

Project No. 58

Project Title: Scanned handwritten document converter software

Presented by

1. Chanon Dithnim section C ID 59070503411
E-mail chanon.dith@mail.kmutt.ac.th

2. Natakorn Chanpetch section C ID 59070503414
E-mail natakorn.voodoo@mail.kmutt.ac.th

3. Pacharapol Dawmukda section D ID 59070503490
Email i.am.boat@hotmail.com

Advisor:

Dr. Unchalisa Taetragool

Date - 17 March 2020

“I have read and approved the content of this report”

Unchalis Tetrageol

.....

အဲဒါနဲ့ (၁၇) နေ့

(.....)



Scanned handwritten document converter software

Chanon Dithnim	section C ID 59070503411
Natakorn Chanpetch	section C ID 59070503414
Pacharapol Dawmukda	section D ID 59070503490

A Project Submitted in Partial Fulfillment of the Requirements for the Degree of
Bachelor of Engineering

Department of Computer Engineering, Faculty of Engineering

King's Mongkut University of Technology Thonburi

Academic Year 2019

Scanned handwritten document converter software

Chanon Dithnim	section C ID 59070503411
Natakorn Chanpetch	section C ID 59070503414
Pacharapol Dawmukda	section D ID 59070503490

A Project Submitted in Partial Fulfillment of the Requirements for the Degree of
Bachelor of Engineering

Department of Computer Engineering, Faculty of Engineering

King's Mongkut University of Technology Thonburi

Academic Year 2019

(Dr. Unchalisa Taetragool)

..... Advisor

Project Committee

..... Committee

..... Committee

Project Title: Scanned handwritten document converter software

Project Credit: 3 Credits

Project Participant: Mr. Chanon Dithnim
 Mr. Natakorn Chanpetch
 Mr. Pacharapol Dawmukda

Advisor: Dr. Unchalisa Taetragool

Degree of Study Bachelor's Degree

Department of Computer Engineering

Academic Year 2019

Abstract

Project Name Scanned handwritten document converter software

Developers Mr. Natakorn Chanpetch

Mr. Chanon Dithnim

Mr. Pacharapol Dawmukda

Project Advisor Name Dr. Unchalisa Taetragool

A Scanned handwritten document converter software is a handwriting recognition program. It can be installed and performed on a personal computer. A program has the ability to convert from handwriting to digital texts. The main purposes are to decrease the time of document management and to reduce human mistakes. The implementation of our program is divided into three main parts which are graphical user interface, deep learning model, and database. A graphical user interface part is composed of Tkinter which is a Python library. A convenient communication between users and the software is an intention of the GUI part. A Convolutional Neural Network model is performed by TensorFlow. It is being utilized to predict from a single handwritten character to digital text consistently from the first to the last character input. Lastly, Non-relational database system is used to store the converted data as a JSON form. Non-relational database system is a flexible storage to store the data because it no need to design any schema that convenient to read and write operation of NoSQL and it is faster than traditional database (relational database).

Contents

Chapter 1 Introduction

1.1 Keywords	1
1.2 Problem statement, Motivation and Potential benefits	1
1.3 Project Type(s)	1
1.4 Objective	2
1.5 Project scope	2
1.6 Proposed Methodology	3
1.7 Equipment and Tools required	4
1.8 Original Engineering Content	5
1.9 Tasks and Schedule	6-8
1.10 Table tasks deliverables for Term 1 & Term 2	8

Chapter 2 Background/ Related Theory and Related Literature

2.1 Background	9
2.2 Related Theory	10-19
2.2.1 Convolutional neural network	10-16
2.2.1.1 Adaptive Moment Estimation (Adam optimizer)	13
2.2.1.2 Rectified linear unit (ReLU function)	14
2.2.1.3 SoftMax Function	15
2.2.1.4 Categorical cross entropy (Loss function)	15
2.2.1.5 Gradient descent	16
2.2.2 NoSQL	17
2.2.3 Mathematical Morphology	17-19
2.3 Technologies & Development Tools	20
2.3.1 Tensorflow	20
2.3.2 Jupyter notebook	20
2.3.3 Google colab	20
2.3.4 Tkinter	20
2.3.5 MongoDB	20
2.4 Related Research & Literature Review	21-22

Chapter 3 Method and Design

3.1 Architecture Software	23-31
3.1.1 Architecture	23
3.1.2 Feature lists	24
3.1.3 Design framework	25
3.1.4 Use case diagram and Sequence diagram	26-28

3.1.5 Software limitation	28
3.1.6 Research category	29
3.1.7 Software Requirement list	30-31
3.2 Frontend	32-34
3.2.1 User interface screen layouts	32-34
3.2.1.1 Tutorial page	32
3.2.1.2 Inserting, converting, and preview a document page	33
3.2.1.3 Displaying scanned information page	34
3.3 Modeling	35-41
3.3.1 Data	35-41
3.3.1.1 Gathering data from internet	35-36
3.3.1.2 Imitated data	37-40
3.4 Backend	41
3.4.1 Database schema	41
3.5 Flow Program	42

Chapter 4 Implementation, Results, and Discussion

4.1 Frontend implementation	43-44
4.1.1 User interface	43-44
4.2 Modeling implementation	45-80
4.2.1 Data consolidation	45-50
4.2.2 Data preprocessing	51-71
4.2.2.1 Capturing data	51-55
4.2.2.2 Binarization	56
4.2.2.3 Thinning	56
4.2.2.4 Slant correction	57
4.2.2.5 Word segmentation	57-58
4.2.2.6 Character segmentation	58-71
4.2.3 Machine learning	72-80
4.2.4 Validation function	80-81
4.3 Database system	81

Chapter 5 Overall and Conclusion

Reference	84-85
------------------	-------

Chapter 1 Introduction

1.1 Keywords: Convolutional Neural Network (CNN), Non-relational database (NoSQL), Artificial Intelligence (AI), Machine learning (ML), Deep learning (DL), JSON form

1.2 Problem Statement, Motivation, and Potential Benefits

Nowadays, Thailand has lots of organizations that are related to documents, and the amount of documentation of each company has a trend to be increasing steadily. However, some companies still use traditional document management, which is using a staff to manage documents by typing information into a database after getting a document that contains handwriting from people. Sometimes, this solution leads to a mistake that can happen from the staff. Moreover, the mistake might occur from the different handwriting of a person, and sometimes the complexity of handwriting is a problem that decreases the efficient interpretation of staff, making document management of staff depends on staff experience. Furthermore, increasing of documents is one of the factors that it causes to slow working of staff, including a lack of experience of staff, making an organization makes a mistake and a delay in terms of document management.

From the problem, as we mentioned above, we provide a solution that can solve these problems, making the rise of precision of document management and also reduce a period of document management. This program is created for conversion from handwriting to digital texts, to support an organization and a staff who relates to document management, this program can transform a handwriting image to digital texts.

Therefore, this program is a good option to deal with the situation that many companies are facing. In addition, it does not only support them in terms of increasing a reputation but also making a rise in their document management processing efficiency.

1.3 Project Type(s): Potential commercial product and Addressing a document social or environmental problem.

1.4 Objective:

In this project, we aim to build and develop a program that can correctly and quickly convert a Thai handwriting document into a digital text file. We have adopted various computing engineering knowledge and tools to solve real-world problems. Our project, thus, focus on the 3 main objectives as follows:

- 4.1. To develop an efficient Thai handwriting converter software
- 4.2. To study various technologies and the effectiveness of different methods that can be used to develop the software
- 4.3. To improve the document management process by eliminating some manual steps and avoiding mistakes in the process that might occur from staff.

1.5 Project Scope

At present, most organizations always work with a large number of documents, they use a document to communicate data between departments and gather data from someone. The main work of the document is typing information on the document into the database by the employee. It is an important action that makes drawbacks to their organization because the employee might insert incorrect information into the storage system, typing data into the database by employee takes more time than usual, and their storage system does not respond for large data. The problem that we are facing is how we reduce data typing time into a database, how to decrease or eliminate the mistake of an employee from adding the wrong information into a database, and improving the performance of the database system. We would like to create software that can solve those problems. Moreover, the software can read the handwritten documents, then convert that information to digital format for storing in a flexible database system. For our project, we will start in August 2019 and we expect this project to be completed within May 2020. In addition, major milestones consist of five parts: consolidation data, creating a model, creating a database, and construction program. After finishing the project, the program can install on a personal computer and notebook and it needs a scanner for scanning the required document and insert the document into the program. Lastly, we expect our program, it can solve the problems and gain more benefits to the organization. These are the expected outcomes for the project.

1.6 Proposed Methodology

1.6.1 Literature and product exploration

For the first time, we explore existing research or article that associate with handwritten text recognition on the internet. Then, we study content and knowledge that stay on the research and articles.

1.6.2 Consolidation data

Explore and gather data from various sources of data such as the internet and collect data from real people. Then, create the main dataset that is combined from those sources and break it into three parts (testing set, training set, and validation set)

1.6.3 Data preparation

This step is preparation data to be suitable for working in the next step. The preparation is reshaping data, fixing data size, and charging aspect ratio of data.

1.6.4 Create a model

Design a deep learning model with a set of parameters. Then, use a given dataset from the second step to train and test the model and tune it by a new set of parameters.

1.6.5 Build a software

Design functional and interface for users easy to use our software. The software must be easy to development and maintenance by the way need to have data security.

1.6.6 Construct a database system

Design back-end connects with front-end and constructs non-relational database (JSON Form) with define key-value of the input parameter.

1.6.7 Create a complete software

To combine software with the model that we are trained completely. And then, we deliver to the user to testing our software.

1.6.8 Deploy software

The software is able to maintain and develop to make a higher efficiency for the model and software.

1.7 Equipment and tools required

The problems of this project, we have two main issues that are converting information on a document from handwritten format to digital format and creating a storage system for a large amount of data. For converting the document, we will use a combined approach between Convolutional Neural Network to train a model that can accurately classify each character from a word. For the storage system, we are going to use a non-relational database that can manipulate the increasing of a large number of data in the future. The kind of non-relational database that we will use documents store database. It is a system that uses a JSON form to store data, storing the format of JSON is similar to a document in the real situation. From the solution that we mentioned above, we will create a program that consolidates these abilities to release for installation on a personal computer.

The main technologies we will use,

For software on a personal computer, we create a front-end by using Tkinter that is a Python module for creating a GUI application, and back-end we will use a non-relational database (NoSQL) which is a document store database. The framework of the backend that we would like to use, which is MongoDB.

For modeling, the framework that we will use is Tensorflow that is developed from Google, appropriate for developing a model in terms of deep learning. The advantages of Tensorflow is open source by using language python. This framework can be work with CPU and GPU and handle all of the operating systems (Windows, macOS, and Linux).

1.8 Original Engineering Content

In this section, the content can be separated into three parts which are software, modeling, and storage system.

1.8.1 Software

Software, we are going to use python programming, it can handle our database system and modeling. There are lots of libraries that support user interface simulation for users.

1.8.2 Modeling

Utilizing CNN in the field of feature extraction to find the important features of the image, then the output of CNN is a character that the model predicted correctly. Moreover, CNN is suitable for classifying the characters through training and testing sets of data.

1.8.3 Database system

For the storage system part, we are going to use a non-relational database for collecting all data of our program. Moreover, a type of non-relational database that we will use is a document store database in JSON format, because this kind of system can deal with a huge amount of data that has been increasing. By the way, the JSON form has properties that are appropriate with documents in real situations, because it can store data as is without any transformation or schema. So, this database system is a good option to solve the problems that we mentioned in the second part.

1.9 Task and Schedule

Tasks	Aug			Sep			Oct			Nov			Dec			Jan			Feb			Mar			Apr			May		
1. Background research																														
2. Consolidation data																														
3. Data preprocessing																														
4. Create a model																														
5. Build a software																														
6. Construct a database system																														
7. Create a completely software																														
8. Deployment software																														
9. Writing and Editing report																														
10. Creating a presentation																														

Figure 1.1: Schedule of project

1) Background research

2) Data consolidation

- Create an imitated dataset (handwritten document from real people)
- Gather data on the internet
- Combine both datasets as the main dataset
- Divide the main dataset into three parts: training data set, validation data set, and test data set (main data set)

3) Create a model

- Data preprocessing
 - Capturing data
 - Binarization
 - Thinning
 - Skew correction
 - Slant correction
 - Word segmentation
 - Character segmentation
- Create the model
 - Build a Convolutional Neural Network model
- Training, testing and tuning model
 - Train the model with a training dataset
 - Test the model with the validation dataset
 - Test the model with a testing dataset
 - Tuning parameter

4) Build a software

- Design system
- Construct a software
- Testing software (just only software)

5) Construct a database system

- Design system
- Build a database system
- Testing database system

6) Create a complete software

- Compound three components together
- Testing the software

7) Deployment software**8) Writing and editing report****9) Creating a presentation****1.10 Deliverables**

Tasks
First semester
1.Preprocessing model
2.Design database
3.Design program
4.Deep learning models prototype (CNN) without tuning parameter
Second semester
1.Deep learning model (CNN) (include tuning parameters)
2.Database system
3.User interfaces program (GUI)
4.Compound three components

Figure 1.2: Deliverable of first semester and second semester

Chapter 2 Background, Theory and Related Research

2.1 Background

This project has a background from a development team that we found problems of staff in an organization. The problem is using staff to type information into a database. So, this affects staff about a work process, wasting time, and sometimes might have some mistakes from a human error. Staff needs to re-check again, and maybe there are some mistakes from the staff again. Therefore, our development team wants to provide software that is able to read Thai language document information from human handwriting. The software converts information into character digital text form, which documentation comes from scanned and transferred to a system of our software. The process of converting through a deep learning process. Deep learning is a subset field of machine learning that is handled for high-level features work through the network such as character recognition, computer vision, etc. After the conversion is finished, the software will transfer digital data to store into a database which is called NoSQL or Non-relational database. It is high-level for data management and also supports various data. However, NoSQL is not too complicated to design, and it is different from SQL. SQL needs to design relations and stored in rows and columns.

From our development team mentioned before, we want to reduce the process and time of a staff that needs to type information into a database. We want to increase efficiency by using a program to read handwriting from documents. Moreover, we want to improve handwriting recognition and a database to be more efficient and support a growing up of information in the future.

2.2 Related Theory

2.2.1 Convolutional Neural Network (CNN)

CNN is a class of Deep Learning for an interpretation that an input image handwriting belongs to which pattern. CNN can be separated into three main layers which are the convolution layer, pooling layer, fully connected layer. So, each of the layers will have various roles as you can see by the pictures below.

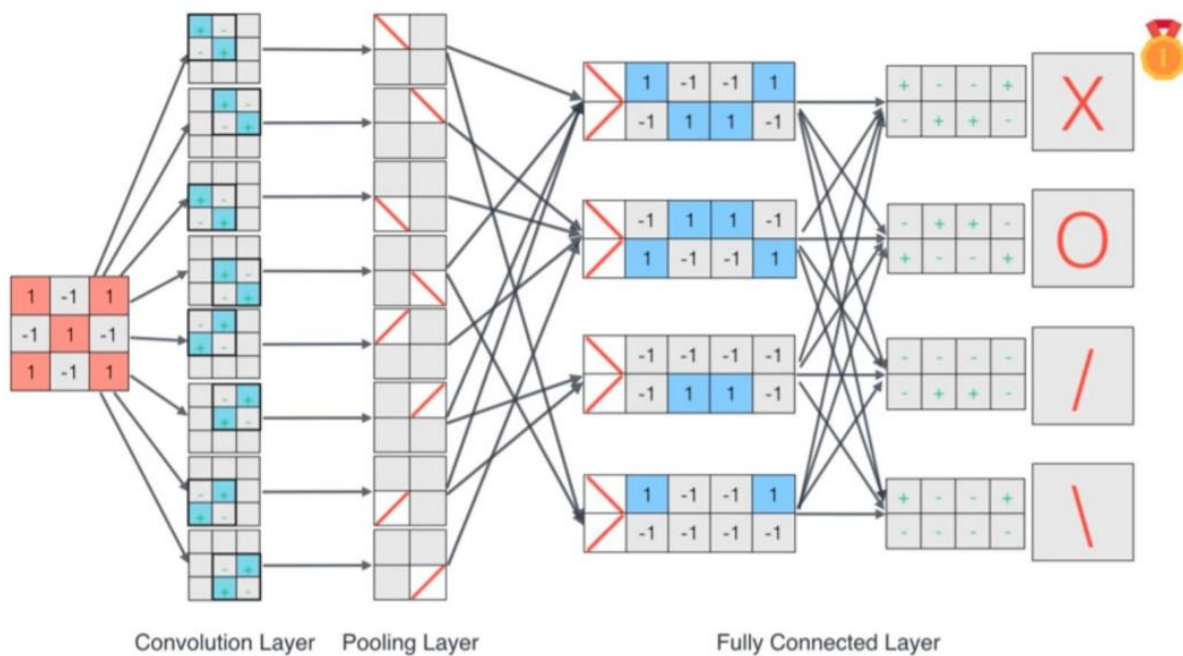


Figure 2.1: Process of CNN This is Convolutional Neural Network Architecture

In the Convolution layer, the filter is created for putting on an image to search for information on an input image. Then, the pooling layer will interpret information to minimize to the smaller metric size. The fully connected will get the input from the pooling layer to classify which pattern it belongs to.

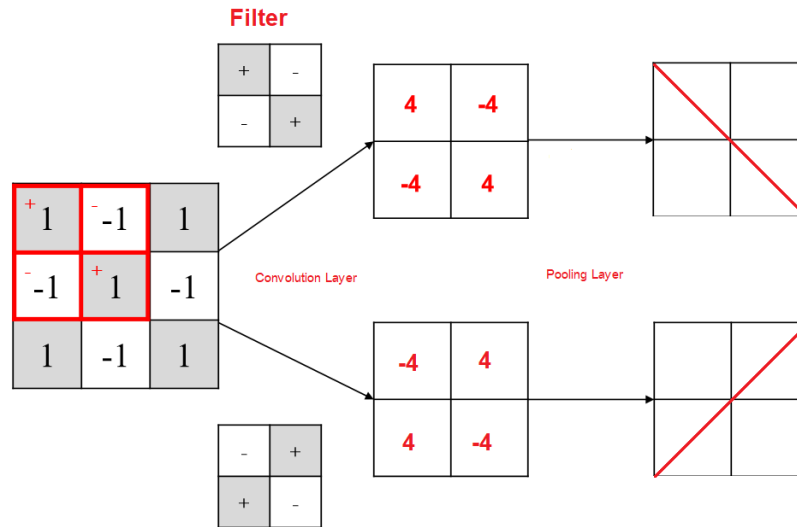


Figure 2.2: Convolution and Pooling Layer

Convolution layer is a layer that creates a filter or called the kernel to find a prominent point on the input image by filter decide input for dot product values between input and weight that filter will find all foreground of the image and continue until it completes every pixel of the image. So, when the filter is put on the input image each time there are summarizations of values on the filters. Moreover, it Will have an "activation function" added to help make calculations even easier. Which the producer chooses to use the "Rectifier Linear Unit (ReLU)" which will adjust the results Calculations less than 0 are all 0, but if the result is greater than 0, the value will not be adjusted.

Then the pooling layer will interpret the summarized values then convert into patterns that refer to the filter. The pooling layer decreases the size of input by selecting only the highest value (follow the size of the pooling layer).

