Senior Design Project Report
CSE 499
Bangla Word Detection by Extracting Texts From Images

Submitted By

Name	ID
S. M. Ashiqur Rahman	1510039642
Md. Golam Mortuza Bhuiyan	1511710042
Waseq Ayon	1521061642
Abdullah Al Nauman Shuvo	1510041042

Supervisor

Syed Athar Bin Amir
Lecturer

ELECTRICAL AND COMPUTER ENGINEERING

NORTH SOUTH UNIVERSITY

Summer 2020

Agreement Form

This is to certify that this Project is our original work. No part of this work has been submitted elsewhere partially or fully for the award of any other degree or diploma. Any material reproduced in this project has been properly acknowledged.

Declared By

.....

Name:

ID:

.....

Name:

ID:

.....

Name:

ID:

.....

Name:

ID:

APPROVAL

We take great pleasure in submitting our senior design project report on "**Bangla Word Detection by Extracting Words From Images**". This report is prepared as a requirement of the Capstone Design Project CSE/EEE/ETE 499 A & B which is a two semester long senior design course. This course involves multidisciplinary teams of students who build and test custom designed systems, components or engineering processes. We would like to request you to accept this report as a partial fulfillment of Bachelor Of Science degree under Electrical and Computer Engineering Department of North South University.

Approved By:

Supervisor's Signature

.....
Syed Athar Bin Amir
Lecturer
Department of Electrical and Computer Engineering
North South University, Dhaka, Bangladesh

Department Chair's Signature

.....
Dr. Mohammad Rezaul Bari
Associate Professor
Department of Electrical and Computer Engineering
North South University, Dhaka, Bangladesh

ACKNOWLEDGMENT

First of all, we wish to express our gratitude to the Almighty for giving us the strength to perform our responsibilities and complete the report.

The capstone project program is very helpful to bridge the gap between the theoretical knowledge and real life experience as part of the Bachelor of Science (BSc) program. This report hasn been designed to have practical experience through theoretical understanding.

We would like to express our gratitude towards our supervisor Mr. Athar Bin Amir sir for his instructions and help. Without his instructions and guidelines, we wouldn't be able to reach this point.

We also acknowledge our profound sense of gratitude to all the teachers who have been instrumental for providing us the technical knowledge and moral support to complete the project with full understanding.

Thanks to Banglalekha Isolated and Bangla AI. We used their resources and implemented them in our project.

Also, our heartiest gratefulness to the people who helped and supported us since the very beginning.

ABSTRACT

We will create a model that will be taking images as input and extract the Bangla words from there and translate it into English. The main idea is to recognize Bangla texts from images, extract it from there and translate it to English. We will use the knowledge on OCR, NLP, Google Cloud API, etc to make it happen. This application will be helpful to foreigners who will come to visit Bangladesh but don't know Bangla. They can read any instruction and signs written in Bangla only by taking a photo of the sign and uploading it to our application to get the output.

Table of Contents

List of Figures

CHAPTER 1

OVERVIEW

1.1 Introduction

"**Bangla Word Detection by Extracting Words From Images**" is a ocr based application which will extract Bangla words from images and translate it from Bangla to English using google translation API. The purpose of our project is to create an OCR (Optical Character Recognition) engine which would take an image from the user and recognize the Bangla words from and extract it then translating it for the user. It would serve as a user friendly software which can help a foreigner travelling in Bangladesh. In the running world, there is growing demand for the software systems to recognize characters in computer system when information is scanned through paper documents as we know that we have number of newspapers and books which are in printed format related to different subjects. These days there is a huge demand in "storing the information available in these paper documents in to a computer storage disk and then later reusing this information by searching process"

1.2 Project Details

OCR (optical character recognition) is the use of technology to distinguish printed or handwritten text characters inside digital images of physical documents, such as a scanned paper document. The basic process of OCR involves examining the text of a document and translating the characters into code that can be used for data processing. OCR is sometimes also referred to as text recognition.

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image.

Training data is used to train an algorithm. Generally, training data is a certain percentage of an overall dataset along with a testing set. As a rule, the better the training data, the better the algorithm or classifier performs.

Google Translate API is a free multilingual machine translation service developed by Google, to translate text. It is such an API that helps developers build web based applications that would translate various languages.

