *Support Vector Machine (and Statistical Learning Theory) Tutorial*. NEC Labs America.

17. [Mehryar Mohri](https://en.wikipedia.org/wiki/Mehryar_Mohri), Afshin Rostamizadeh & Ameet Talwalkar .(2012). *Foundations of Machine Learning*, The MIT Press [ISBN 9780262018258](https://en.wikipedia.org/wiki/Special:BookSources/9780262018258).

18. Hosch, William. (2014). *Machine learning*. Retrieved from <http://www.britannica.com/technology/machine-learning>

19. [TemelBilisim](https://www.youtube.com/channel/UCGGUqyBGoGYU9_IsKul9ryQ) .(2014). *Support Vector Machines (SVM) by Example.* Retrieved from <https://www.youtube.com/watch?v=5zRmhOUjjGY>

20*. Neuroph Java Neural Network Framework*. Retrieved from <http://neuroph.sourceforge.net/index.html>

21. *MNIST database of handwritten digits*. Retrieved from <http://yann.lecun.com/exdb/mnist/>

22. Ekberg, Filip. (2014, Feb. 10). *Understanding Peak-Finding*. Retrieved from <http://www.filipekberg.se/2014/02/10/understanding-peak-finding/>

23. Wang, Xiujan, Zheng Khangfeng, & Guo, Jun. (2006). *Inertial and Big Drop Fall Algorithm*. Retrieved from http://www.intjit.org/cms/journal/volume/12/4/124\_5.pdf

# **APPENDIX A**

**Data Sets**

The raw images of the data sets used to gather and train that has been presented in this paper may be found in a supplemental file named **datasets.zip**.

# **APPENDIX B**

**Data Sets Model**

The accompanying file contains the models that have been created for testing, which contains collected data sets from the researchers and from the MNIST database. These models will show how much it can recognize the handwritten images. See file **DatasetsModels.zip**.

**EMILYN MITZ N. BULAGAO**

**PERSONAL INFORMATION**

**Name:** Emilyn Mitz N. Bulagao

**Date of Birth:** July 09, 1995

**Age:** 22

**Home Address:** Dumaguete City

**Contact Details:** emilynnbulagao@su.edu.ph

0935 864 3012

**EDUCATION**

**College Education:** Silliman University

(2012 - 2016) Bachelors of Science in Information Technology

Hibbard Ave., Dumaguete City

**Secondary Education:** High School

(2008 - 2012) Dumaguete City

**WORK EXPERIENCE**

Intern

Smart Communications, Inc.

N. Bacalso, Cebu City

April to May, 2015

Beta Tester

“Rancho Ranch” by Moocho Brain

**SKILLS**

* Develop Programs using C, C++, Android and Java
* Web Design and Development using HTML, PHP and CCS
* Database Management
* Graphic editing (Adobe Photoshop, Paint Tool SAI)
* 3D Graphics and Animation (Blender, Maya)
* Animation (Adobe Flash)

**REFERENCES**

Available upon request

**DEANNE F. CHENG**

**PERSONAL INFORMATION**

**Name:** Deanne F. Cheng

**Date of Birth:** July 09, 1995

**Age:** 22

**Home Address:** Dipolog City

**Contact Details:** deannefcheng@su.edu.ph

0935 864 3012

**EDUCATION**

**College Education:** Silliman University

(2012 - 2016) Bachelors of Science in Information Technology

Hibbard Ave., Dumaguete City

**Secondary Education:** Zamboanga del Norte National High School

(2008 - 2012) Dumaguete City

**WORK EXPERIENCE**

Intern

Smart Communications, Inc.

N. Bacalso, Cebu City

April to May, 2015

Beta Tester

“Rancho Ranch” by Moocho Brain

**SKILLS**

* Develop Programs using C, C++, Android and Java
* Web Design and Development using HTML, PHP and CCS
* Database Management
* Graphic editing (Adobe Photoshop, Paint Tool SAI)
* 3D Graphics and Animation (Blender, Maya)
* Animation (Adobe Flash)

**REFERENCES**

Available upon request

**APRIL DIANNE B. GAHUMAN**

**PERSONAL INFORMATION**

**Name:** April Dianne B. Gahuman

**Date of Birth:** April 26, 1996

**Age:** 21

**Home Address:** Dapitan City

**Contact Details:** aprilbgahuman@su.edu.ph

0977 424 0694

**EDUCATION**

**College Education:** Silliman University

(2012 - 2016) Bachelors of Science in Information Technology

Hibbard Ave., Dumaguete City

**Secondary Education:** Rizal Memorial Institute of Dapitan City, Inc.

(2008 - 2012) Dapitan City

**WORK EXPERIENCE**

Intern

iCOMM International, Inc.

Skyrise 1 IT Park, Cebu City

April to May, 2015

**SKILLS**

* Develop Programs using C, C#, Android and Java
* Web Design and Development using HTML, AngularJS, Javascript and CCS
* Database Management
* Leadership and Human Relations Skills

**REFERENCES**

Available upon request