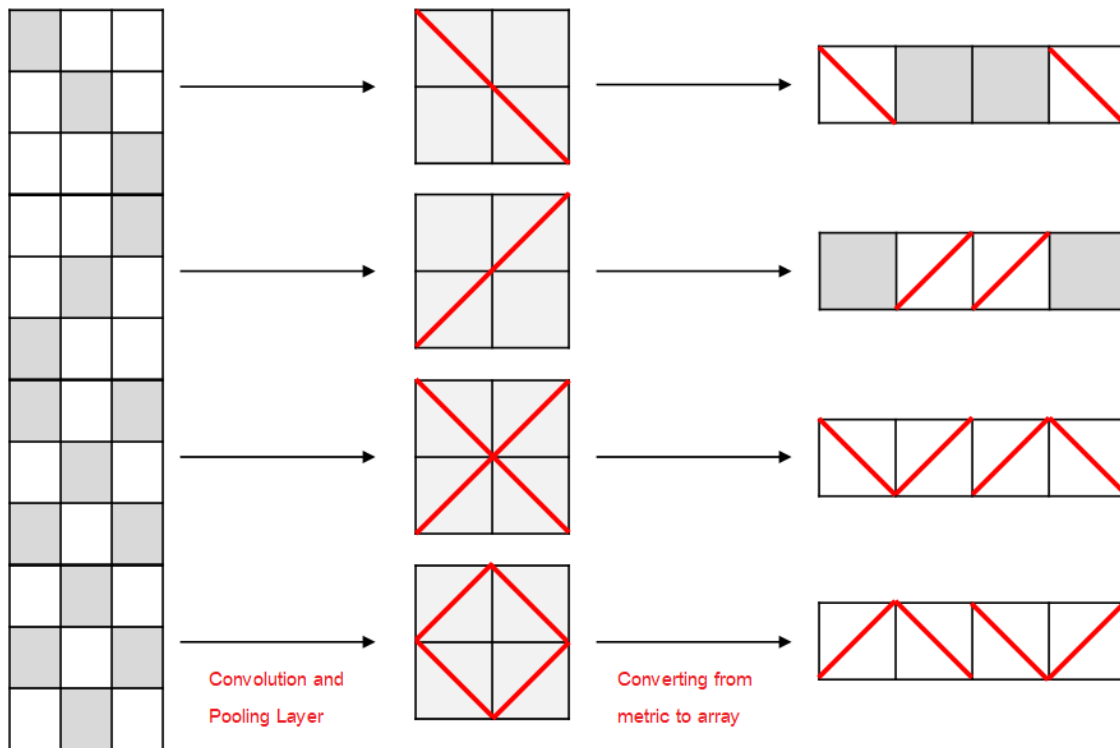


Figure 2.3: Converting from a metric to an array that it is understandable for a computer

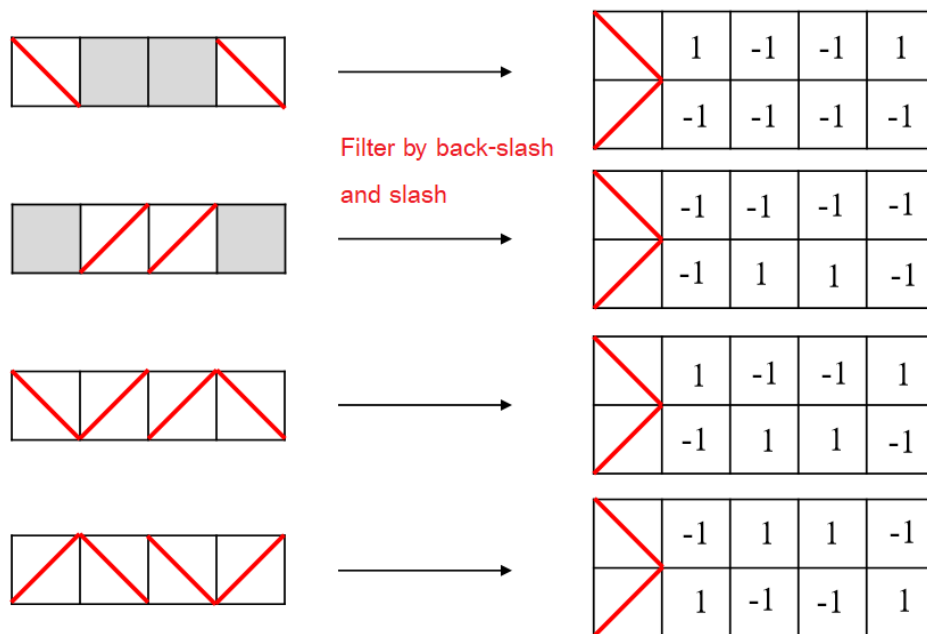


Figure 2.4: The arrays will be filtered by a backslash and slash again to be the final pattern

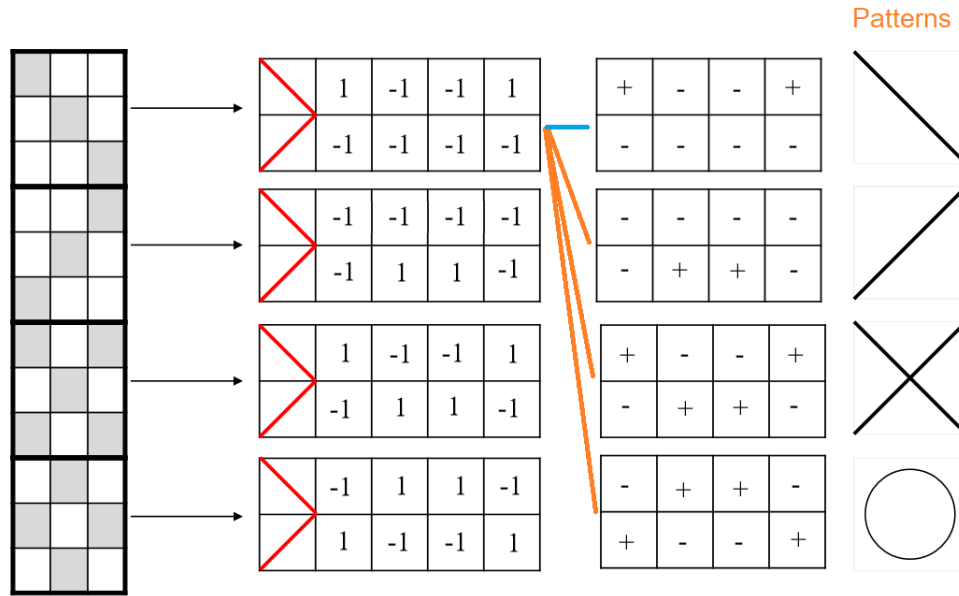


Figure 2.5: Showing the pattern checking of fully connected layer

After that, the fully connected layer will compare the inputs to existing patterns to check that which pattern is matched with the input. By the way, the blue one stands for the matched pattern and the oranges are compared patterns. For the other arrays will do likewise the illustration array. The Fully Connected Layer in this layer, the "Flatten" values sent from the previous layer are in the "single dimension array" arranged by "activation function" which comes in to help "SoftMax". To compare the maximum value that the input received from the beginning should be suitable with which class to predict as output that the received handwriting should be the correct character.

2.2.1.1 Adaptive Moment Estimation (Adam optimizer)

Adam is an adaptive learning rate optimization algorithm that has been designed specifically for deep neural networks. So, we choose Adam because it has the ability to train deep learning by having a fast converge rate and give a precise result.

2.2.1.2 Rectified Linear Unit (ReLU function)

ReLU is the most used activation function for the convolutional neural network

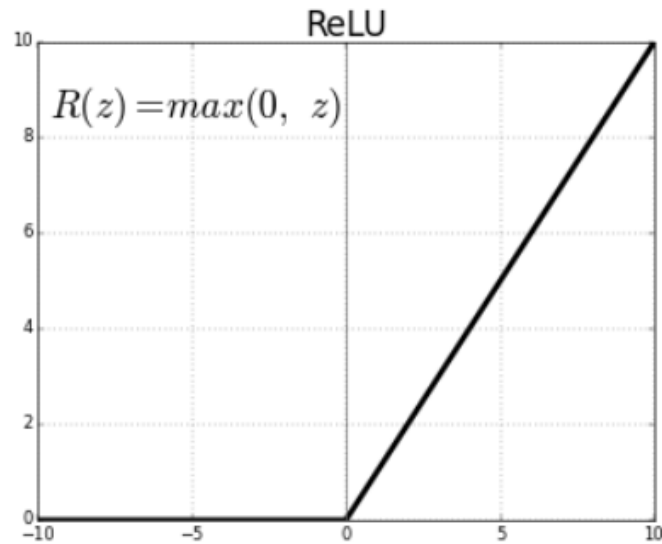


Figure 2.6: ReLU Function Graph

As you can see from the figure, ReLU is half rectified (from bottom). $R(z)$ is zero when z is less than zero and $R(z)$ is equal to z when z is above or equal to zero. So, we decide to use ReLU on each convolution layer because the value adaptation of ReLU has an impact on a model that it can be trained fastly.

$$f(x) = \max(0, x) = \begin{cases} 0 & \text{for } x \leq 0 \\ x & \text{for } x > 0 \end{cases}$$

Figure 2.7: ReLU Equation

2.2.1.3 SoftMax Function

SoftMax Function is particularly useful in a multiclass classification when the output has to be one and only one class, and it is mainly used to normalize neural network output to fit between zero and one. Moreover, it is used to represent the certainty probability in the network output. So, the normalization is calculated by dividing the exp of the examined output by the summation of the exp value of each possible output as you can see from the equation below.

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

Figure 2.8: SoftMax Equation

2.2.1.4 Categorical Cross entropy (loss function)

It is also called as SoftMax Loss because it is a Softmax Activation plus a Cross-Entropy Loss. If this loss is used we will train CNN to output a probability over the C classes for each image. It is used for multi-class classification

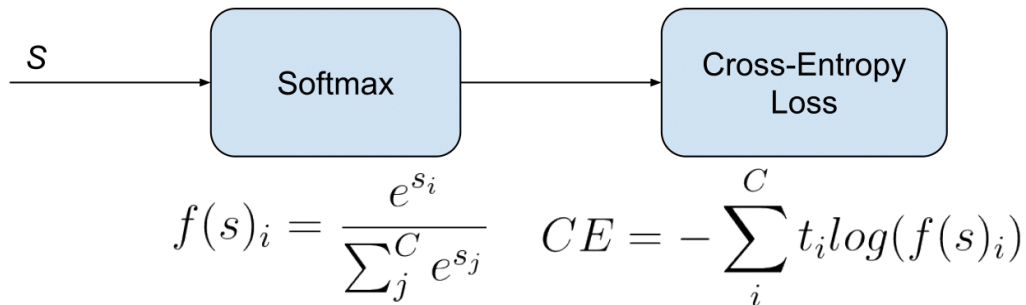


Figure 2.9: SoftMax Equation

2.2.1.5 Gradient Descent

Gradient descent is a learning algorithm or weight adjustment algorithm in deep learning that used to learn some error from a result and improve it for more efficiency. Moreover, this algorithm is represented with a mathematical equation to calculate a value that brings it to evaluate for a new value. The purpose of Gradient descent is to look for the lowest error as much as possible by adjusting some important values within a deep learning technique.

$$w_{ij} := w_{ij} - \alpha * \frac{\partial Cost}{\partial w_{ij}}$$

↑new weight ↑old weight ↑learning rate ↑gradient of cost function

Figure 2.10: The mathematical equation of this algorithm

Figure 2.10 above is a mathematical equation of this algorithm. A variable on the left is the new weight that is a result of this equation. On the right side is a gradient of the cost function, it is the error value that calculates from a result of deep learning. For the last one, it is a constant value that controls the rate of learning to find the lowest error.

As we mentioned above, searching for the lowest value of error is a major role of a Gradient descent algorithm. Figure 2.11 is a graph in three dimensions that represent the error of deep learning. Its work is gradually down to the bottom of the graph and repeatedly found the point of lowest error.

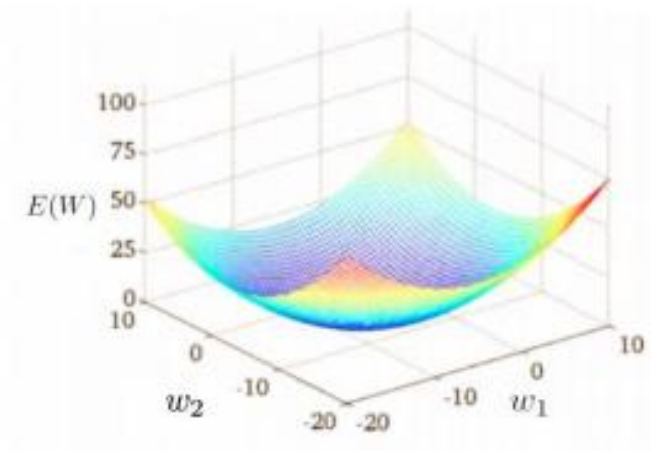


Figure 2.11: Error value between w1 and w2

2.2.2 NoSQL

NoSQL, stands for Not only SQL, is a place to store data, that can accommodate several kinds of data. NoSQL does not require any designing for schema, it provides more speech and handles a large amount of data that has been increasing.

2.2.3 Mathematical Morphology

Mathematical Morphology is a technique in digital image processing. It is a transformation of shape or structure of an object in pixels of an image by using eight basic operations from set theory as shown in Figure 2.12 and structuring elements. The morphology is generally used to transform the region for binary images (grayscale) and can be widely performed in other operations of digital image processing. For the structuring element, a 3x3 block, called a kernel, includes a set of values of the matrix to convert the structure of the object. The kernel is shifted across the image from left to right and top to bottom. While the kernel is shifting over the image, it will compute the value of a matrix between it and a block image with some basic operation in set theory.

Symbol	Name
$\in, \emptyset \text{ and } \rightarrow$	Standard set notation
$A \subset B$	Included
A^c	Complement
\hat{A}	Transposition
$A \cup B$	Union
$A \cap B$	Intersection
$A - B$	Difference

Figure 2.12: Eight operations set theory

In morphology, there are two major operations, namely dilation and erosion, and two minor operations, namely opening and closing, for the analysis. First, dilation is an enlargement or expanding the boundaries of the region of foreground pixels and filling holes in the object. Figures 2.13 and 2.14 show an equation of dilation and an example of the dilation, respectively. It uses an intersection operation between the grey-scale image and the structure element.

$$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \emptyset\}.$$

Figure 2.13: The equation of dilation

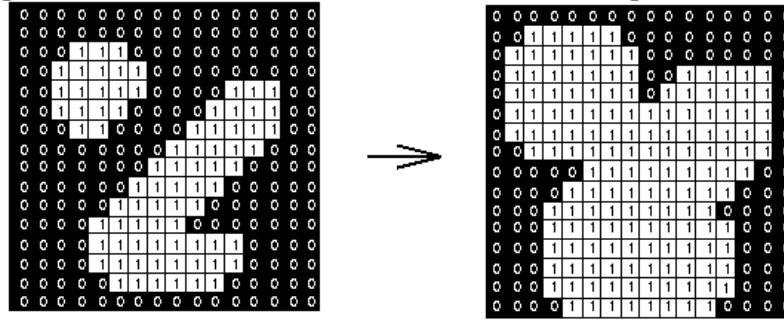


Figure 2.14: An example of dilation

Erosion is used to reduce the boundaries of regions of foreground pixels and increase holes in the object. For the equation of erosion as shown in Figure 2.15, it uses an included operation between the image and kernel. Figure 2.16 illustrates an example of the erosion operation.

$$\mathbf{A} \ominus \mathbf{B} = \{z \mid (\mathbf{B})_z \subseteq \mathbf{A}\}.$$

Figure 2.15: The equation of erosion

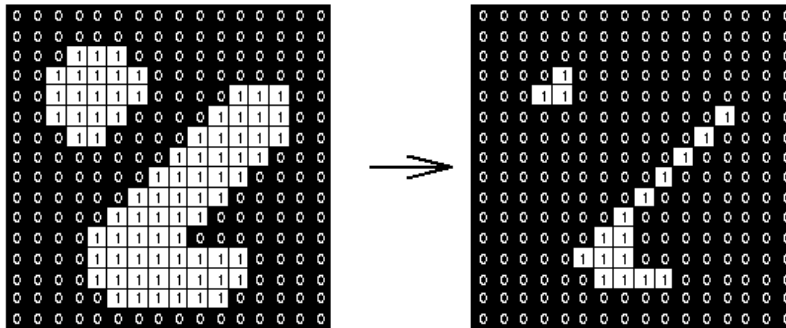


Figure 2.16: An example of erosion

For two minor operations, they are a combination in terms of dilation and erosion methods with basic operations in set theory. First, an opening operation is implemented as erosion and then followed by dilation with the same kernel. Figure 2.17 displays an equation of opening and showing a result of opening. This operation is created to eliminate or segment all pixels in the image that are too small by using a structuring element. Second, a closing operation is the opposite of opening. It is constructed by dilation and followed by erosion with the same structuring element. The closing operation is used to connect objects that are quite close to each other and fills up small holes. A result of closing and its equation are shown in Figure 2.18.



Figure 2.17: The result and equation of opening operation



Figure 2.18: The result and equation of closing operation

2.3 Technologies & Development Tools

2.3.1 TensorFlow

TensorFlow is a framework that was released and developed by Google. The purpose of this framework is to create for programmers who work with numerical computation, large-scale ML or deep learning to build efficient models for prediction. The advantages of TensorFlow is an open-source framework using python language that supports python versions 2 and 3. TensorFlow can be handled both CPU and GPU and support all of the operating systems (Windows, macOS, and Linux).

2.3.2 Jupyter Notebook

Jupyter Notebook is a framework that is suitable for the Python language and supports Python versions 2 and 3. However, this framework is appropriate for machine learning (ML) or deep learning (DL). This framework has many libraries for computing the data such as Numpy, Scikit-Learn, Matplotlib, Tensorflow (TF) or OpenCV that run through the Anaconda Navigator for support all of the operating systems (Windows, macOS, and Linux).

2.3.3 Tkinter

Tkinter is a Python module for creating a GUI application. It has been developed from Tk GUI Toolkit. it can be used in many frameworks such as PyCharm, Sublime Text, and Visual Studio.

2.3.3 Google Colab

Google Colab is a framework that uses cloud service and supports free GPU to improve Python language. Colab is suitable for developing deep learning applications using libraries such as Keras, TensorFlow, PyTorch and OpenCV.

2.3.5 MongoDB

MongoDB is a document database, which means the NoSQL database. MongoDB stores data like JSON-form by record key-value in the document. This kind of database does not require a fixed schema, it has quite more flexibility and provides a simple setup on the computer. MongoDB is a good option to use for data storage.

2.4 Related Research & Literature Review

The name of the previous research that we studied is Handwritten Text Recognition using deep learning. This research is written by three teachers from Stanford University. Background of research is many people still use traditional ways to take their notes when they learn some lessons in the class or have a meeting such as a pen, pencil, and paper. Moreover, most notes are the writer's handwriting. These conventional ways are there are disadvantages. For example, it is difficult to store on electronic devices such as a personal computer, tablet, and smartphone. And another example, searching, and analysis is hard to do because you have to read all the pages if you want to find the desired information. It is hard work and time-consuming. To solve this problem, they will construct a tool that can read handwritten text from their notes, then convert that information into digital format for simple storing, searching, and analysis. For the strategy to reach the goal of the research, they use two major approaches. First, classifying words directly by using a deep learning technique that calls Convolutional Neural Network and other way classifying words indirectly by using a Long short-term memory in the Recurrent Neural Network field.

Handwritten Text Recognition using deep learning research consists of seven parts: abstract, introduction, machine learning, data, methods, experiment, and future work. From reading this article, we found a significant three issues for discussion. Firstly, it is an advantage of this paper. They quite clearly explain almost every part, it is strongly pointing that make us rapidly understand all their process. Another good point, they always identify problems that occur during their explanation in every part and give a solution to those problems. For example, in part of data, it will take extremely long in their experiment if they use all data to train their model of deep learning. They suggest us to select 50 words with at least 20 occurrences for fixing this issue. The last one is my favorite part, which is a comparison of all their experiments to show all the accuracy of each model of deep learning. For the second issue, they are drawbacks. They not unambiguously explain preprocessing in part of data, it is an important part to prepare data for deep learning models. Moreover, some parameters for tuning deep learning mode, it does not show detail in their experiment. Lastly, a comparison between our project and its research. We use deep learning techniques that are the same with their strategy, the deep learning techniques are Convolutional and Recurrent Neural Network. For a different point, we will create imitated data as an input dataset for training our deep learning model. In their input data, they use the only dataset on the internet, it really not updates and not suitable for our model.

From what we discussed above, this project has a target group to support them. The real-world stakeholder is all companies that work with the document because they have to use many kinds of documents to store data from real people or contract to work within the organization. In addition, an employee that has a main role to manage and handle the document is the one who receives the most benefit from this project.

This project is the first program that converts handwritten text to digit format. There are many products in the marketplace, they have the main task that is similar to our project. We will indicate one product, it is almost the same as our product. it is a Pen to Print, that is an application on android and macOS device. This application has a few simple steps for working. However, this application has many things that are different from our project. The first thing is input data, our project uses a scanned document that comes from the printer. The printer has constant accuracy because it uses a high performance of the camera, but the camera on smartphones has many internal and external factors to control for a high quality of input data such as CPU, quality of camera, and environment. Lastly, the application does not suitable for large scales such as organization, hospital or government agency. It does not have its own database that has to use other data stores to keep data such as google cloud, device's memory (SD card), or personal computer. But our project can respond to a big scale, it has its own database system to deal with a huge amount of data that has been increasing in the future.

Chapter 3 Design and Methodology

3.1 Architecture software

3.1.1 Architecture

Our software works along with a scanner to scan a handwriting document for getting an image file. After that, the program will display the preview of the document and directory. When the user clicks a button to convert the handwriting document. Then, our program will transfer data to a pre-processing model and an AI model. The AI model uses CNN to convert the handwriting to digital text and encapsulate them into a JSON file. After that, the program will show the results to the user. Once the user clicks a button to store the results into a database, the program will automatically store the data into a non-relational database system.