1.3 Project Goals

People from all over the world come to visit Bangladesh every year. But moving from one place to another becomes very difficult to move from one place to another. Because all of the signs are in Bengali. So if we can create this application it would become easy for everyone to commute through Bangladesh without the hassle of not understanding what the signs say. So it is a huge opportunity for us to create such an application which would help people from all over the world.

The process of translating the text from the image:

- User enters the image into the model.
- The image is then processed and the text is extracted.
- The extracted text will be shown.

1.4 Summary

In this chapter we discussed the main goals of our project and the basic software that would be implemented in our project. We have described the potential of our project. In the following chapters we will describe our main working process and the tolls that we used. The description will also include the overall structure of our project.

CHAPTER 2

MOTIVATION

2.1 Introduction

In this chapter, we will be discussing the motivations and inspirations of our project. We also talk about the problems which got us thinking of developing such a system.

2.2 Motivation

Our goal is to build an application that will be a lot helpful to foreign people who come to visit our country. Most of the time they face trouble finding the meaning of the street signs and understanding the local notices. Our application will help them to understand those. They will have to just take a picture of what they aren't understanding and put it into our application. Our application will translate the contents into English. This way they can easily understand the signs. Travelers will have no problem understanding road signs and written street notices.

2.3 Summary

From this chapter, our goal, and motivation have been cleared. Provided reasonings are the main motivations of our project.

CHAPTER 3
Purpose

3.1 Introduction

In this chapter we are going to discuss the projects that already exist on text extraction and translation. We shall also talk about the loopholes of the existing system and how we can improve on those shortcomings.

3.2 Systems related to our project

In the running world there is a growing demand for the users to convert the printed documents into electronic documents for maintaining the security of their data. Hence the basic OCR system was invented to convert the data available on papers into computer process-able documents, So that the documents can be editable and reusable. That is the existing system deals with the homogeneous character recognition or character recognition of single languages.Almost all the PDF viewer has a built in OCR in it and there are a lot of online OCR that extracts text from images and translates it. The most common text extracting software are:

- OmniPage Ultimate
- Abbyy FineReader
- Adobe Acrobat Pro DC
- Readiris

Above mentioned OCR are the best available. But not all of them have support for Bangla fonts. Also they are really expensive and only available for the professional.

- PDFelement
- FreeOCR
- i2OCR
- Online OCR

This is a free online OCR available for everyone. But they have less accuracy and minimum support for Bangla fonts.

Due to Bengali language being so diverse it has little support in the OCR industry. So our project can create an OCR only for Bangla fonts. From where we can train our own datasets and not worry about the available resources.

3.3 Proposed Solution

Our projects provide a solution for those who do not have access to the top OCR engines can easily use our software to translate what is written right in front of him without spending any money and which will have a solid Dataset of Bangla fonts.

3.4 Summary

Our project can be considered as a basic software which has a very good potential to fill the lack Bengali OCR. If we can implement our vision completely it will be free to use OCR that would help people from all over the world.

CHAPTER 4
TECHNICAL DESIGN

4.1 Introduction

In this chapter we discuss the basic technical design of our project. By going through the design of the whole system which would make it easier to understand the entire data structure of the system.