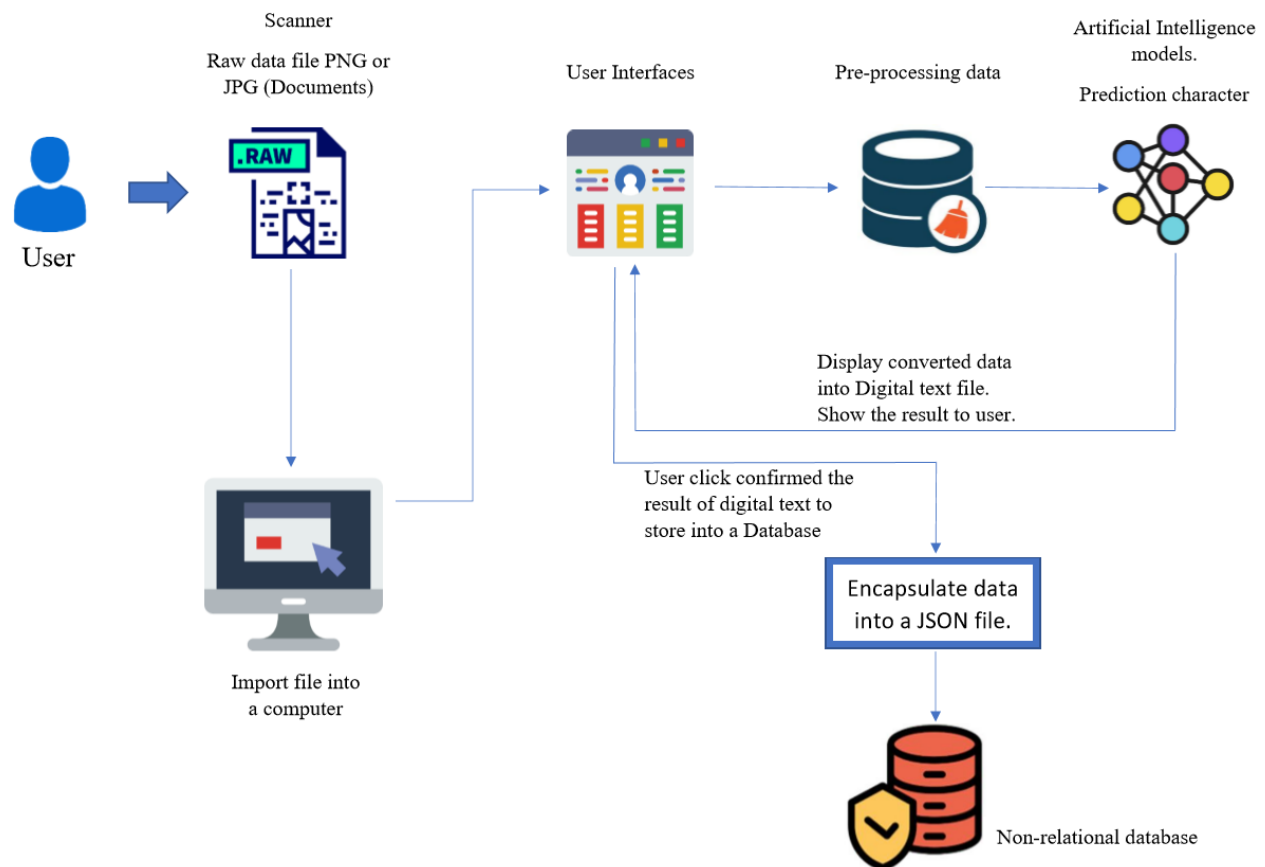


Figure 3.1: Architecture system of software

3.1.2 Feature lists

3.1.2.1 Import an image

For this feature, a user can select desired data from internal memory on the computer in order to make the form conversion.

3.1.2.2 Filter data steps (pre-processing)

After importing data, this step will change or transform the data format and improve it to be an appropriate form for the converting process (deep learning). The pre-processing step consists of seven parts, including capturing data, binarization, skew and slant correction, thinning, word segmentation and character segmentation.

3.1.2.3 Convert from handwriting to a digital text

This is the main part of the program. The data from the previous step is fetched into the deep learning model (Convolutional neural network). The model will extract a set of features of data and make a prediction. So, it will return a result in the digital character form.

3.1.2.4 Edit information

In this feature, it is created to let users check all information after the converting step and before inserting them into the storage system. If there are any incorrect data that occur from the software or human error, the user can edit the information by themselves. Moreover, this step will encapsulate the digital characters into a JSON file.

3.1.2.5 Insert into a database

The database is the last part. After converting the form, the data in the digital format or JSON file, it will be stored automatically into the NoSQL database system.

3.1.3 Design framework

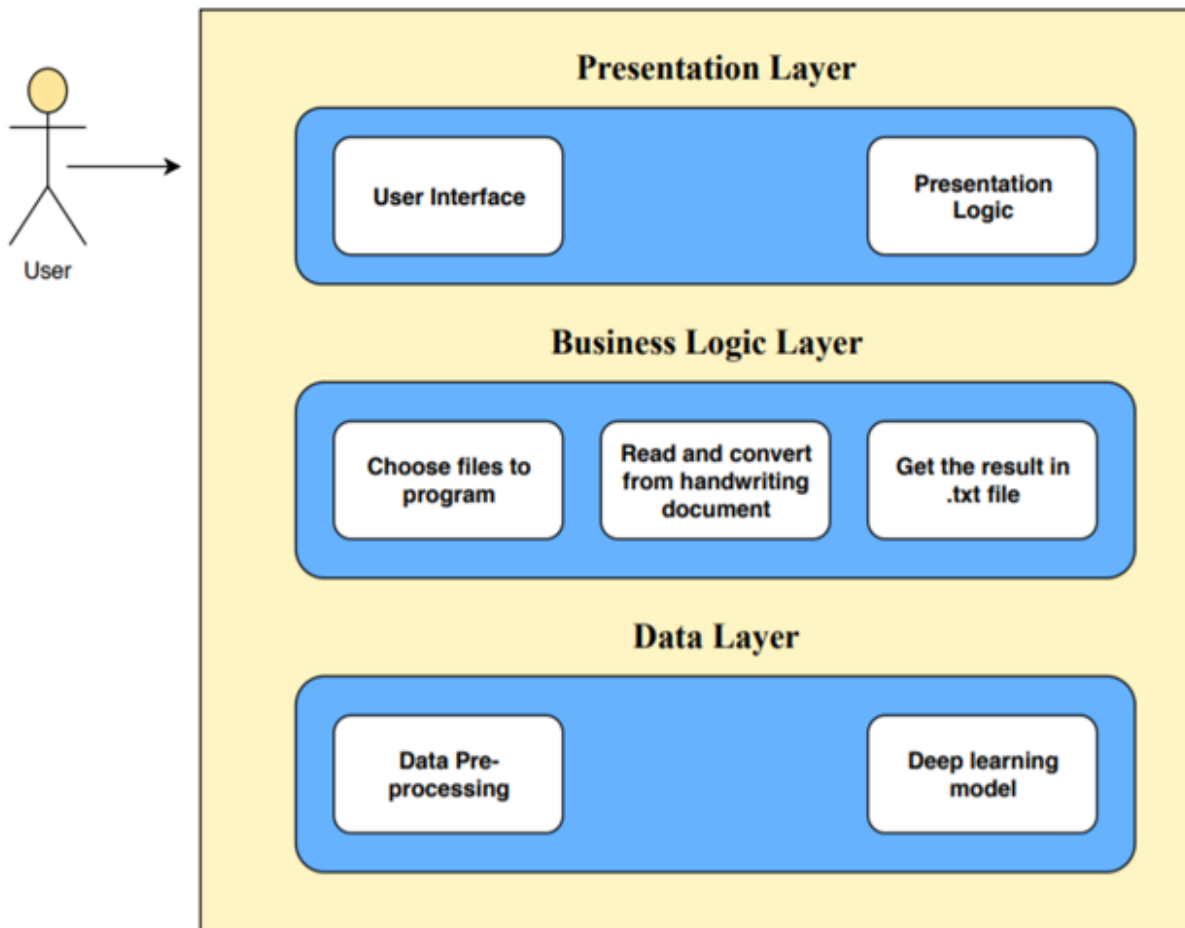


Figure 3.2: The framework of the system

3.1.3.1 Presentation Layer

This component contains communication between users and our program.

3.1.3.2 Business Logic Layer

This component links between the Presentation Layer and the Data Layer. This layer converts the handwriting document from the user-imported file into our program.

3.1.3.3 Data Layer

The last component is to prepare the data for use in our program. The model will learn characters from the pre-processed data more efficiently for prediction.

3.1.4 Use case diagram and Sequence diagram

3.1.4.1 Use case diagram

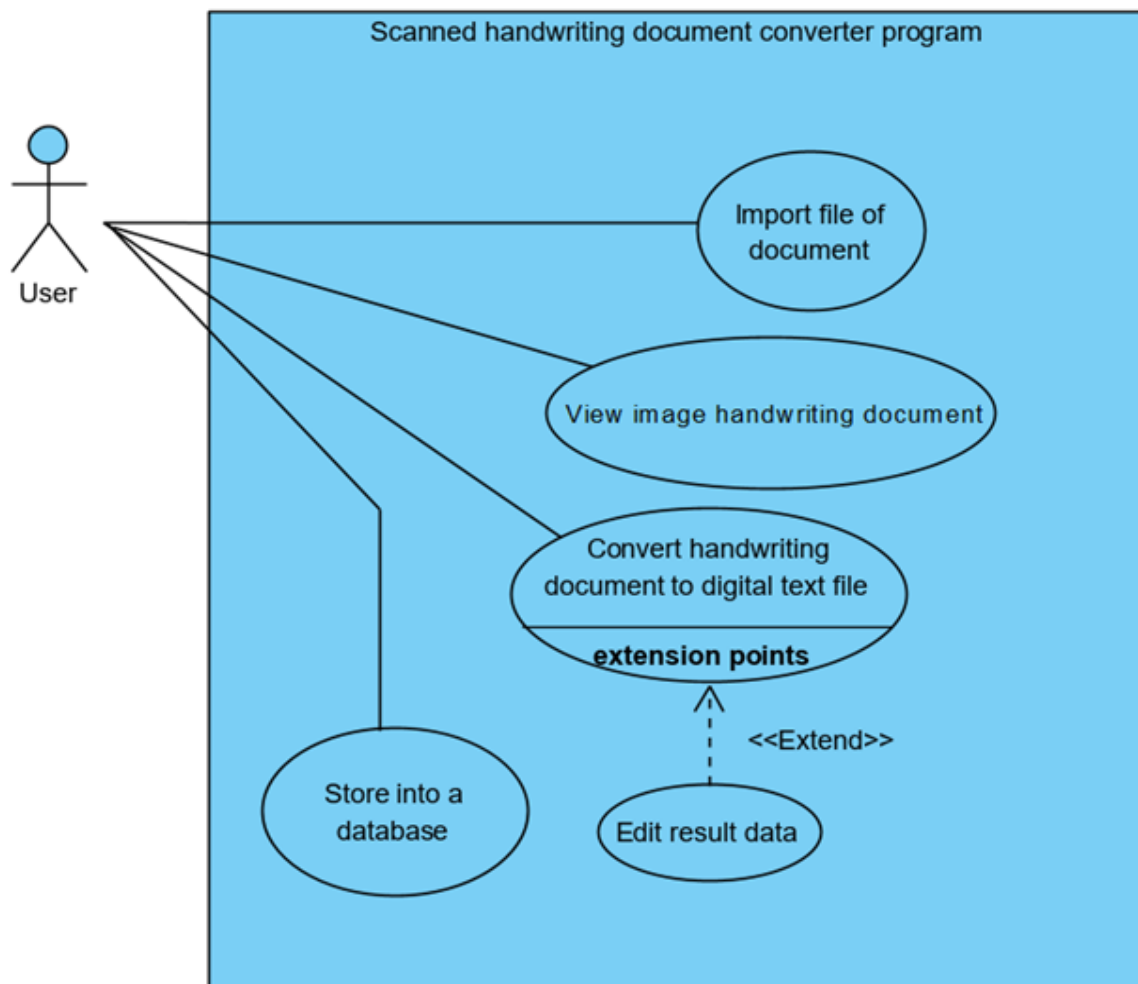


Figure 3.3: A use case diagram for our software

- 1) Import file PNG or JPG document, to display users which directory that users will import to document to our program.
- 2) View image handwriting, to display users in which the user can see the preview of an image that user input into a program.
- 3) Convert handwriting document, to display by users be able to click a button to convert handwriting document so, our program will automatically convert and show the result to a user. After that, the user can edit the result if the result is not correct or edit something that is missing or adding more information.
- 4) Store into a database, after the user finished editing and checking the result, so the user is able to click a save button to store data into a database system.

3.1.4.2 Sequence diagram

Scenario 1: User import file to our program

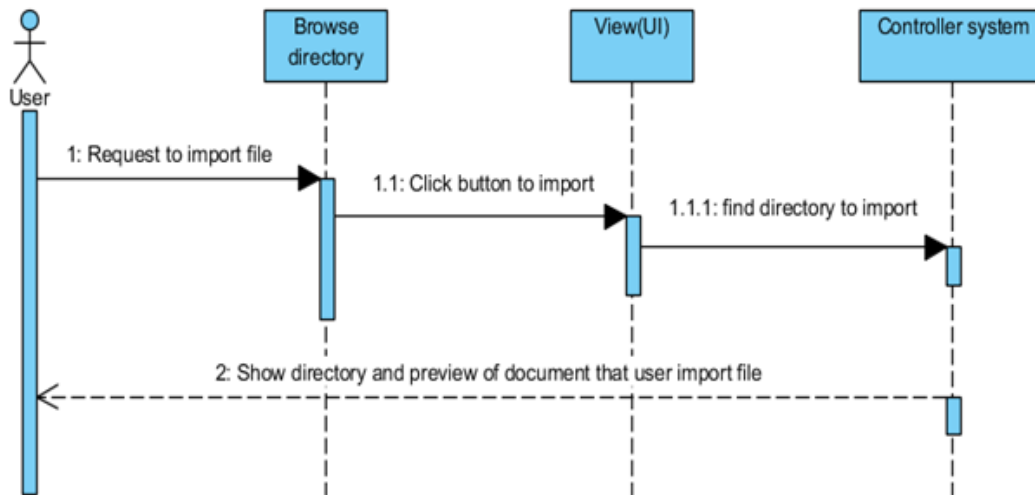


Figure 3.4: Import file to program

The user would like to import a file to our program so the program already has a button to click import file and the pop-up will show the directory that the user would like to import a handwritten document file to our program. After that, when the user finishes importing the file, the program will show a preview of the handwriting document and directory of this file.

Scenario 2: User convert handwriting document to a digital text file

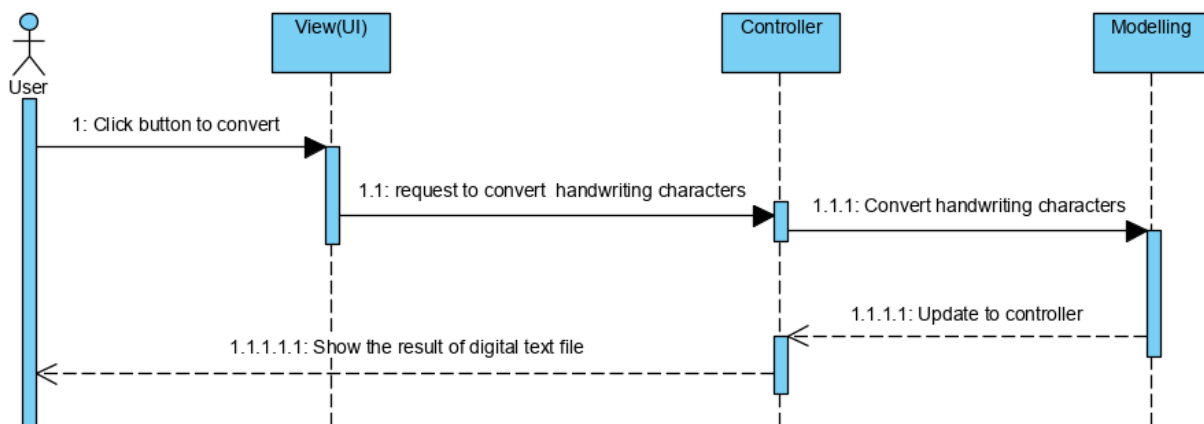


Figure 3.5: Convert handwriting document to a digital text file

When the user would like to convert the handwriting document, he/she can click a button to convert a handwriting document. Then, the program will automatically convert it through our model. After the program finishes converting, the program will show results to the user. If the results are not correct, the user can edit the incorrect information.

Scenario 3: User store into a database

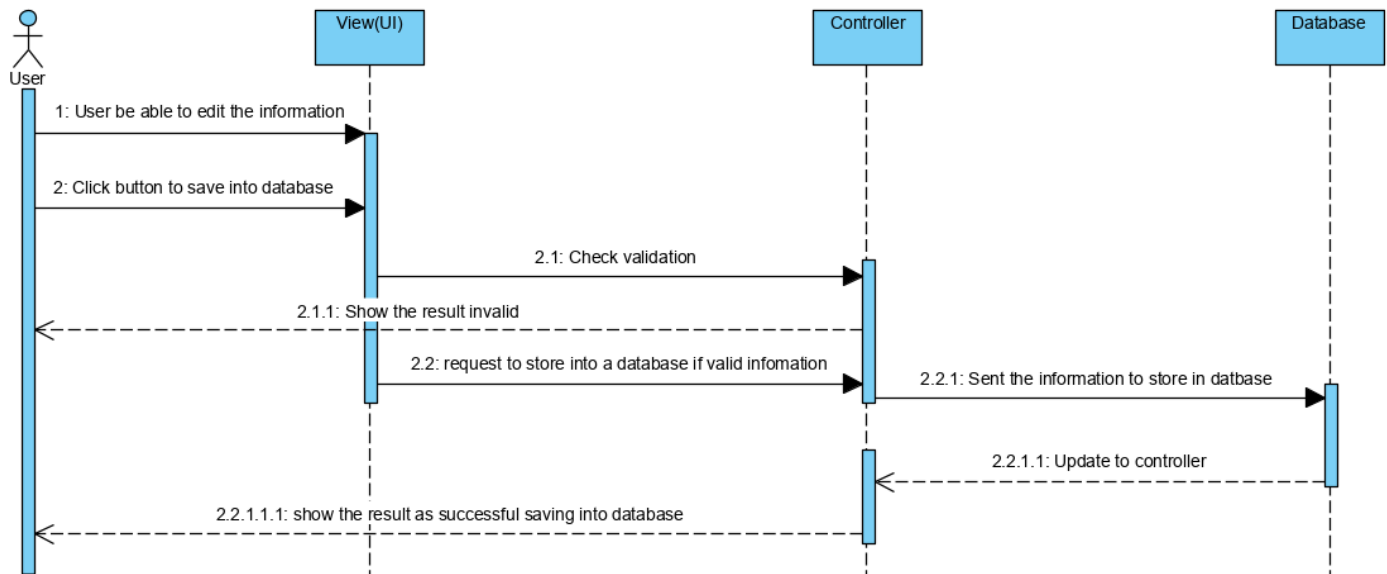


Figure 3.6: Store into a database

The user can store information into a database by clicking a save button to store a result. Then, the program will check validate information of the user before transferring information to a non-relational database. After that, the program will show the pop-up message whether it is done successfully or unsuccessfully to the user.

3.1.5 Software limitation

- 1) The software cannot convert other languages other than Thai.
- 2) The program can only support files PNG and JPG.
- 3) The program cannot scan more than one document.
- 4) The program can only support the operating system on Windows
- 5) The program can only be used on a personal computer and notebook.
- 6) Users are not allowed to use the program through an application on electronic devices such as smartphones and tablets.
- 7) The software cannot read bad (unreadable) handwriting such as the doctor handwritten.
- 8) A document or form is always scanned from the scanner before importing it into the program.

3.1.6 Research category

The software has required a list of experimental plans for evaluating the result of our research. From the research that we have studied, we can identify four objectives to be a goal of our project. However, we decided to use those objectives as a list of experimental plans for estimation of the result. Moreover, we created a few experimental plans in addition to four objectives for evaluating the outcome of previous research.

3.1.6.1 To develop a Thai handwriting system to be more efficient.

Indicator: An accuracy of prediction is not less than 70 percent

Measurement method: Use a criterion or some variables in mathematics for measuring accuracy.

3.1.6.2 To develop the Thai handwriting system to be more efficient from previous research.

Indicator: An accuracy of prediction is better than previous research

Measurement method: Compare precision between our model and their model from previous research.

3.1.6.3 To help the work process documents more quickly especially reduce the process of typing documents in the database.

Indicator: Working in the document process per piece takes less time around 20 percent.

Measurement method: Compare the average time of the traditional way and new way.

3.1.6.4 To reduce mistakes in a document that might occur from staff (human error).

Indicator: Number of requests for changing the information on the document from the user are reduced by 40 percent.

Measurement method: Compare a number of requests of the traditional way and new way.

3.1.6.5 To study the various technologies and different methods to develop the Thai handwriting system.

Indicator: The algorithm we use for our program is different from the previous research.

Measurement method: Compare accuracy between our algorithm and research that we are expecting more accuracy than research.

3.1.7 Software Requirement list



Figure 3.7: Google Colab

Google Colab is a framework run through the server. The purpose of this framework is to use cloud service and supports free GPU to improve Python language. Our program requires Google Colab for computing deep learning that contains Keras libraries, TensorFlow or OpenCV to build efficient models for Thai handwriting. Moreover, Google Colab is able to support large scale of deep learning to compute the amount of dataset.



Figure 3.8: Tkinter is a library in python

Tkinter is a Python module for creating a GUI application. it can be used in many frameworks to perform a GUI. Our program requires Tkinter to execute as the execution of a graphic user interface to receive input from the user, make a conversion from Thai handwriting to digital texts, then perform a storing system to store data into a database system (MongoDB)



Figure 3.9: TensorFlow is a library in python

TensorFlow is a framework that was released and developed by Google. The purpose of this framework is to create for programmers who work with numerical computation, large-scale ML or deep learning to build efficient models for prediction. Our program requires TensorFlow to perform with the dataset that we are prepared for the work process to train and test our model prediction. Moreover, TensorFlow can support handwriting recognition which is a large-scale deep neural network that handled by using Python language.



Figure 3.10: Jupyter Notebook

Jupyter Notebook is a framework run through Anaconda Navigator. The purpose of this framework is to create a program by using Python language. Our program requires Jupyter Notebook for computing deep learning and GUI that computes with GPU that has many libraries such as Numpy, Matplot, OpenCV, TensorFlow (TF) or Tkinter for handling the Thai handwriting document converter program.



Figure 3.11: MongoDB

Our program is an offline software, it does not require the internet to work for running the program. We used a MongoDB, which is a non-relational database or NoSQL. We selected it because MongoDB has three important abilities. The first one is management with a large amount of information that has been growing all the time. Next, it is suitable for unstructured data such as images or videos. Lastly, MongoDB is designed to scale out formats that are dividing a task into small parts, and then execute all parts at the same time. These three abilities make MongoDB work faster than traditional databases or SQL.

3.2 Frontend

3.2.1 User interface screen layouts

3.2.1.1 Tutorial page

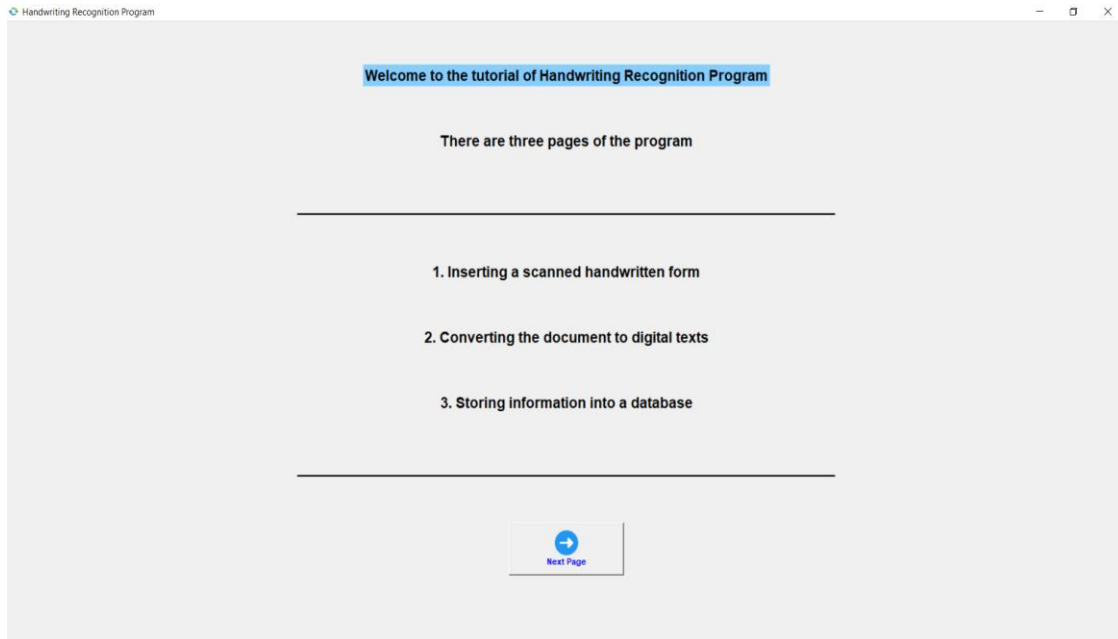


Figure 3.12: Tutorial page

The first page of software is an instruction page. This screen will explain all the steps that users have to know in order to use the program. The tutorial consists of three steps. The first step is to insert a scanned handwritten document file. Next, the software will convert the imported document. Then, the last step is to check information and store them into the database.

3.2.1.2 Inserting, converting, and preview a document page

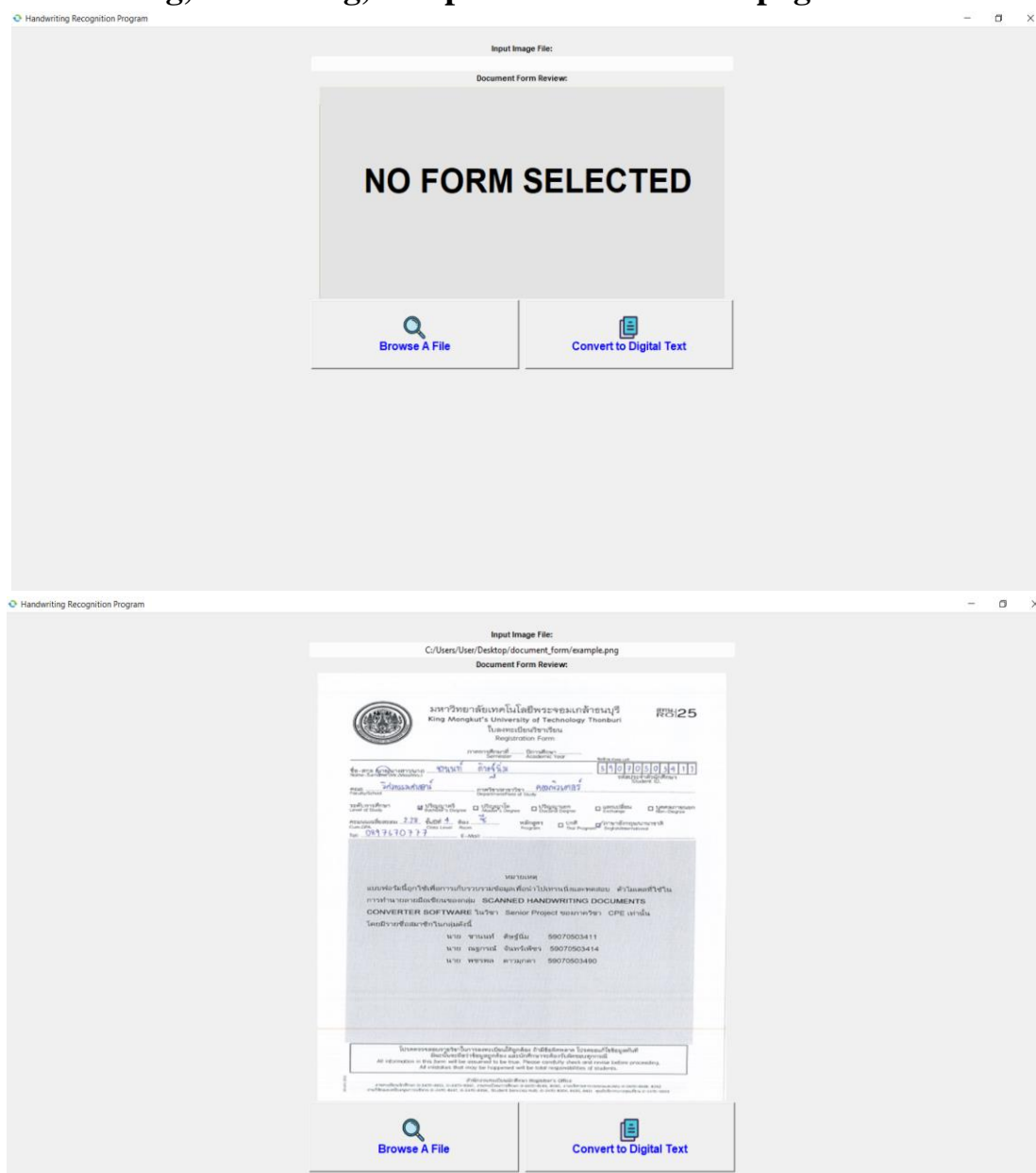


Figure 3.13: Inserting a document (Upper) and Preview a scanned document (Lower)


An inserting page is a page that can select the desired document to make a converting step, but before inserting step, the user needs to scan the document from the scanner machine to receive a digital file on a personal computer. After the user selected the scanned document file by clicking a search input button on the left page below. The screen will display a preview picture and path of the document for the user in Figure 3.13. If users would like to convert information on the document, they can click a convert button on the right page below for starting to convert the process.

3.2.1.3 Displaying scanned information page

Handwriting Recognition Program

Digital Texts Review:

Student ID:	59070503411
First Name:	ชานนที
Last Name:	ปัญญ์
Faculty:	วิศวกรรมศาสตร์
Department:	คอมพิวเตอร์
GPAX:	2.28
Class Level:	4
Class Room:	ซี
Telephone:	0897670577
Email:	


CANCEL



SAVE

Figure 3.14: Checking and storing information

For the last page of the program, the software is checking and storing information. After the user clicks a convert button from the previous page, the screen will automatically change to this page for indicating all information that is commuted from handwritten text to digital format. The user can correct the information and fix them on the screen by double-clicking on information that would like to change and then type a new data replace them. When they finished the checking process and would like to store data into the storage system, they can click a store button on the center page below for storing data into the database system.

3.3 Modeling

3.3.1 Data

3.3.1.1 Gathering data from internet

These data are the 44 alphabets, 21 vowels, and 4 tone markers. We consolidated follow 68 of Thai language by each of the alphabets, vowels and tone markers we collected various handwriting from real people. By the way, we need these data to prepare for training the model that is important for predicting the Thai language.



Figure 3.15: Dataset Thai language alphabets, vowels and tone makers



Figure 3.16: Example of the Thai alphabet of handwriting people

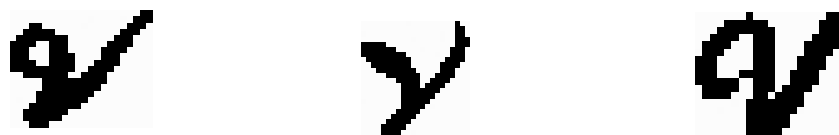


Figure 3.17: Example of Thai tone markers of handwriting people

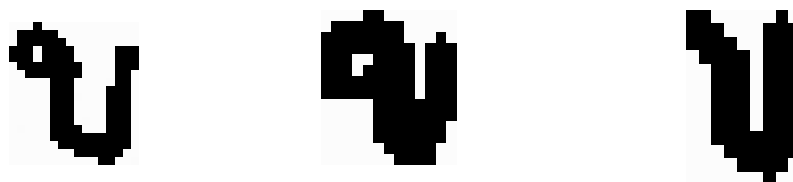


Figure 3.18: Example of Thai vowels of handwriting people

3.3.1.2 Imitated data

There are two forms that we have created for gathering handwriting. Firstly, Figure 3.20 shows the registration document from King Mongkut's University of Technology Thonburi (KMUTT). We use this form to consolidate the data from students within KMUTT as a dataset for the project because the registration form is quite similar to a document that we have found the problem from real situations such as registration in the hospital. It needs handwriting from customers or students to fill all data into the document. For the Imitated form, it has a lot of data on the document, but we selected only nine parts that included student id, first name, last name, faculty, department, GPA, class level, and room. Figure 3.21 presents an example of a handwritten document that we have consolidated data from KMUTT students. Secondly, we have created another form which is a set of unique sentence forms. This form consists of two pages and each page has 10 sentences for writing the sentence followed by given unique sentences. We use this form to collect handwriting from people who are aged around 16 to 60 years. Figure 3.22 and 3.23 below is an example of sentence form.

มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
King Mongkut's University of Technology Thonburi
ใบลงทะเบียนวิชาเรียน
Registration Form

ภาคการศึกษาที่ ปีการศึกษา
Semester Academic Year

ชื่อ-สกุล (นาย/นางสาว/นาง)
Name-Surname (Mr./Miss/Mrs.)

รหัสประจำตัวนักศึกษา
Student ID

คณะ ภาควิชา/สาขาวิชา
Faculty/School Department/Field of Study

ระดับการศึกษา
Level of Study

☐ ปริญญาตรี Bachelor's Degree ☐ ปริญญาโท Master's Degree ☐ ปริญญาเอก Doctoral Degree ☐ แลกเปลี่ยน Exchange ☐ บุคคลภายนอก Non-Degree

คะแนนเฉลี่ยสะสม ชั้นปีที่ ห้อง หลักสูตร
Cum.GPA Class Level Room Program

☐ ปกติ Thai Program ☐ ภาษาอังกฤษ/นานาชาติ English/International

Tel: E-Mail:

หมายเหตุ
แบบฟอร์มนี้ถูกใช้เพื่อการเก็บรวบรวมข้อมูลเพื่อนำไปเทรนนิ่งและทดสอบ ตัวโมเดลที่ใช้
ในการทำนายลายมือเขียนของกลุ่ม SCANNED HANDWRITING DOCUMENTS
CONVERTER SOFTWARE ในวิชา Senior Project ของภาควิชา CPE เท่านั้น
โดยมีสมาชิกในกลุ่มดังนี้
นาย ชานนท์ ดิษฐ์นิ่ม 59070503411
นาย ณฐกรณ จันทระพิเชษฐ์ 59070503414
นาย พชรพล ดามมกดา 59070503490

โปรดตรวจสอบความถูกต้องในการลงทะเบียนให้ถูกต้อง ถ้ามีข้อผิดพลาด โปรดขอแก้ไขข้อมูลทันที
All information in this form will be assumed to be true. Please carefully check and revise before proceeding.
All mistakes that may be happened will be total responsibilities of students.

สำนักงานทะเบียนนักศึกษา Registrar's Office
งานทะเบียนนักศึกษา O-2470-8151, O-2470-8347, งานทะเบียนการศึกษา O-2470-8149, 8150, งานบริหารทางสอนและสอบ O-2470-8148, 8352
งานวิจัยและสนับสนุนการบริหาร O-2470-8147, O-2470-8356, Student Services Hub: O-2470-8354, 8420, 8421 ศูนย์บริการบางขุนเทียน O-2470-9493

Figure 3.19: Imitated register form


 มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
 King Mongkut's University of Technology Thonburi
 ใบลงทะเบียนวิชาเรียน
 Registration Form

ฤดูการศึกษาที่ ปีการศึกษา
 Semester Academic Year

ชื่อ-สกุล (นาย/นางสาว/นาง) จันทนา ดิมจิโน
 Name-Surname (Mr./Miss/Mrs.)

วิชา วิศวกรรมศาสตร์
 Faculty/School

ภาควิชา/สาขาวิชา คอมพิวเตอร์
 Department/Field of Study

ระดับการศึกษา ☒ ปริญญาตรี
 Level of Study Bachelor's Degree

☐ ปริญญาโท
 Master's Degree

☐ ปริญญาเอก
 Doctoral Degree

☐ แลกเปลี่ยน
 Exchange

☐ บุคคลภายนอก
 Non-Degree

คะแนนเฉลี่ยสะสม 2.28
 Cum.GPA

ชั้นปีที่ 4
 Class Level

ห้อง ๒๕
 Room

หลักสูตร ☐ ปกติ
 Program Thai Program

☒ ภาษาอังกฤษ/นานาชาติ
 English/International

Tel: 0897670777 E-Mail:

รหัสนักศึกษา 59070503411
 Student ID

Figure 3.20: An example of handwriting text

ชื่อ – นามสกุล

เพศ

อายุ

ปี

เกิดอุทกภัยขึ้นที่ทางจังหวัดภาคใต้

ขภาพุดด้วยคำที่มีความเกี่ยวข้องกันถูกต้องตามระเบียบของภาษา

พระเกรซดีระสังทุกเข้ามิด

ภารโรงสมนึกเป็นคนดี เขาชอบทำบุญให้ทานตอนเช้าๆ

อารีขายผลไม้ทุกวันหน้าตาดีแต่ยังไม่มีแฟน

นิธิโรจน์ไม่ชี้แจงเรียนหนังสือ

วันฉายไม่เคยกินส้มตำ เพราะเธอไม่ชอบมะละกอ

เด่นกรุณาถอดรองเท้าก่อนเข้าห้องเรียน

โปรดทำความเคารพ เมื่อเดินผ่านประธานในพิธี

ทำงานให้หนักเพื่อเพียงพอจะได้ประสบความสำเร็จ

Figure 3.21: The first page of the second form

สามารถหาซื้อสินค้าได้ตามร้านค้าทั่วไป

สูตรเฉพาะของณัฐฐิ์ใช้ผสมสตรอร์เบอร์รี่

อยู่ตรงไหนแล้วช่วยก็อยากอยู่ตรงนั้นไปนานๆ

พินทุกรอบแบบนี้จะรอดได้ไง ถ้ามันต้องรีบไปซื้อมาลองเลย

บรรยากาศศึกษาคึกคักขนาดนี้ คนรักปลาจะพลาดได้อย่างไร

รูปและลายเส้นของเมอปรางใครๆก็อยากได้

พีเจอรี่นี้ออกแบบมาเพื่อให้คนพิการและผู้มีปัญหาทางสายตาใช้งานนำทางได้ดีขึ้น

เวลาไม่อาจช่วยรักษาทุกสิ่ง แต่การยอมรับความจริงจะรักษาทุกอย่าง

เริ่มลงหลักปักฐานฝ่าหัวใจกับใครควรคิดให้ดีๆ

พื้นที่ของที่ขับเคลื่อนประเทศชาติพัฒนามีสามอย่าง

Figure 3.22: The second page of the second form

3.4 Backend

3.4.1 Database schema

In our system, we decided to use a non-relational database as it is a flexible database and can handle a huge amount of data. A JSON form is used to store data because this kind of format is similar to the structure of a document. The structure of the file or schema includes student id, first name, last name, faculty, department, GPA, class level, room, and telephone number. All of them are information on the imitated registration form. The picture below is an example data of one student in a JSON file.

```
{
  "Student": [
    {
      "studentID": "59070503411"
      "firstName": "ชานนท์"
      "lastName": "ดิษฐ์นิ่ม"
      "faculty": "วิศวกรรมศาสตร์"
      "department": "คอมพิวเตอร์"
      "GPA": "2.28"
      "classLevel": "4"
      "room": "สี่"
      "tel": "0897670577"
    }
  ]
}
```

Figure: 3.23: The structure of JSON file

3.5 Program Workflow

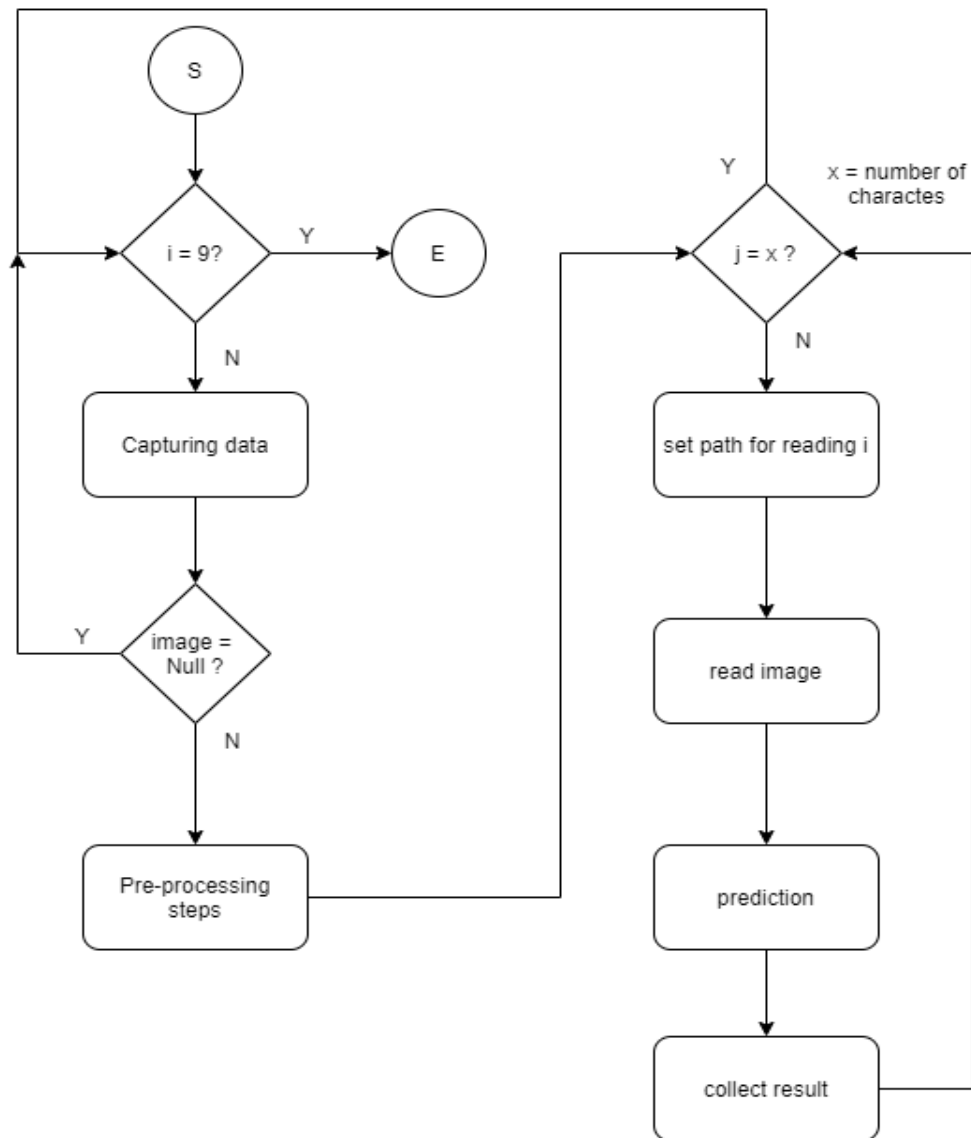


Figure 3.24: Workflow of the program

Firstly, $i = 9$ means there are nine variables which are first name, last name, faculty, department, telephone, GPA, year, classroom, and student ID. Each of the variables will be cropped and checked whether it appeared or not. For example, if some information is not written the next variable will be cropped and checked, respectively. If the variable exists, it will get into the pre-processing steps. After completing the pre-processing steps, a loop will run continually according to the number of characters. The pictures are read following the path and then every picture that is in the path will be predicted and collected.

Chapter 4: Implementation, Results, and Discussion

4.1 Frontend implementation

4.1.1 User interface

After a user understood the information on a tutorial page and made a file browsing by click. He will see the file path and his document. So, he will be able to convert from handwriting to digital texts from clicking on a convert to digital text button. After that, the program will be showing a digital-texts review to him, and he can decide between saving into a database or canceling it to browse a new file.