4.2 Data flow Process

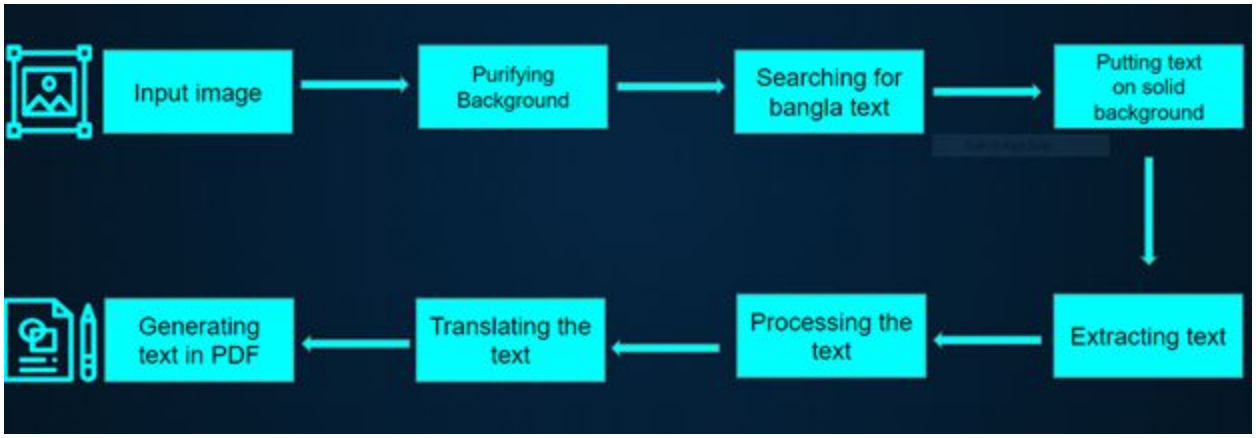


Figure 4.1: Data Flow Diagram

We will first create a database of letters from most used Bangla fonts then train that dataset. It will purify any noise or blurry object in the background. The text detection stage seeks to detect the presence of text in a given image. Then the text will be added in a solid background. Then that text will be extracted which will be translated.

4.3 Summary

As mentioned above, the technical design has enabled us to get a clear picture of how our system is operating. The above data flow diagram will help us to understand our working process.

Chapter 5

Software Description

5.1 Introduction

In this chapter, we will discuss the software and environment we used for this project. The technical description of our project will be discussed in this section. We will discuss the dataset that we first trained to start the project. Then we shall discuss the second dataset we made from "BanglaLekha" . Then we shall discuss the third dataset we made from scratch.

5.2 Dataset

BanglaLekha-Isolated

This dataset, BanglaLekha-Isolated, is a collection of Bangla handwritten isolated character samples. It contains samples of 50 Bangla basic characters, 10 Bangla numerals and 24 selected compound characters. 2000 handwriting samples for each of the 84 characters were collected, digitized and pre-processed. After discarding mistakes and scribbles, 1, 66,105 handwritten character images were included in the final dataset. The dataset also includes information about the age and gender of the subjects from whom the handwriting samples were collected. This information is mapped to each individual image.

Figure 5.1: BanglaLekha-Isolated

BanglaLekha-Isolated(Font Version)

As we had to create a new dataset to train our model we took the whole dataset of BanglaLekha Isolated and made it into a font version.

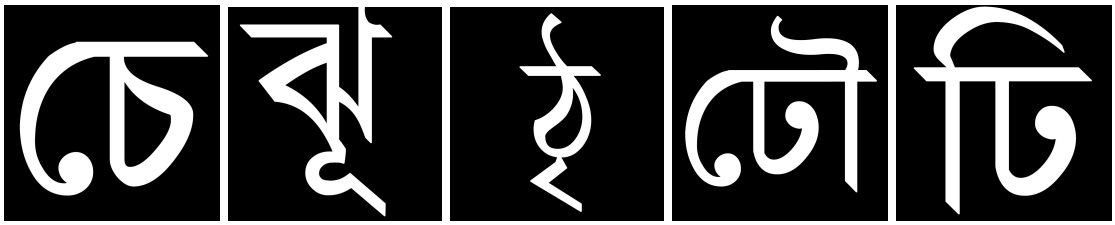


Figure 5.2:BanglaLekha-Isolated(Font Version)

Common Street Signs

We took common street signs as samples and converted the text into digital images. And created three more test datasets over the whole procedure. First test dataset contained almost 2000 data, second test dataset contained 5000 data and Third test dataset contained 4000 data.

Figure 5.3:Common Street Signs(Kalpurush Font)

Figure 5.4:Common Street Signs(Kalpurush Font)

Figure 5.5:Common Street Signs (Mixed Fonts)

5.2 Data training library

· Tensorflow

TensorFlow can train and run deep neural networks for handwritten digit classification, image recognition, word embedding, recurrent neural networks, sequence-to-sequence models for machine translation, natural language processing, and PDE (partial differential equation) based simulations. Best of all, TensorFlow supports production prediction at scale, with the same models used for training.TensorFlow allows developers to create dataflow graphs—structures that describe how data moves through a graph, or a series of processing nodes. Each node in the graph represents a mathematical operation, and each connection or edge between nodes is a multidimensional data array, or tensor. TensorFlow provides all of this for the programmer by way of the Python language. Python is easy to learn and work with, and provides convenient ways to express how high-level abstractions can be coupled together. Nodes and tensors in TensorFlow are Python objects, and TensorFlow applications are themselves Python applications.The single biggest benefit TensorFlow provides for machine learning development is abstraction. Instead of dealing with the nitty-gritty details of implementing algorithms, or figuring out proper ways to hitch the output of one function to the input of another, the developer can focus on the overall logic of the application. TensorFlow takes care of the details behind the scenes.