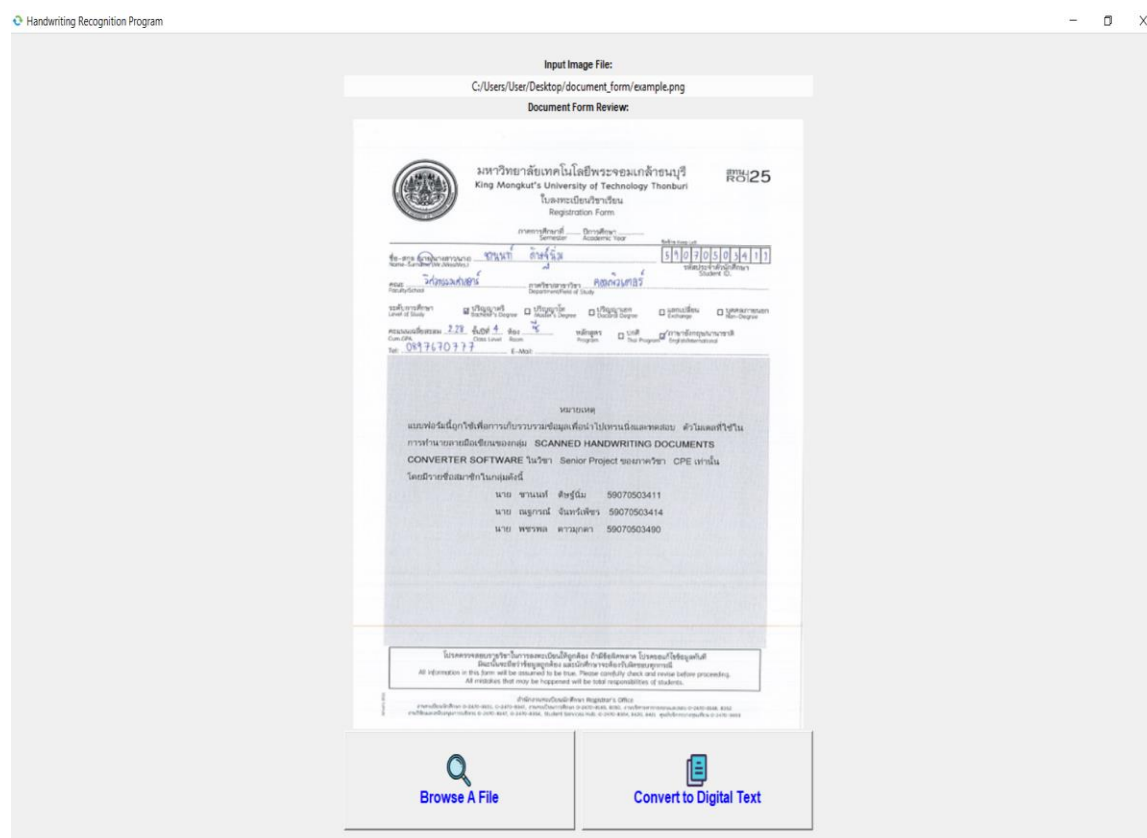



Figure 4.1: The result of browsing a file

Handwriting Recognition Program

Digital Texts Review:

Student ID:	59070503411
First Name:	ชานนัท
Last Name:	ดิษฐ์นิม
Faculty:	วิศวกรรมศาสตร์
Department:	คอมพิวเตอร์
GPAX:	2.28
Class Level:	4
Class Room:	ซี
Telephone:	0897670577
Email:	


CANCEL



SAVE

Figure 4.2: The result of converting to digital texts

4.2 Modeling implementation

4.2.1 Data consolidation

For the first implementation, it is a handwritten consolidation. We have two main resources to gather the handwritten data which are the internet and real people. For the internet, the name of this dataset are Alice off-line Thai handwritten characters or ALICE-THI, and All Hears me Thai handwritten characters. They have gathered all characters from 150 Thai writers such as letters and vowels. Next, gather from real people, we have created two of our own forms for filling information on a document which is a KMUTT registered form and a sentence form. For the KMUTT form, we have consolidated information from 85 students who study in the KMUTT and are aged from 20 to 23 years. Next, the second form, consists of 10 unique sentences and we have collected from 30 people, but in this form, it has many unique and several handwriting people who are aged from 16 - 80 years. After gathering data from both forms, we have to make a character segmentation for splitting a word or sentence into pieces, such as manually segmenting or using a program. Figures 4.1 - 4.6 are examples of each form from different writers.

ภาคการศึกษาที่ 1 ปีการศึกษา 2019
Semester Academic Year

ชื่อ-สกุล (นาย/นางสาว/นาง) นกขันธ์ สอน
Name-Surname (Mr./Miss/Mrs.)

รหัสประจำตัวนักศึกษา 59070503415
Student ID.

คณะ วิศวกรรมศาสตร์ ภาควิชา/สาขาวิชา วิศวกรรมคอมพิวเตอร์
Faculty/School Department/Field of Study

ระดับการศึกษา ☒ ปริญญาตรี ☐ ปริญญาโท ☐ ปริญญาเอก ☐ แลกเปลี่ยน ☐ บุคคลภายนอก
Level of Study Bachelor's Degree Master's Degree Doctoral Degree Exchange Non-Degree

คะแนนเฉลี่ยสะสม 3.39 ชั้นปีที่ 4 ห้อง C หลักสูตร ปกติ ภาษาอังกฤษ/นานาชาติ
Cum.GPA Class Level Room Program Thai Program English/International

Tel: 08-1300-0702 E-Mail: nongkhan.suan@kmutt.ac.th

Figure 4.3: An example of KMUTT register form

ภาคการศึกษาที่ 1 ปีการศึกษา 2562
Semester Academic Year

ชื่อ-สกุล (นาย/นางสาว/นาง) นกขันธ์ สอน
Name-Surname (Mr./Miss/Mrs.)

รหัสประจำตัวนักศึกษา 59070503437
Student ID.

คณะ วิศวกรรมศาสตร์ ภาควิชา/สาขาวิชา วิศวกรรมคอมพิวเตอร์
Faculty/School Department/Field of Study

ระดับการศึกษา ☒ ปริญญาตรี ☐ ปริญญาโท ☐ ปริญญาเอก ☐ แลกเปลี่ยน ☐ บุคคลภายนอก
Level of Study Bachelor's Degree Master's Degree Doctoral Degree Exchange Non-Degree

คะแนนเฉลี่ยสะสม 3.99 ชั้นปีที่ 4 ห้อง C หลักสูตร ปกติ ภาษาอังกฤษ/นานาชาติ
Cum.GPA Class Level Room Program Thai Program English/International

Tel: 011-234-5678 E-Mail: brown.friends@line.com

Figure 4.4: One example of KMUTT register form

แมวอ้วนเล่นในสวนเดินวนไปวนมา
 แมว อ้วน เล่น ใน สวน เดิน วน ไป วน มา

ภารโรงสมนึกเป็นคนดี เขาชอบทำบุญให้ ทาน เยอะๆ
 ภารโรง สมนึก เป็น คน ดี เขา ชอบ ทำ บุญ ให้ ทาน เยอะ ๆ

อาชีวะผลไม้ทุกวัน หน้าตาดี แต่ยังไม่ม่แฟน
 อาชีวะ ผลไม้ ทุก วัน หน้าตา ดี แต่ ยัง ไม่ มี แฟน

นิธิโรจน์ไม่ใช่เกย์เรียนหนังสือ
 นิธิโรจน์ ไม่ ใช่ เกย์ เรียน หนังสือ

Figure 4.5: An example of our form

ลักทรัพย์ต้องไปศาลอาญา แต่รักนะคะ ต้องसानสัมพันธ์
 ลัก ทรัพย์ ต้อง ไป ศาล อาญา แต่ รัก นะ คะ ต้อง สาน สัม พัน ธ์

ความโสดไม่ใช่เนื้อคู่ ไม่ต้องอยู่กับกูไปตลอดก็ได้
 ความ โสด ไม่ ใช่ เนื้อ คู่ ไม่ ต้อง อยู่ กับ กู ไป ตลอด ได้

รักใครอย่าทิ้ง รักจริงอย่าจาก รักใครอย่าพรากร รักแต่ปากอย่ารักเลย
 รัก ใคร อย่า ทิ้ง รัก จริง อย่า จาก รัก ใคร อย่า พรากร รัก แต่ ปาก อย่า รัก เลย

Figure 4.6: An example of our form

Then, we separately collected each letter in an individual folder. The table below is all the numbers of each letter in Thai that we have gotten from the two sources.

Number	Middle-level character	Alice dataset	KMUTT form + gathering internet	Count
1	ก	218	97 + 200	515
2	ข	221	43 + 198	462
3	ฃ	212	200	412
4	ค	210	51 + 200	461
5	ฅ	208	200	408
6	ฆ	206	200	406
7	ง	219	60 + 200	479
8	จ	217	63 + 200	480
9	ฉ	203	13 + 200	416
10	ช	210	26 + 199	435
11	ฌ	208	6 + 199	413
12	ฎ	207	199	406
13	ฏ	213	20 + 198	431
14	ฐ	221	195	416
15	ฑ	220	195	415
16	ฒ	221	194	415
17	ณ	213	198	411
18	ด	214	198	412
19	ณ	208	1 + 200	409
20	ต	207	85 + 200	492
21	ถ	210	41 + 200	451
22	ถ	217	200	417
23	ท	200	75 + 200	475
24	ธ	211	3+210	424
25	น	205	110 + 211	526
26	บ	206	69 + 209	484
27	ป	221	34 + 211	466
28	ผ	214	10 + 211	435
29	ฝ	221	211	422
30	พ	212	53 + 211	476
31	ฟ	220	5 + 209	434
32	ภ	209	1 + 211	421

33	ม	213	92 + 208	513
34	ย	213	39 + 206	458
35	ร	219	79 + 205	503
36	ฤ	221	200	421
37	ล	216	49 + 206	471
38	ฦ	221	188	409
39	ว	211	52 + 203	466
40	ฬ	213	4 + 202	419
41	ศ	215	2 + 205	422
42	ษ	212	45 + 203	460
43	ห	207	36 + 202	445
44	พ	218	198	416
45	อ	221	92 + 196	509
46	ฮ	220	195	415

Figure 4.7: The table of all letter in middle level

47	ฅ	218	96	314
48	ฌ	214	11 + 92	317
49	ฌ	221	86 + 94	401
50	ฌ	220	31 + 94	345
51	ฌ	221	33 + 96	350
52	ฌ	221	56 + 94	371
53	ฌ	221	5 + 96	322
54	ฌ	220	94 + 94	408

Figure 4.8: The table of all vowels in middle level

Number	Upper-level vowel	Alice dataset	KMUTT form + gathering internet	count
1	ᵉ	220	91 + 96	407
2	ᵇ	219	39 + 94	352
3	ᵃ	217	51 + 93	361
4	ᵇ	221	3 + 96	320
5	ᵃ	221	33 + 69	323
6	ᵇ	207	24 + 90	321
7	ᵇ	221	52 + 96	369
8	ᵉ	221	70 + 94	385
9	ᵉ	209	89	298
10	ᵉ	199	96	295
11	ᵉ	210	33 + 96	339
12	ᵉ	168	15	183

Figure 4.9: The table of all letter in upper level

Number	Lower-level vowel	Alice dataset	KMUTT form + gathering internet	count
1	ᵉ	220	79 + 94	393
2	ᵉ	190	58 + 93	341

Figure 4.10: The table of all letter in lower level

Number	Digit	KMUTT form + gathering internet	count
0	0	52	52
1	1	40	40
2	2	39	39
3	3	45	45
4	4	44	44
5	5	60	60
6	6	45	45
7	7	50	50
8	8	40	40
9	9	50	50

Figure 4.11: The table of all digit data

After we finished data consolidation and data segmentation, there are two problems that we are facing in this process. The first problem is our forms do not have a diversity of letters because after segmenting consolidated data into pieces, there are a few characters that disappear in the dataset and the number of some letters are too small such as ๗, ๘, ๙, and ๐. The second problem is time-consuming. When separating a word or sentence into characters, we have to split them by ourselves because our character segmentation program works incompletely. However, we decided to create a new form that can append some letters that do not appear for solving the first problem. For time-consuming, we need to find a new algorithm that can work perfectly for the character segmentation.

4.2.2 Data preprocessing

4.2.2.1 Capturing data

The purpose of this step is to capture all interesting data on a document. In the KMUTT register form, eight pieces of information will be converted to the digital format. They are student id, first name, last name, faculty, department, GPA, class level, room, and telephone number. Firstly, we used a Python Imaging Library or PIL that provides interpreters with image editing abilities. So, we need to manually find the position of each information on the document, each position consists of 4 sides of each desired information. Moreover, the PIL has a useful function to cut that information.

After capturing, three problems occur while we implement a capturing data step. The first problem is having noise on the cropped image. It is not useful information and is a major obstacle in this step. Moreover, there are three kinds of noise we have found. Which are a dotted underline, printed character, vertical line, and small dot noise in Figure 4.12. For the first type of noise, it is the dotted underline, which is a line to identify a writing position for filling information on the paper. We eliminated the line by creating a function that can detect the position and remove it from the cropped image. A concept of this function is computing the sum of pixels in each horizontal line or horizontal histogram, then finding the highest sum of pixels. So, we will assume that position is the dotted line. After we found the position of the line, we will extend the position by 3 indexes in a row because if we remove only the highest sum of pixels in a horizontal line, it will have some remaining dotted underline (see Figure 4.13). Figure 4.14 shows the result of removing the first kind of noise. The removing dotted underline, it has a disadvantage which is any characters are written on the underline, it always deletes from the cropped image in Figure 4.15 This problem occurs at a time from 94 consolidated forms and also there is no solution for solving. However, it is not an important problem in this section.

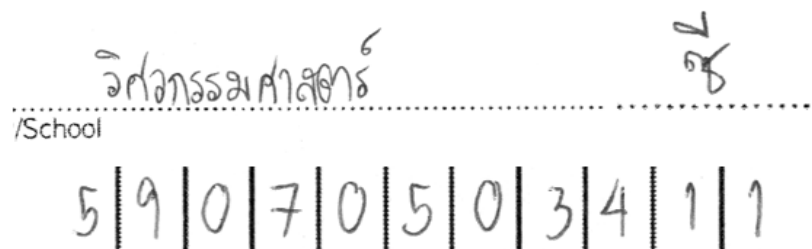


Figure 4.12: An example of each kind of noise

Left top: printed character, Right top: dotted underline and Bottom: frame student ID



Figure 4.13: The result of removing the dotted underline



Figure 4.14: The disadvantage of dotted underline

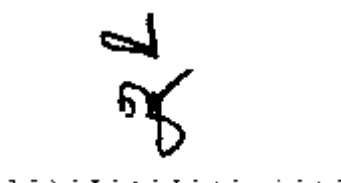


Figure 4.15: Remaining dotted underline

Next, it is a printed character. All documents always have the character to denote what type of data that a user needs to write information on it. Unfortunately, this kind of noise, there is no pattern to automatically detect a position for removing. We have to manually find it on each cropped image and create a function to remove the printed character in the only faculty and department (see Figure 4.16). Third noise, vertical line, it only occurs on student ID. We eliminated it by creating a method that is similar to the function above. This method will find the highest sum of pixels in each vertical line or vertical histogram and mark the highest sum of lines for removing. Before removing the vertical line, we will extend the position of the highest sum pixels by three for deleting the line as much as possible. However, we cannot eliminate the vertical line in student ID, there is some small noise that is on the student ID image in Figure 4.17 However, this consequence is another issue that we need to solve as quickly as possible.

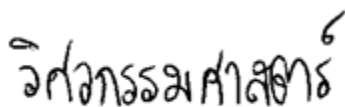


Figure 4.16: The result of removing dotted underline and repeated character



Figure 4.17: The result of removing vertical line and noises

From our research, we found a method to remove the noise. It is a set of techniques of morphology in image processing. Meanwhile, we use a method that is an opening operation with

a 2x2 block of the kernel for deleting the noise in all cropped images. During using the morphology technique, it has a problem to make something with characters on the cropped images. Which is the inconsistency of the letter (see Figure 4.18), because it depends on the heaviness of people's handwriting. For the cropped images that do not digit, we not allow the morphology technique to process on character images since it can change some pixels within the image and make it hard to do other steps in the future such as segmentation step. In the digit images, they can use this method for eliminating noise because the complexity of the digit images is less than the character images. However, we still have a new problem during solving the noise problem which is the inconsistency of the letter that humans wrote the form.

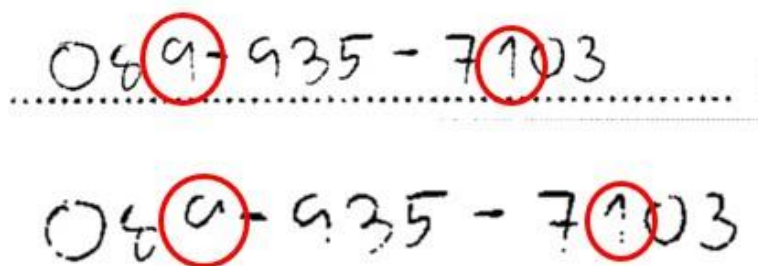


Figure 4.18: Inconsistency of the letter

The problem in the above paragraph, we can solve it by using other morphology methods which are dilation and closing operations. Firstly, dilation is a method that is used to expand the boundary of a region to digit size in an image. Another one, it is a closing operation which is connecting near regions to each other. For our experiment in the digit images, this technique is not efficient in the image that has many cursive digits such as phone number and GPA because an enlarging region in both operations makes the cursive digits stick together (see Figure 4.19). However, there is only one digit image that completely works without any problem. Which is a student ID because there is a lot of blank space between digits that make the dilation and closing operation have more efficiency in Figure 4.20.

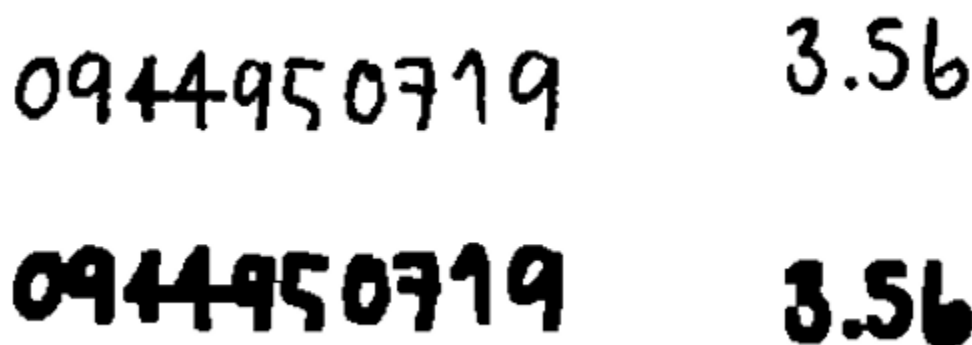


Figure 4.19: Dilation and closing with cursive digits

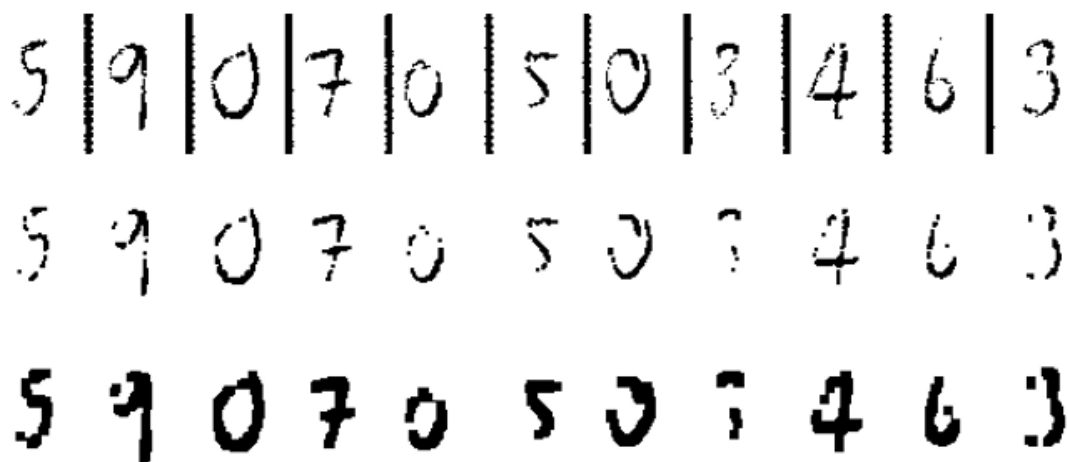


Figure 4.20: Before and After using morphology

Top: cropped image, Center: after using opening operation, and Bottom: after using closing and dilation

For the second problem, it is scanning a document. A scanner is another thing that affects the capturing data step. There are two factors to the problem which are the band's product and position of scanning documents. From our experiment, scanning the same document and the same position in different bands and generation of the scanner, it provides different results. Next, the position of scanning is another factor that makes a result of scanning hard to implement the capturing data. Generally, the scanner has only one slot for scanning and there are four angles to lay the paper on the scan slot before scanning. As you can see in Figure 4.21, it shows the result after capturing data with the same position of capturing and different positions of scanning. So, this step cannot be used with all scanners, it needs to manually find the position of capturing and identify the position of scanning to a user.

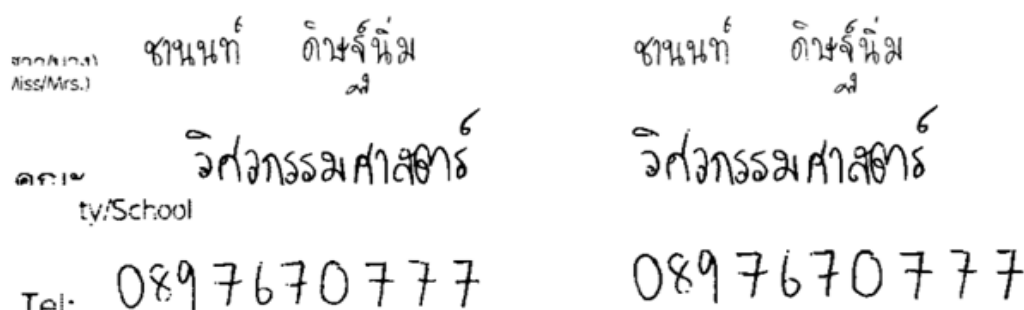


Figure 4.21: The result of capturing information in the same scanner, but the different position of scanning.

Left: Scan a document from bottom right angle Right: Scan a document from other angles.

Lastly, the third problem is not filling the information in some sections on the KMUTT register form. As you can see in Figure 4.22, there is no filler's phone number; they just fill a dash. Moreover, sometimes the filler does not write anything on the paper. At the first time, the algorithm captures those blank spaces or dash and then passes it to the next step. This is wasting time because the algorithm has to process not use data in many next steps. To solve the problem, we created a condition which can detect those data and then ignore them. After cropping data in each section, the capped image will be calculated to find a percentage of all black pixels within the image and then compare it to a threshold value as two. If the percentage of all black pixels is less than the threshold value, the algorithm will assume the cropped image as an empty space and ignore it.

ชื่อ-สกุล (นาย/นางสาว/นาง) Name-Surname (Mr./Miss/Mrs.)		พงศ์พล อามฤตา		รหัสประจำตัวนักศึกษา Student ID.		5 9 0 7 0 5 0 3 4 9 0	
คณะ Faculty/School		วิศวกรรมศาสตร์		ภาควิชา/สาขาวิชา Department/Field of Study		คอมพิวเตอร์	
ระดับการศึกษา Level of Study		<input checked="" type="checkbox"/> ปริญญาตรี Bachelor's Degree		<input type="checkbox"/> ปริญญาโท Master's Degree		<input type="checkbox"/> ปริญญาเอก Doctoral Degree	
		<input type="checkbox"/> แลกเปลี่ยน Exchange		<input type="checkbox"/> บุคคลภายนอก Non-Degree			
คะแนนเฉลี่ยสะสม Cum.GPA.		4.00		ชั้นปีที่ Class Level		4	
		ห้อง Room				8	
หลักสูตร Program		<input type="checkbox"/> ปกติ Thai Program		<input checked="" type="checkbox"/> ภาษาอังกฤษ/นานาชาติ English/International			
Tel: -		E-Mail:				พงศ์พล@hotmail.com	

Figure 4.22: Phone number section is filled on the paper.

4.2.2.2 Binarization (Gray Scale)

Binarization is a process to transform a color image into a grayscale. Moreover, the process of binarization is to transform pixels in the foreground of the image that value between 0 to 255 to only 0 and 1 that 0 means of black color and 1 means of white color (Figure 4.23).



Figure 4.23: before binarization (left) and after binarization (right)

4.2.2.3 Thinning

Thinning is a process to decrease the size of the character. The process of thinning is to decrease the density of characters that has to work through binarization and then deciding to find hit and miss value to reduce the value of the pixel of characters. In this process, after thinning, it is suitable for using character segmentation because it is easier to cut character by character in the sentence. Figure 4.24 indicates before and after using the thinning step.

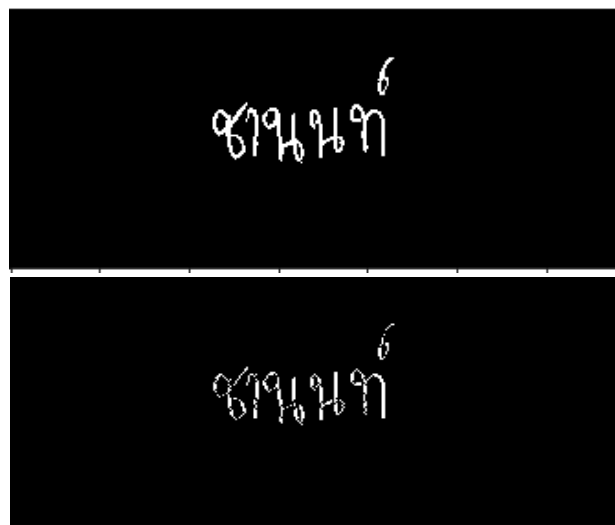


Figure 4.24: before binarization (left) and after binarization (right)

4.2.2.4 Slant correction

Slant correction is to adjust a skew character to be straight. For this process, use an image of a character work through 'For loop' that can adjust the angle between -45 to 45 degrees by increasing only 5 degrees in each image. Example, 'For loop number 1' that has an angle of a skew character -45 degrees and 'For loop number 2' has an angle of a skew character -40 degrees so, if algorithm finds all angle between -45 to 45 degrees then an algorithm will find the best by used "argmax" to adjust skew character in the sentence.

For Figure 4.25-4.26 below, we provide a special case if humans are writing italic characters in the sentence.

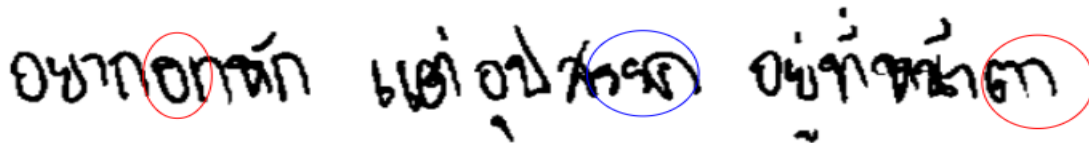


Figure 4.25: Before using Slant correction

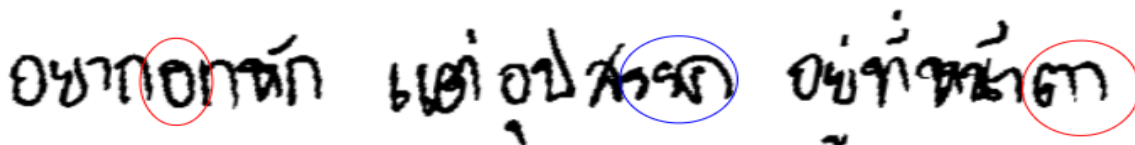


Figure 4.26: After using Slant correction

4.2.2.5 Word segmentation

Word segmentation is separating a sentence into words, it is especially used with only the name section on the KMUTT register form because the name consists of 2 parts which are first name and last name, but others have only one. Moreover, a crapped image is fetched into an algorithm as an input and calculates the sum of all pixels in a vertical line or vertical histogram. After the vertical histogram step, the algorithm will find ranges of blank space in the image. As you can see from a graph in Figure 4.27 below, there are three ranges of blank space in the sentence.

After we have the range of blank space, we will often have three ranges of blank space. For the first range, in front of the first name, the algorithm calculates a position by subtracting 10 with the last position of the range. Next, between first name and last name, it will compute the result of subtraction or different subtraction between the first and last index of the second range, and then divide it by 2 to be used as a segmented point. Finally, the last range, the algorithm calculates the third segmented point by bringing the last index of the last range and 10 for making an addition. In Figure 4.28, it indicates three lines after calculating the algorithm and a result of the word segmentation.

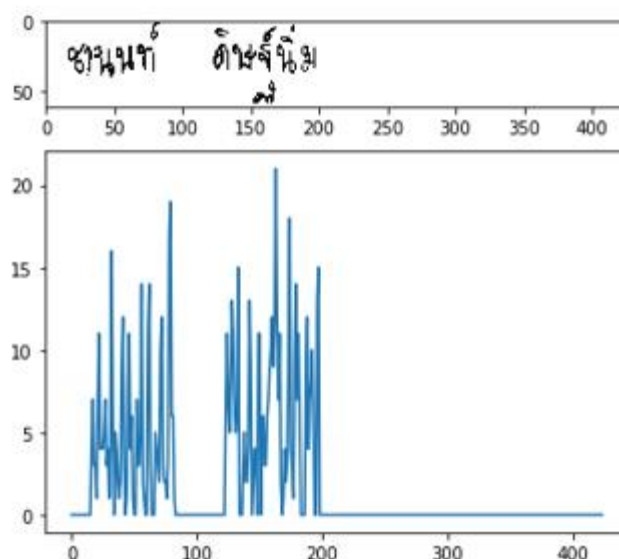


Figure 4.27: The sum of pixels in each vertical line (line chart)

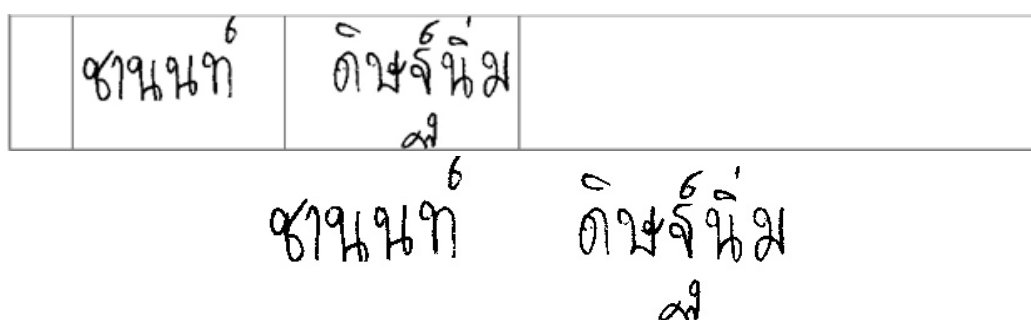


Figure 4.28: Above picture: three-segmented lines after computing of algorithm
Below picture: the result of word segmentation

4.2.2.6 Character segmentation

Character segmentation is an important and last step in data preprocessing, it has a lot of details and many parts in this section. This step is created to break a word or sentence into characters and an input of the character segmentation is an edited image from slant correction. During implementation in the first and second semesters, we can summarize and divide it into two main strategies for clear understanding. For the first strategy, it consists of two parts which are axis-segmentation and identifying levels. For the first part, it is the first step that can independently split the word and sentence image into character images. A major concept of the axis-segmentation is using a blank space between characters in the vertical and horizontal histogram or x- and y-axis.

Vertical segmentation, it is a finding line between characters in the y-axis. A process of this algorithm will compute the sum of pixels on all columns within an image and detect only a range of columns that has a value as zero because those ranges are blank space between characters or letters. After that, the algorithm will calculate a center value in each range of columns to find an appropriate point (Figure 4.29), but the first and last range will use another method. For the first range, an appropriate point will be the last index of range and subtraction by five. In the last range, it assigns a suitable line by using the first index of the last range and sum by three. So, we fetch a set of segmented points to another step for segmenting the word into characters. The next step, it is called separating characters (Figure 4.30).

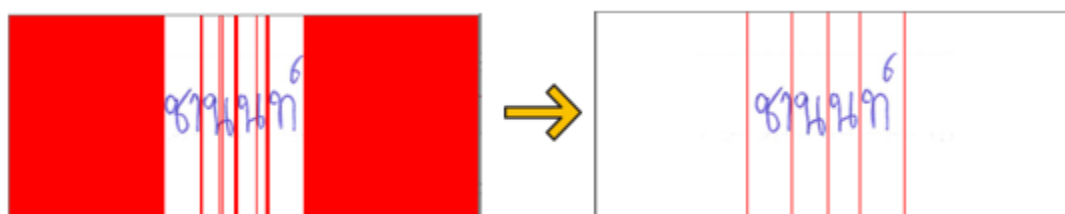


Figure 4.29: Finds a blank space in each column (left) and calculates an average point for segmentation (right).



Figure 4.30: The result after making segmentation.

Before going to the second part, we need to thoroughly explain the basic structure of the Thai language. A writing Thai language, it includes three main elements which are upper level, middle level, and lower level (see Figure 4.31) The first component, upper level, is tone maker for marking a pronunciation in each tone level in Figure 4.32. Moreover, sometimes, this level can have a vowel and tone at the same time such as มี้ , จี้ , กี้ , and นู้ . Next, it is a middle level (see Figure 4.33). This level has two parts which are the Thai alphabet and vowel. Lastly, lower level, it is a bottom part of all components. There are only two vowels at this level in Figure 4.34 below.

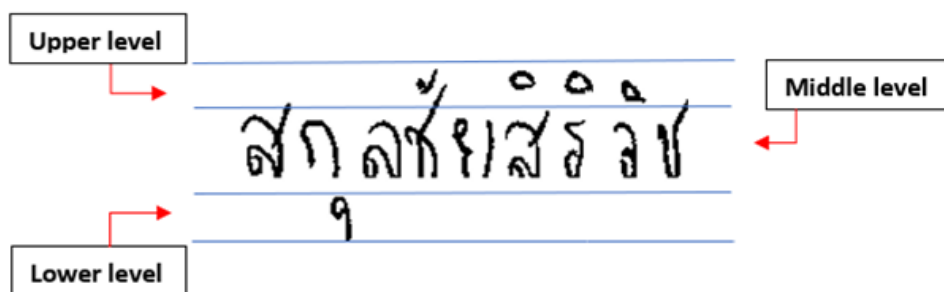


Figure 4.31: Three levels in Thai



Figure 4.32: A few examples of Tone marker

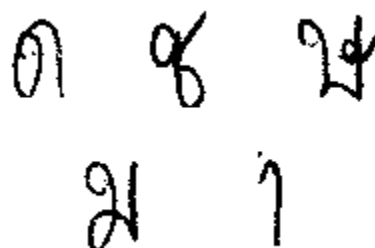


Figure 4.33: A few examples of middle

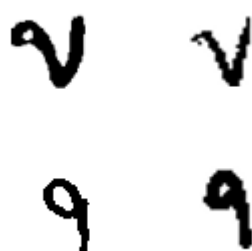


Figure 4.34: A few examples of lower level

Next, the second part is a horizontal segmentation. The role and working of this step are similar to the previous process, but it computes and segments in a horizontal line. So, this process is used to split between the middle level and other levels. During segmenting, the algorithm has to identify levels in each segmented character by defining the number of characters after making horizontal segmentation. Then if there is more than one character, the algorithm will calculate the sum of all black pixels within each character image and compare the sum value together. For the highest sum image, we assume that it will always be the middle level and the lower sum will automatically be other levels. Generally, horizontal segmentation always starts the process from above to the bottom of the character image and sorts a segmented character respectively. In Figure 4.35, it is another example of a character image on a different level, a character image has three elements which are tone level, sub-upper level, and middle level. The first two segmented images will be assigned the upper level because the sum of black pixels is less than the third segmented image. After we know each part has its level, this algorithm will automatically save the segmented image as PNG and name it by adding a digit of split-sequence and level. For naming each level, it has three forms which are u, m, and l. U stands for upper level, M is a middle level and the last form, L, is a lower level. We need to name the picture in this format because there are three models for prediction in machine learning. Figure 4.36 below is a result of segmentation.



Figure 4.35: Sublevels in the upper level



Figure 4.36: Result of segmentation

However, this step still has a large and hard problem for solving which is having no space between characters in vertical and horizontal segmentation. As you can see from the previous figure, the first one of the results, it is not completely splitting. It should use a new strategy or improve our algorithm that can segment characters without blank space. So, we will discuss the new technique in the next paragraph.

In the second semester, our team has continually researched on the internet and consulted our advisor to find a possible way for segmenting without blank space. After that, we found a lot of useful techniques in many pieces of research. Then we decided to break a segmentation part

into two models. Which are character and digit segmentation because the character segmentation has different factors and complexity more than digit segmentation such as digit has no upper and lower levels and the probability of occurring blank between characters. However, we still use the same techniques and the idea of algorithms in both methods.

For character segmentation, this step can be broken into three major parts which are pre-segmentation, vertical segmentation, and horizontal segmentation (see Figure 4.37). Firstly, the pre-segmentation step, there are two main steps that are removing blank space and keeping only the middle level. So, we are going to show you step by step about how they work.

Character Segmentation

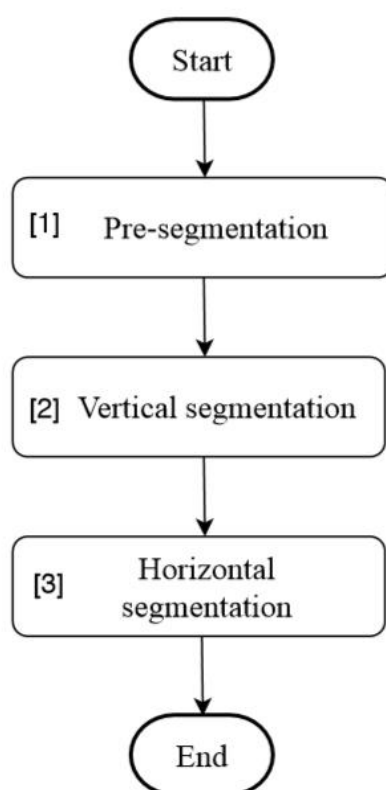


Figure 4.37: Overview flowchart of the character segmentation

For the first step, the input will be detected for blank space of both upside and downside on a horizontal axis by a horizontal histogram algorithm. Technically, they will be eliminated by the summarization of black pixels of the handwriting in each row (Figure 4.38). Which of the rows have a calculation as zero, it will be defined as blank space and it will be cleared up by using a hypothetical black space

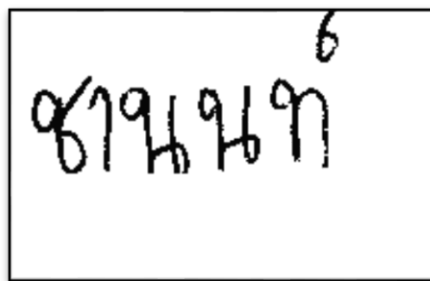


Figure 4.38: The input image

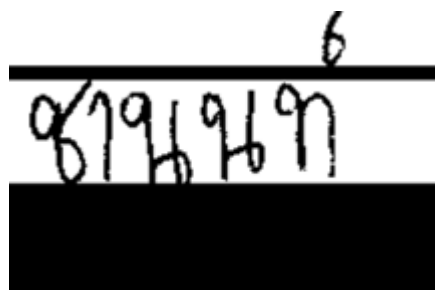


Figure 4.39: Blank space detected

As you can see from Figure 4.39, the blank space is removed after they have been detected as a hypothetical black color space. So, the remaining input will be only handwriting that fits in the horizontal axis as you can see from the picture Figure 4.40

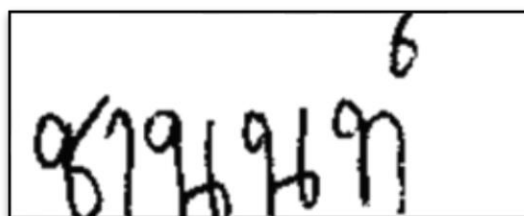


Figure 4.40: Blank space removed

The second step is keeping only the middle level. According to [1], A concept of their method is using a value of a horizontal histogram that has more than $1/2.8$ of its maximum as a threshold. Then, we use this threshold in the horizontal segmentation to detect only rows that have the summarization of black pixels more than the threshold value. It provides only the middle zone. After we have the middle image in Figure 4.41, we use the vertical segmentation, but the result was not good there are some segmented mistakes between “9” and “7” (see Figure 4.42).

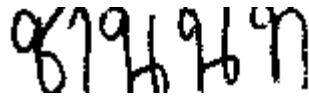


Figure 4.41: Upper and lower level removed



Figure 4.42: Result after vertical segment with the middle image

We continually try to use this method with other images, and the result is better as you can see from Figure 4.43.

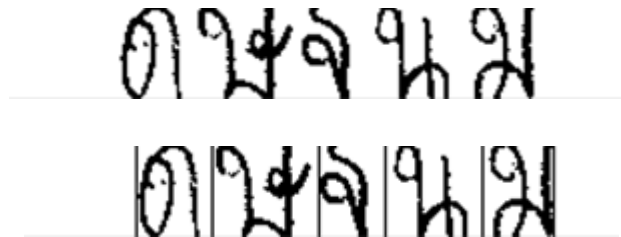


Figure 4.44: Vertical segment with the middle image

Nevertheless, this method is an indirect method to solve a segmenting without blank space that we have found in the first semester. If there is no blank space between characters, our algorithm still provides the same result.

Next, vertical segmentation part, before implementing this part, we have received a useful hypothesis from our advisor which is using some mathematical value to be a threshold in a vertical segmentation such as mean, mode, quartile, etc. we started at the quartile because the first quartile value is quite close to a zero which is a blank space in the first strategy. The first time, we did not use the pre-segmentation in this step because we need to see a performance of our hypothesis that can solve the problem or not. Our algorithm uses a vertical histogram by computing the sum of all columns of pixels in the y-axis and calculating the first quartile to be a threshold value. A result of using the first quartile with many character images, it did not work completely in Figure 4.45. There are many correct and incorrect points to make splitting. Moreover, the first quartile often is zero because a character image has white pixels more than black pixels. From the problem that we found in our experiments, we decided to find more conditions to avoid those problems. For the first quartile often is zero, we solved this problem by computing full quartiles and two more extra which are first (25%), second (50%), third (75%) quartile, 20%, and 35% respectively. After computing a set of quartiles, our algorithm will compare them in ascending order for finding the first non-zero to be used as the threshold value of segmentation. The second problem, incorrect and correct segmentation, we create a condition to check all values in vertical histograms that are

less than the value of threshold by examining all pixels of the detected column in a vertical line from top to bottom.

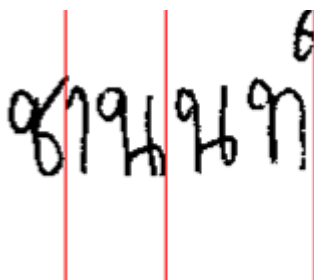


Figure: 4.45: The result of using a set of quartiles

As you can see from the result above, many wrong segmented lines are deleted from the image. So, the condition almost completely solves the problem and also provides a satisfying result. However, we need to keep finding a more method that can completely remove unnecessary lines.

The issue from the previous paragraph, we have an idea to build a function that uses a set of correct segmented lines to be the main part to make a splitting step and then compare it to another set of mixed segmented lines. For a set of correct segmented lines, we always use a vertical histogram with the threshold value is zero, then we implement the vertical histogram again with a value of threshold as non-zero to find a set of mixed segmented lines. Next step of the function, it will compare two sets of segmented lines following a condition which is a difference of subtraction between the first and second set need to be at least 8. If the difference of subtraction is less than 8, an element in the second set is eliminated. In other words, the function will delete an element in the second set that has the difference of subtraction with the first set less than 8. Figure 4.46 indicates the result after using the condition to compare both sets of segmented lines.

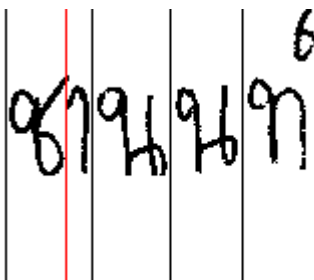


Figure 4.46: The result of using a set of quartiles. The Black lines are segmented by threshold as zero and White lines are segmented by the first non-zero

After we have proved and implemented two hypotheses, we tried to combine both methods. You can see results with many input images in Figure 4.47



Figure 4.47: A few examples of results with many input images

Thirdly, the last main part of character segmentation is horizontal segmentation, it consists of three main parts. Which are reading images from a folder, horizontal segmentation, and identifying levels in Figure 4.48.

Horizontal Segmentation

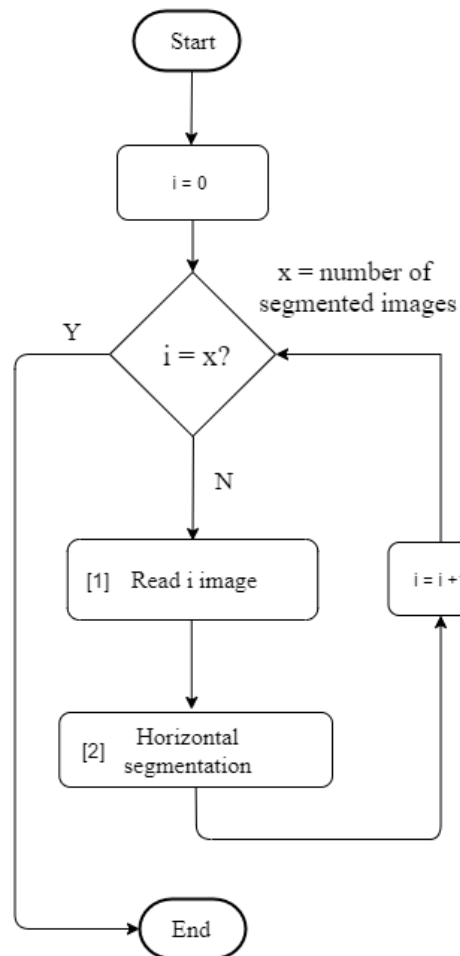


Figure 4.48: Flow chart of horizontal segmentation

An input of this part is a set of segmented images (see Figure 4.49). Firstly, A function will set a path of the image for reading from a target folder. For example, we now are in a first name section, the function will read the image from a folder named “data-doc” in drive C of your computer and then feed it to a horizontal segmentation step. Next, it is the horizontal segmentation section. This step will use a horizontal histogram algorithm to find a set of segmented lines and then identifying levels. The horizontal histogram, it will compute the sum of black pixels in each row. Then, we assign a value of threshold to be zero and then detect only rows that have the sum of blank pixels equal to the threshold. After we have a set of detected rows, the algorithm will find an appropriate point to be a segmented point in each range in Figure 4.50.



Figure 4.49: An input of horizontal segmentation



Figure 4.50: After finding suitable lines in the segmented image

The second part of the horizontal segmentation is identifying levels, it is an important step that is created to detect and identify levels in each section of the segmented image. To thoroughly explain how the identification of the level works, we have to use pseudocode and flowchart in a presentation. A concept of identifying levels, the first segmented image always is used to classify the type of formats and identify a position of segmented lines to remaining segmented images. You can see a flow of work in pseudocode below. After we finish the classification type of level, it will pass the number of segmented lines to the second of identifying levels.

```

1 Initialize global segmented level  $\leftarrow$  NULL
2 Initialize temporary segmented level  $\leftarrow$  NULL
3 Initialize current segmented level  $\leftarrow$  NULL
4 Initialize counter of number of character  $\leftarrow$  0
5 Initialize modes  $\leftarrow$  0
7 For i < number of characters
6     imgPath  $\leftarrow$  file path
7     img  $\leftarrow$  imread(imgPath)
8     compute the sum of blank pixels in every single rows of the image (horizontal histogram)
9     if (i = 0) then
10         find segmented lines from the horizontal histogram
11         current segmented level  $\leftarrow$  segmented lines
12         if (segmented line = 2) then
13             if (segmented image does not has lower level) then
14                 insert zero in initial current segmented level
15                 global level  $\leftarrow$  current segmented level
16                 mode  $\leftarrow$  1
17             else if (segmented images does not has upper level) then
18                 insert height of image in initial current segmented level
19                 global level  $\leftarrow$  current segmented level
20                 mode  $\leftarrow$  2
21             else if (segmented image has only middle level) then
22                 global level  $\leftarrow$  current segmented level
23                 mode  $\leftarrow$  4
24             else
25                 insert zero in initial current segmented level
26                 insert height of image in initial current segmented level
27                 global level  $\leftarrow$  current segmented level
28                 mode  $\leftarrow$  3
29         else if (segmented line = 3) then
30             if (segmented image does not has lower level) then
31                 insert zero in initial current segmented level
32                 global level  $\leftarrow$  current segmented level
33             else if (segmented images does not has upper level) then
34                 pop the first value of current segmented level
35                 insert zero in initial current segmented level
36                 insert height of image in initial current segmented level
37                 global level  $\leftarrow$  current segmented level
38             else
39                 mode  $\leftarrow$  0
40         else
41             find segmented lines in the sum of black pixels
42             remove segmented line between the first image and current image
43             call a function to the second part of identify levels
44             add number of segmented character to counter of number of character
45 return counter of number of character

```

Next, the second part of identifying levels will separate and identify levels in each part of the segmented image.

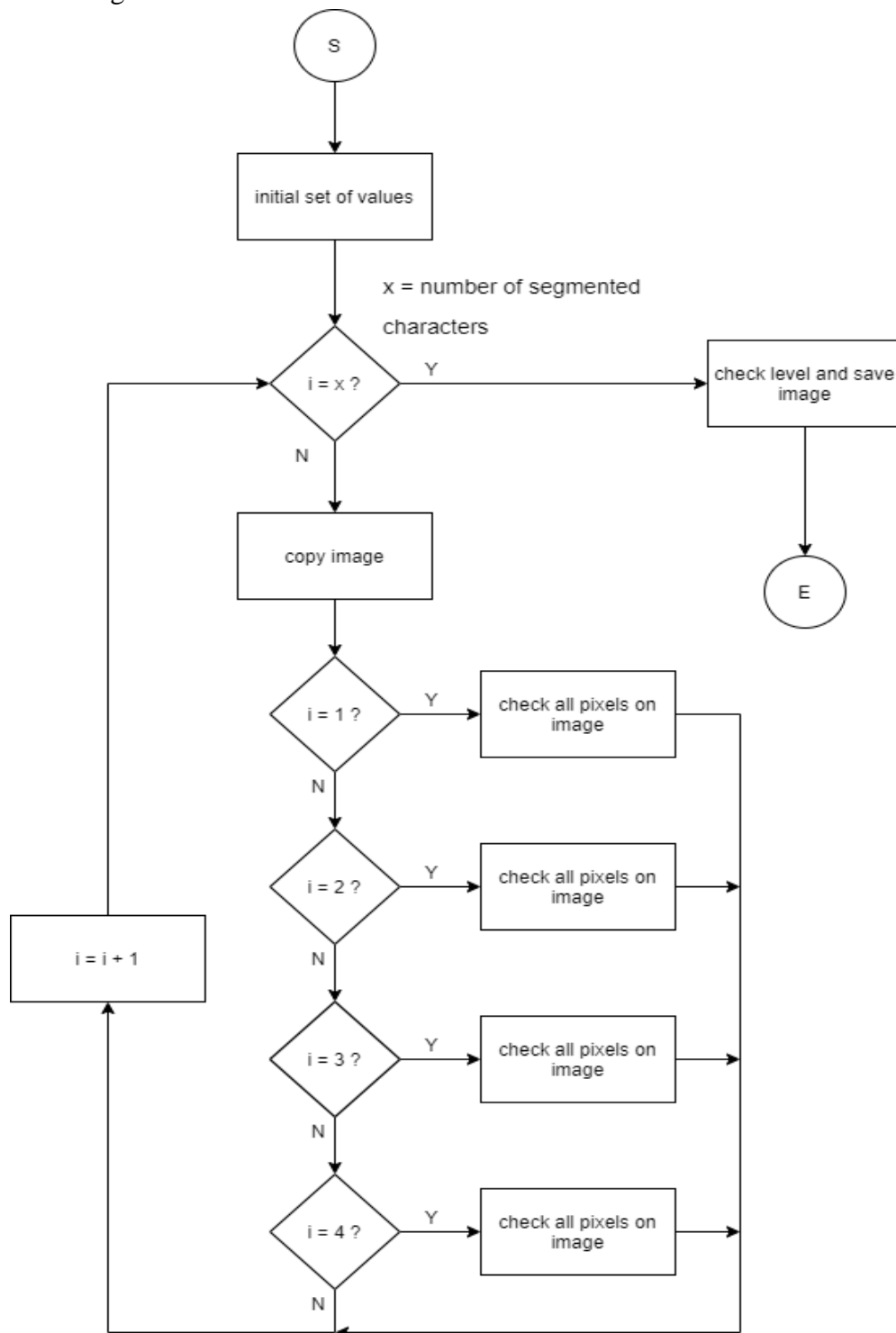


Figure 4.51: Flow chart of second part of identifying levels

Four variables will be initialized which are checkUpper, checkUpper2, checkMiddle, and CheckLower. The main purpose is to check an existing level by using these variables. So, we started setting them up as “True” at first. The loop will run following the number of segmented horizontal zones that received from the previous step. The data of the horizontal zone will be copied, the first upper zone will be executed at the first one. After that, the copied image will be checked. If the percentage of black pixels is less than 3.7, it will be assumed that it is not suitable and then it will be ignored for the next step which is saving data for prediction. Moreover, the initial variable that we mentioned before will be set to be “False” depending on the case that it reached. After the loop reaches the last segmented horizontal zone, the algorithm will only allow the black pixels of the horizontal zone that are more than 3.7 to identify a level by checking which of the initial is still being “True” and then it will be saved into a prediction folder.

For the number segmentation, first things first we would like to notice that we build another segmentation individually from the character segmentation because we do perform a digit model separately as well to be executed by the number segmentation particularly. So, we are going to elaborate it in detail step by step to show you how it works. Hypothetically, there is an input image which is a KMUTT registered form. There is information including character info and digit info, but in this part, we are going to be focusing on only digit info which are student id, GPA, class level, and telephone. Firstly, every single information such as student id will be operated by a horizontal histogram algorithm to eliminate blank space (white space) for both up and down space of the input picture by the summarization of black pixels in each row. Then, in each of the rows will be having a calculation of the black pixel. If which of the rows is zero, it will be defined as blank space and it will be cleared up. So, the remaining space of the input will be only the digits on the horizontal zone. Secondly, all of the digit information will be normalized continually by having conditions to implement each of the information differently. By the way, just to remind once again there are four digit information which are student id, GPA, class level, and telephone respectively. So, we are going to explain to them one by one. For the student id will be executed by a vertical histogram algorithm to explore blank space and get rid of it as what the horizontal histogram algorithm does as we clarified before, but the difference is just an axis. However, for the other digit information will be operated by the same vertical histogram algorithm as the student id, but there is an extended algorithm that will be operating by using a first-quartile value as a character segmentation that we explained before to look for the best suitable intersection of the input. Thirdly, there is an algorithm to detect uncommon things in the input such as point, dot by finding the ratio between black and white pixels to prevent saving them as a single-digit input to predict in the further step. Moreover, there are the criteria of the proportion of telephone number and GPA which can be separated by three points five. For example, if the percentage is more than three points five, the input will be defined as a digit. By the way, for student id and class level, use the two points five percent to interpret as a digit.

4.2.3 Machine learning

Firstly, we started performing handwritten recognition by using three components which are a single CNN model, the abilities of pre-processing steps, and just the Alice off-line Thai handwritten characters dataset. There were a lot of mispredictions. We struggled with the three main problems which are the various handwriting, the Alice dataset contains only too ideal handwriting, and there were too many classes per model. So, we were trying to solve the problems by duplicating the same dataset as double the amount, making the model overfitted. The result did not get better. After that, we decided to separate the model into four models which are the upper, middle, lower, and digit model. Moreover, we train the models by using Alice dataset, All Hears me, and the real people handwriting dataset which are gathered from KMUTT registered and sentence forms. We use pre-processing step abilities as the same, especially the character segmentation to recognize which character belongs to which model for feeding input into each of the models separately. Eventually, the result of the new solution is better than using only a single model by having fewer classes per model and getting more precision compared to the old solution. So, we provide information on each model below as you can see from the pictures.

Upper Model					
Layer	Type	Feature Maps	Size	Kernel Size	Activation
Input	Input Image	1	28x28		
C1	Convolution	333	28x28	3x3	ReLU
S2	Max Pooling		14x14	2x2	
C3	Convolution	222	12x12	3x3	ReLU
S4	Max Pooling		6x6	2x2	
C5	Convolution	111	4x4	3x3	ReLU
S6	Max Pooling		2x2	2x2	
F7	Fully Connected		256		ReLU
Output	Fully Connected		12		SoftMax

Figure 4.52: Upper Model Hyperparameters

Middle Model					
Layer	Type	Feature Maps	Size	Kernel Size	Activation
Input	Input Image	1	28x28		
C1	Convolution	444	28x28	3x3	ReLU
S2	Max Pooling		14x14	2x2	
C3	Convolution	333	12x12	3x3	ReLU
S4	Max Pooling		6x6	2x2	
C5	Convolution	222	4x4	3x3	ReLU
S6	Max Pooling		2x2	2x2	
F7	Fully Connected		256		ReLU
Output	Fully Connected		54		SoftMax

Figure 4.53: Middle Model Hyperparameters

Lower Model					
Layer	Type	Feature Maps	Size	Kernel Size	Activation
Input	Input Image	1	28x28		
C1	Convolution	333	28x28	3x3	ReLU
S2	Max Pooling		14x14	2x2	
C3	Convolution	200	12x12	3x3	ReLU
S4	Max Pooling		6x6	2x2	
C5	Convolution	100	4x4	3x3	ReLU
S6	Max Pooling		2x2	2x2	
F7	Fully Connected		256		ReLU
Output	Fully Connected		2		SoftMax

Figure 4.53: Lower Model Hyperparameters

Digit model					
Layer	Type	Feature Maps	Size	Kernel Size	Activation
Input	Input Image	1	28x28		
C1	Convolution	444	28x28	3x3	ReLU
S2	Max Pooling		14x14	2x2	
C3	Convolution	333	12x12	3x3	ReLU
S4	Max Pooling		6x6	2x2	
C5	Convolution	222	4x4	3x3	ReLU
S6	Max Pooling		2x2	2x2	
F7	Fully Connected		100		ReLU
Output	Fully Connected		10		SoftMax

Figure 4.54: Digit Model Hyperparameters

For the models, we started performing the model result of the perfect character segmentation at first. Then, we consulted with our supervisor to show and discuss the result. She told us that we need to show the model results for both perfect and imperfect character segmentation. So, we determined to separate the dataset of the test set into two parts to show the model results which are perfect and imperfect character segmentation. By the way, we are going to show from the perfect segment and imperfect respectively. As you can see from the figure below, there is an example of the perfect character segmentation.

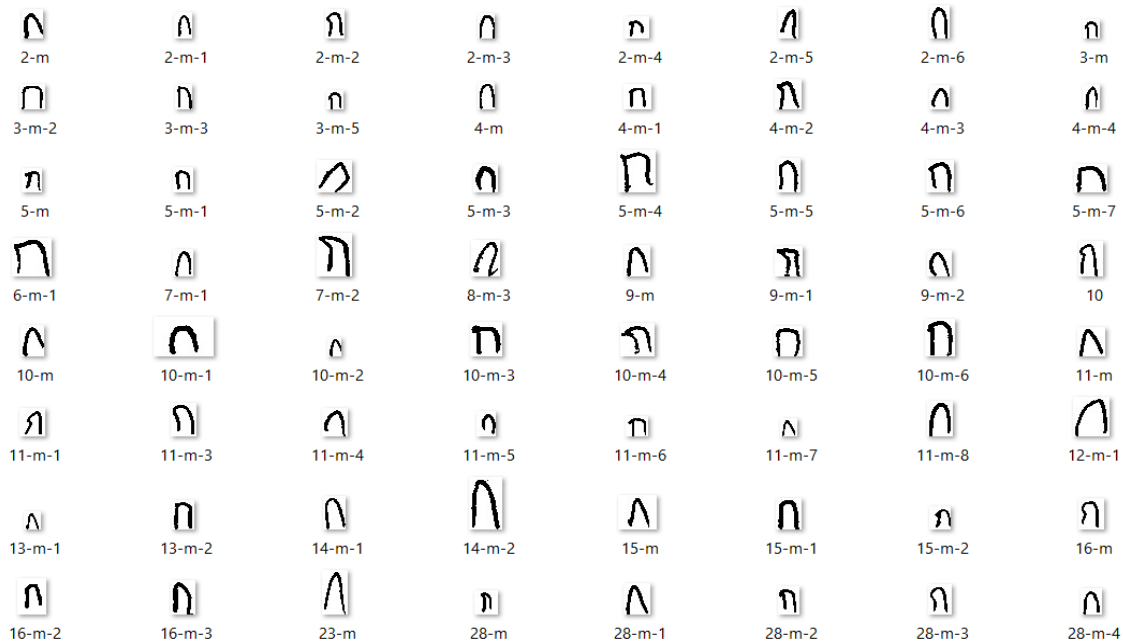


Figure 4.55: The example of the perfect character segmentation

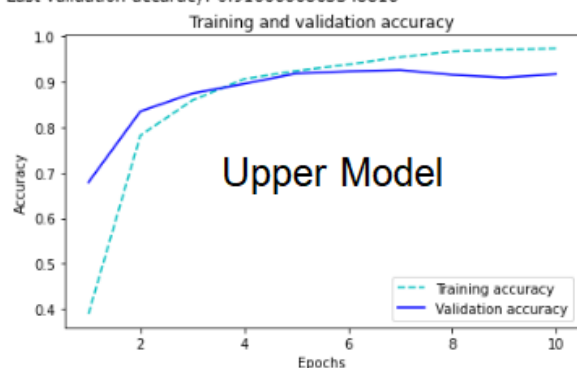
So, from the test dataset above we can get the accuracy, precision, recall, and f-1 score which are included as the result of each model for the perfect character segmentation as below.

Model	Training Accuracy	Validation Accuracy	Test Accuracy
Upper Model	0.9731	0.9166	0.9151
Middle Model	0.9808	0.9211	0.9231
Lower Model	0.9859	0.9814	0.9264
Digit Model	0.9935	0.8817	0.9763

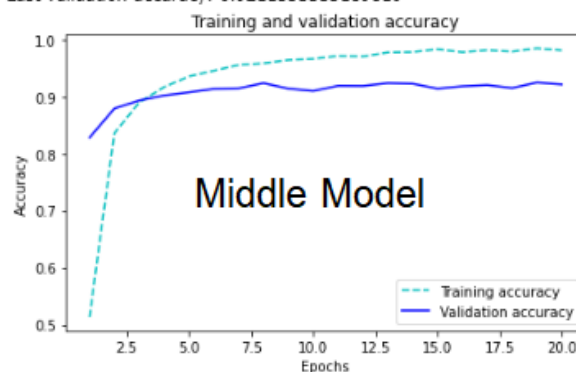
Figure 4.56: Perfect Character segmentation model results

As you can see from Figure 4.56, there are 4-model results provided including a training accuracy, validation accuracy, and test accuracy respectively. So, we are going to show the individual graph of each model which included an epoch and accuracy number for training and validation graphs. On the other hand, the test accuracy evaluated from a confusion matrix will be shown after that.

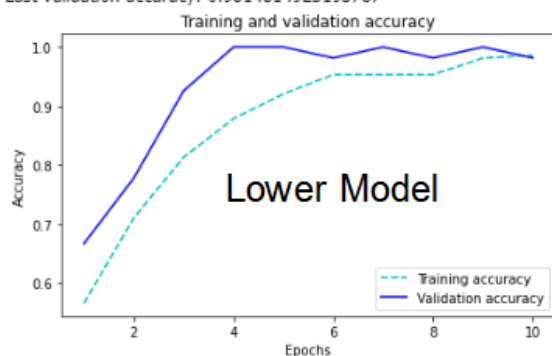
Last train accuracy: 0.9731061
Last validation accuracy: 0.9166666865348816



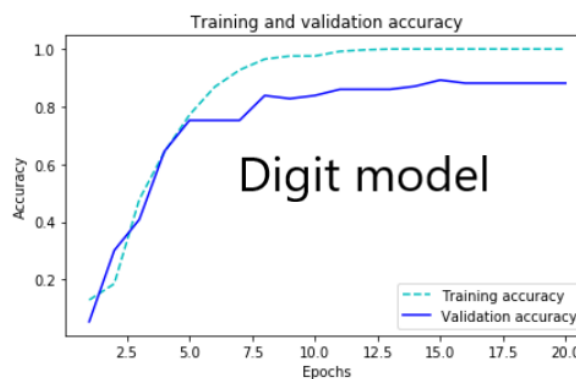
Last train accuracy: 0.9808743
Last validation accuracy: 0.9211553335189819



Last train accuracy: 0.9859813
Last validation accuracy: 0.9814814925193787



Last train accuracy: 1.0
Last validation accuracy: 0.8817204236984253



	precision	recall	f1-score	support
0	0.85	0.97	0.91	79
1	0.93	0.90	0.91	86
2	0.87	0.88	0.87	83
3	1.00	0.84	0.91	70
4	0.80	0.93	0.86	60
5	0.93	0.90	0.92	72
6	0.94	0.97	0.95	74
7	0.94	0.85	0.89	72
8	0.96	0.96	0.96	50
9	1.00	0.97	0.98	64
10	0.92	0.92	0.92	71
11	0.93	0.91	0.92	44
accuracy			0.92	825
macro avg	0.92	0.92	0.92	825
weighted avg	0.92	0.92	0.92	825

Loss: 0.3823084703358737 Accuracy: 0.9151515364646912

Upper Model

	precision	recall	f1-score	support
0	0.94	0.91	0.92	33
1	0.92	0.94	0.93	35
accuracy			0.93	68
macro avg	0.93	0.93	0.93	68
weighted avg	0.93	0.93	0.93	68

Loss: 0.15179296451456406 Accuracy: 0.9264705777168274

Lower Model

	precision	recall	f1-score	support
0	0.94	0.93	0.93	121
1	0.83	0.94	0.88	85
2	0.93	0.90	0.92	63
3	0.87	0.95	0.91	104
4	0.95	0.92	0.94	79
5	0.92	0.95	0.93	74
6	0.98	0.97	0.97	91
7	0.95	0.95	0.95	110
8	1.00	0.99	0.99	71
9	0.83	0.80	0.82	86
10	0.91	0.82	0.86	71
11	0.91	0.99	0.95	73
12	0.96	0.97	0.96	92
13	0.88	0.91	0.89	88
14	0.93	0.87	0.90	76
15	0.99	0.98	0.98	90
16	0.99	0.96	0.97	89
17	1.00	0.94	0.97	94
18	0.99	0.94	0.96	87
19	0.86	0.92	0.89	93
20	0.82	0.91	0.86	95
21	0.96	0.86	0.91	93
22	0.95	0.96	0.95	99
23	0.84	0.94	0.89	79
24	0.97	0.92	0.94	109
25	0.99	0.85	0.91	100
26	0.97	0.97	0.97	92
27	0.88	0.97	0.92	92
28	0.88	0.97	0.93	78
29	0.90	0.96	0.92	90
30	0.99	0.86	0.92	80
31	0.97	0.92	0.94	99
32	0.81	0.96	0.88	109
33	0.96	0.84	0.90	97
34	0.86	0.91	0.88	118
35	0.91	0.85	0.88	87
36	0.91	0.97	0.94	120
37	0.85	0.90	0.88	78
38	0.97	0.94	0.95	125
39	0.98	0.89	0.93	99
40	0.97	0.89	0.93	100
41	0.82	0.94	0.88	103
42	0.89	0.96	0.92	91
43	0.94	0.96	0.95	76
44	0.93	0.93	0.93	87
45	0.95	0.89	0.92	89
46	0.93	0.90	0.92	61
47	0.98	0.97	0.98	66
48	0.99	0.96	0.97	94
49	0.90	0.95	0.92	101
50	0.97	0.74	0.84	53
51	0.97	0.91	0.94	70
52	0.97	0.90	0.94	82
53	0.87	0.95	0.90	55
accuracy			0.92	4804
macro avg	0.93	0.92	0.92	4804
weighted avg	0.93	0.92	0.92	4804

Loss: 0.3985558820486714 Accuracy: 0.9231889843940735

Middle Model

	precision	recall	f1-score	support
0	1.00	0.98	0.99	52
1	1.00	1.00	1.00	40
2	0.97	1.00	0.99	39
3	1.00	0.98	0.99	45
4	0.91	0.98	0.95	44
5	0.97	1.00	0.98	60
6	0.96	0.96	0.96	45
7	0.98	1.00	0.99	50
8	0.97	0.95	0.96	40
9	1.00	0.92	0.96	50
accuracy			0.98	465
macro avg	0.98	0.98	0.98	465
weighted avg	0.98	0.98	0.98	465

Loss: 0.10779679142819938 Accuracy: 0.976344108581543

Digit model

However, for the next figure is an example of the imperfect character segmentation



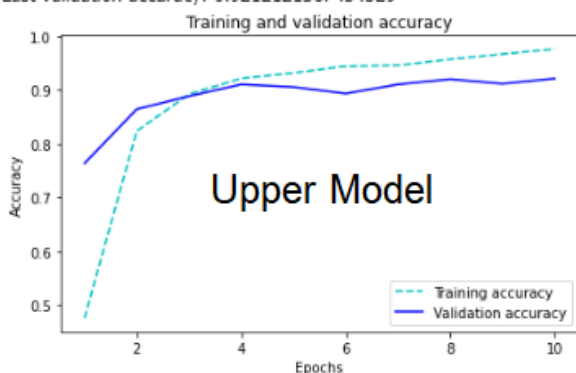
Figure 4.57: The example of the imperfect character segmentation

So, from the test dataset above we can get the accuracy, precision, recall, and f-1 score which are included as the result of each model for the imperfect character segmentation as below.

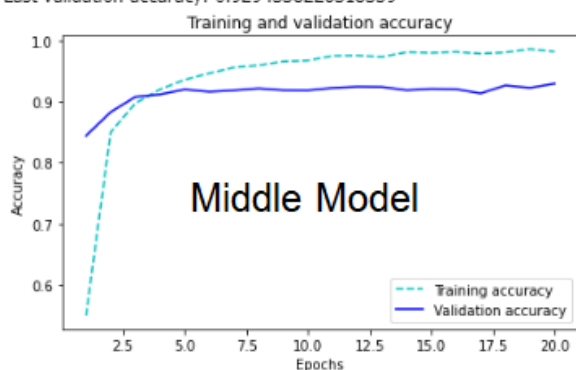
Model	Training Accuracy	Validation Accuracy	Test Accuracy
Upper Model	0.9760	0.9212	0.5638
Middle Model	0.9818	0.9294	0.5469
Lower Model	0.9962	0.9558	0.6000
Digit Model	0.9741	0.8709	0.6052

Figure 4.58: Imperfect character segmentation model results

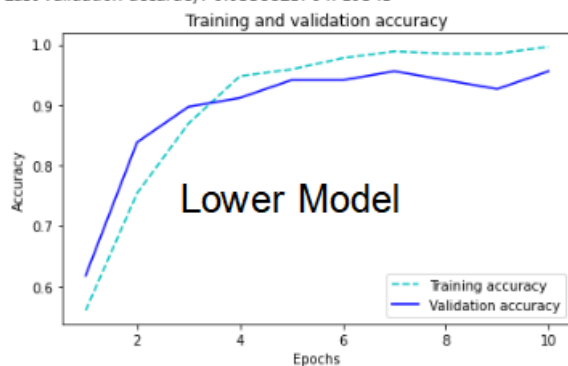
Last train accuracy: 0.9760606
Last validation accuracy: 0.9212121367454529



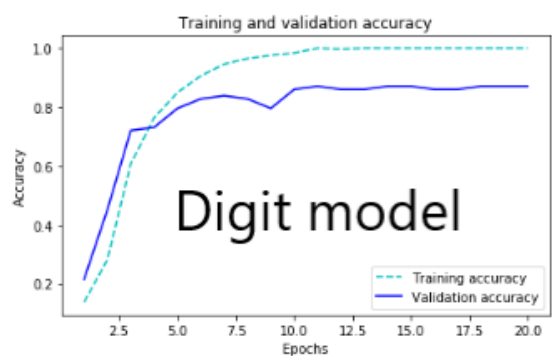
Last train accuracy: 0.9818371
Last validation accuracy: 0.9294338226318359



Last train accuracy: 0.99626863
Last validation accuracy: 0.9558823704719543



Last train accuracy: 1.0
Last validation accuracy: 0.8709677457809



	precision	recall	f1-score	support
0	0.67	0.18	0.29	11
1	0.91	0.70	0.79	46
2	0.75	0.67	0.71	9
3	0.00	0.00	0.00	0
4	0.33	1.00	0.50	1
5	0.00	0.00	0.00	0
6	0.11	1.00	0.20	1
7	0.00	0.00	0.00	1
8	0.00	0.00	0.00	0
9	0.00	0.00	0.00	0
10	0.73	0.46	0.56	24
11	0.00	0.00	0.00	1
accuracy			0.56	94
macro avg	0.29	0.33	0.25	94
weighted avg	0.79	0.56	0.64	94

Loss: 2.843695359027132 Accuracy: 0.563829779624939

Upper Model

	precision	recall	f1-score	support
0	0.88	0.61	0.72	46
1	0.22	0.56	0.31	9
accuracy			0.60	55
macro avg	0.55	0.58	0.52	55
weighted avg	0.77	0.60	0.65	55

Loss: 1.182950782775879 Accuracy: 0.6000000238418579

Lower Model

	precision	recall	f1-score	support
0	0.83	0.69	0.75	87
1	0.07	0.67	0.13	6
2	0.00	0.00	0.00	0
3	0.66	0.42	0.51	50
4	0.00	0.00	0.00	0
5	0.08	0.50	0.13	2
6	0.37	0.93	0.53	27
7	0.44	0.71	0.55	17
8	0.00	0.00	0.00	0
9	0.23	0.41	0.29	27
10	0.11	0.50	0.18	2
11	0.07	1.00	0.13	1
12	0.29	0.40	0.33	10
13	0.00	0.00	0.00	2
14	0.00	0.00	0.00	42
15	0.03	0.04	0.04	26
16	0.00	0.00	0.00	26
17	0.27	0.50	0.35	6
18	0.20	0.80	0.32	15
19	0.67	0.37	0.48	27
20	0.77	0.55	0.64	96
21	0.21	0.50	0.30	6
22	0.64	0.59	0.62	27
23	0.00	0.00	0.00	6
24	0.69	0.62	0.65	47
25	0.33	0.12	0.18	8
26	0.33	0.42	0.37	12
27	0.40	0.67	0.50	3
28	0.00	0.00	0.00	0
29	0.71	0.43	0.54	46
30	0.50	0.33	0.40	3
31	0.80	0.50	0.62	8
32	0.92	0.66	0.77	131
33	0.82	0.30	0.44	30
34	0.82	0.66	0.73	284
35	0.00	0.00	0.00	10
36	0.33	0.50	0.40	30
37	0.00	0.00	0.00	4
38	0.86	0.62	0.72	219
39	0.33	0.56	0.42	9
40	0.81	0.41	0.55	138
41	0.53	0.53	0.53	74
42	0.40	0.75	0.52	8
43	0.07	0.50	0.12	2
44	0.64	0.49	0.56	69
45	0.00	0.00	0.00	0
46	0.00	0.00	0.00	0
47	0.25	0.14	0.18	7
48	0.62	0.79	0.70	99
49	0.49	0.69	0.57	103
50	0.09	0.80	0.16	5
51	0.00	0.00	0.00	1
52	0.00	0.00	0.00	1
53	0.00	0.00	0.00	24
accuracy			0.55	1883
macro avg	0.33	0.39	0.31	1883
weighted avg	0.65	0.55	0.57	1883

Loss: 6.021118085307133 Accuracy: 0.546999454498291

Middle Model

	precision	recall	f1-score	support
0	1.00	0.14	0.25	50
1	0.33	0.68	0.45	19
2	0.82	0.64	0.72	22
3	0.82	0.47	0.60	19
4	0.57	0.90	0.69	29
5	0.71	0.81	0.76	31
6	0.40	0.80	0.53	20
7	0.81	0.65	0.72	26
8	0.80	0.80	0.80	20
9	0.60	0.60	0.60	30
accuracy			0.61	266
macro avg	0.69	0.65	0.61	266
weighted avg	0.72	0.61	0.58	266

Loss: 5.02078025860894 Accuracy: 0.6052631735801697

Digit model

4.2.4 Validation Function

This part is about checking information when a user edits information before clicking to store into a database system. This function is performed to check student id, first name, last name, faculty, department, GPA, class level, classroom, telephone number, and email. The process of validation function is after the program converted the form of the user. So, the user will be able to edit missing information. Then, this function will check the information entered by the user is correct or not because the program requires only the Thai language to enter the information. By the way, we are accepting only emails that could be another language, but the user needs to enter yourself.

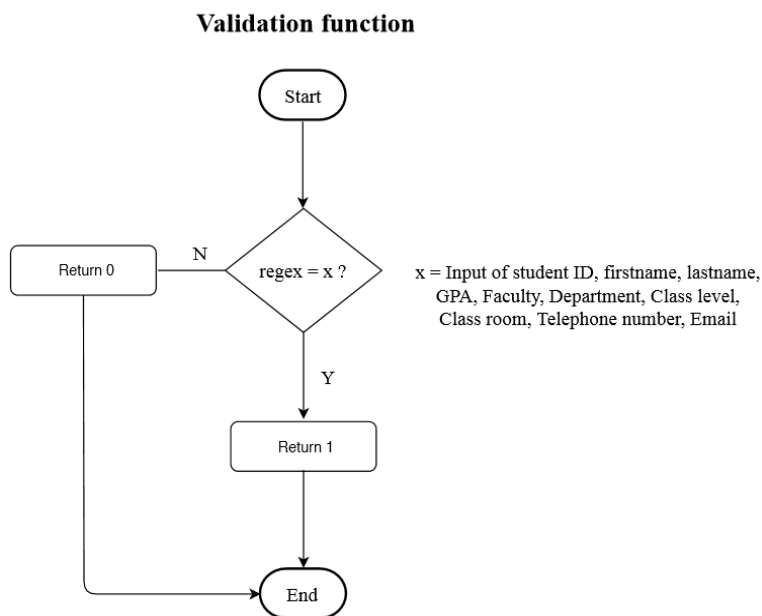


Figure 4.59: Overview flowchart of the validation function

Chapter 5: Conclusion

In this project, we have created a software that can convert Thai handwriting words or sentences to digit texts and then keep it into a storage system. The program consists of a graphic user interface (GUI), converting steps, and storing information, respectively. For the GUI, users can browse a file, review and validate information, and save data in the database system. To convert from handwriting to digital texts, two main functions consist of pre-processing and machine learning. Normalizing an input image to a suitable format before sending it to a machine learning is the main intention of the pre-processing steps. The machine learning model will predict the input from the data pre-processing steps to digital texts. When there is a prediction from handwriting to digital texts, the program will be able to store them into a database system. This whole process helps us to reach our first objective that is to develop efficient Thai handwriting converter software. Our second objective is to study various technologies and the effectiveness of different methods that can be used to develop the software. From studying several papers, there are many technologies that we learned and adapted to utilize our software. Moreover, we have learned a lot about the functionalities of the technologies and how to connect them as a complete program. We noticed that every component from different tools has got their important roles to achieve. Besides, we have learned that some technologies and methods are not suitable to apply to our program. Thus, we need to select only appropriate technologies to use for our software. Unfortunately, due to the Covid-19 situation, we cannot achieve the last objective which is to improve the document management process by eliminating some manual steps and avoiding mistakes in the process that might occur from staff.

During the implementation, we have found two major problems. The first problem occurred in the data preprocessing step which consists of five parts: capturing data, binarization, thinning, slant correction, word segmentation, and character segmentation. All steps provide a good overall result, but two parts are still problems. The main problems of capturing data are noise and scanner. For the scanner problem, different angles of scanning in the scan slot provide different results. Moreover, scanning a document using a different scanner indicates the different results as well. To solve this problem, a user always has to use the same angle of scanning in the slot to receive a good result. Noise after capturing data includes dotted underline, vertical linen, printed character, and small noises. Firstly, we use the horizontal histogram to detect and delete positions of the dotted underline, then apply the vertical histogram to detect and remove the position of the vertical line in only the student ID section. Thirdly, we manually find the position of the printed line and then remove it in the faculty and department section. Last, the small noises on the capped image are eliminated by the morphology technique.

The second problem is the character segmentation. First, the problem is no blank space between characters. Therefore, the algorithm cannot completely segment a sentence into characters. To solve this problem, we have researched many techniques from the previous literature and on the Internet as well as consulted our advisor. We came up with a new idea that uses a concept of quartile to find a threshold that can identify the region or position for making a split between cursive characters. We start with computing the sum of pixels in each vertical line and use the quartile concept to find an appropriate value between the smallest and the median of the computed data. After that, we will run a loop on all data and shift it to detect segment regions. Lastly, the algorithm will estimate the detected line for cutting. However, the overlap of characters and some kind of cursive handwriting problems are still the main obstacles in the character segmentation. Second, a wrong segmentation happens with some of alphabets and vowels that are at a lower level or upper level such as *ay*, *ay*, *il*, *il*, and *ay*. A character segmentation algorithm always splits each level independently. So, some part of the character is classified to a wrong level. To solve this problem, we need to find a new technique that can detect some part of the alphabet and identify it as a middle part.

For the models at the beginning, we used only a single model to predict every class such as vowels, and alphabet. By the way, the result was not good. So, we consulted with our advisor and she suggested separating the models into three level-models which are upper, middle, and lower models. After we have done about building and tuning the models, we found that for the first solution which is using only a single model had too many classes to predict. That is why there were a lot of mistaken predictions. We started gathering more datasets on both the internet and KMUTT registered form to train and tune the hyperparameters of the models for receiving a high accuracy and we did it by getting about 0.92 accuracy. However, that is a prediction from perfect segmented characters on the ideal. We received a recommendation from the advisor again to predict with imperfect segmented characters to measure the model performances by using not only accuracy, but a precision, recall, and the f-1 score should be included. So, we received the result of an overall three-model with imperfect segmented characters about 0.6. On the other hand, there is another model which is a digit model. We do not have any problem because we have an instance that is quite a perfect cut with the number segmentation. Moreover, handwritten patterns of numbers are not complicated compared to letters and vowels. Thus, the digit model is easier than the other models.

Reference

1. Handwritten Text Recognition using Deep Learning
 - <https://pdfs.semanticscholar.org/3339/237110cd5fd3fa8206623e9b740be1d72c9e.pdf>
2. Python GUI – Tkinter
 - <https://www.geeksforgeeks.org/python-gui-tkinter/>
3. Introduction to GUI With Tkinter In Python
 - <https://www.datacamp.com/community/tutorials/gui-tkinter-python>
4. Skew correction
 - <https://www.pyimagesearch.com/2017/02/20/text-skew-correction-opencv-python/>
5. What Is MongoDB?
 - <https://www.mongodb.com/what-is-mongodb>
6. MongoDB
 - <https://searchdatamanagement.techtarget.com/definition/MongoDB>
7. Convolutional neural network
 - <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53>
 - <http://cs231n.github.io/convolutional-networks/>
8. SoftMax
 - <https://www.quora.com/What-is-Softmax-in-CNN?fbclid=IwAR1bSqVKIEHQc01MZSG9aOtD8sMikMTKf4Q9IJpi7JRxPdsWyzLw1CQkr3Q>
9. Categorical Cross-Entropy
 - https://gombru.github.io/2018/05/23/cross_entropy_loss/?fbclid=IwAR1VCH_IWyaehC3XLnunEzl5TBmTZGxF3lUzwZCC0GWh6HSSd9GkbbkUiPCk
10. ReLU
 - <https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>
11. Morphology
 - https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html

- <https://www.cs.auckland.ac.nz/courses/compsci773s1c/lectures/ImageProcessing-html/topic4.htm>
- <https://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm>
- <https://homepages.inf.ed.ac.uk/rbf/HIPR2/matmorph.htm>
- Textbook - Digital image processing second edition. Author: Rafael C. Gonzalez and Richard E. Woods.
- Handwritten Character Segmentation Using Transformation-Based Learning E. Kavallieratou, E. Stamatatos, N. Fakotakis, and G. Kokkinakis [1]

12. Regular expression

- <https://regexr.com/>
- <https://docs.python.org/2.4/lib/re-syntax.html>

13. Alice dataset

- <https://www.ai.rug.nl/~mrolarik/ALICE-THI/>

14. All Hears me dataset

- <https://github.com/AllHearsMe/best2019>