5.3 Web application

- **Anaconda**

Anaconda is a free and open-source distribution of Python solely used in data science, statistical analysis, and machine learning. It is a large and powerful platform that comprises several packages and is most widely used for science and data-oriented programming. The Anaconda distribution is used by over 13 million users and has over 1500+ open source packages.Anaconda is a distribution of packages built for data science. It comes with conda, a package, and environment manager. We usually used conda to create environments for isolating our projects that use different versions of Python and/or different versions of packages. We also use it to install, uninstall, and update packages in our project environments. When you download Anaconda for the first time it comes with conda, Python, and over 150 scientific packages and their dependencies.Anaconda is popular because it brings many of the tools used in data science and machine learning with just one install, so it's great for having short and simple setup.Like Virtualenv, Anaconda also uses the concept of creating environments so as to isolate different libraries and versions. Anaconda also introduces its own package manager, called conda, from where you can install libraries. Additionally, Anaconda still has the useful interaction with pip that allows you to install any additional libraries which are not available in the Anaconda package manager.

Jupyter Notebook

The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. The Jupyter Notebook App can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet. In addition to displaying/editing/running notebook documents, the Jupyter Notebook App has a “Dashboard” (Notebook Dashboard), a “control panel” showing local files and allowing them to open notebook documents or shutting down their kernels.A notebook integrates code and its output into a single document that combines visualizations, narrative text, mathematical equations, and other rich media. In other words: it's a single document where you can run code, display the output, and also add explanations, formulas, charts, and make your work more transparent, understandable, repeatable, and shareable.

Using Notebooks is now a major part of the data science workflow at companies across the globe. If your goal is to work with data, using a Notebook will speed up your workflow and make it easier to communicate and share your results.Best of all, as part of the open source Project Jupyter, Jupyter Notebooks are completely free. You can download the software on its own, or as part of the Anaconda data science toolkit.

5.4 Summary

In this chapter, we have mentioned the required resources we have used and discussed them.

These are the only resources to train our data and learn about accuracy

Chapter 6
Design Implementation

6.1 Introduction

In this chapter we are going to discuss how we have managed to train our dataset and the result we have achieved.

6.2 Data Training

We trained our first Model with Bangla Lekha Isolated dataset. Then we modified our model and created a digital font version of Bangla Lekha Isolated. After that we created our second model and created three more Test Datasets. We took common street signs as samples and converted the text into digital images. And created three more test datasets over the whole procedure. First test dataset contained almost 2000 data, the second test dataset contained 5000 data and Third test dataset contained 4000 data.

jupyter nayeem Last Checkpoint: 12/12/2019 (unsaved changes)

Logout

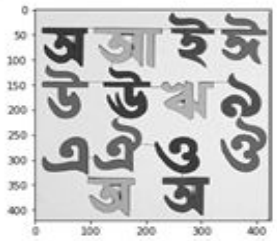
File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3

Run Code

In [3]:

```
import numpy as np
import matplotlib.pyplot as plt
import os
import cv2
from tqdm import tqdm



DATADIR = "/home/nayeem/Downloads/BanglaLekha-Isolated/Images/"

CATEGORIES = [ "85"]

for category in CATEGORIES:
    path = os.path.join(DATADIR,category)
    for img in os.listdir(path):
        img_array= cv2.imread(os.path.join(path,img),cv2.IMREAD_GRAYSCALE)
        plt.imshow(img_array, cmap='gray')
        plt.show()

        break
    break
```



In [2]:

```
print(img_array)
```

```
[[255 218 192 ... 252 255 253]
 [218 212 215 ... 255 252 248]
 [202 214 227 ... 255 255 248]
 ...
 [209 217 219 ... 136 148 144]
 [210 210 212 ... 143 145 145]
 [217 206 206 ... 147 144 146]]
```

Figure 6.1: Training BanglaLekha-Isolated

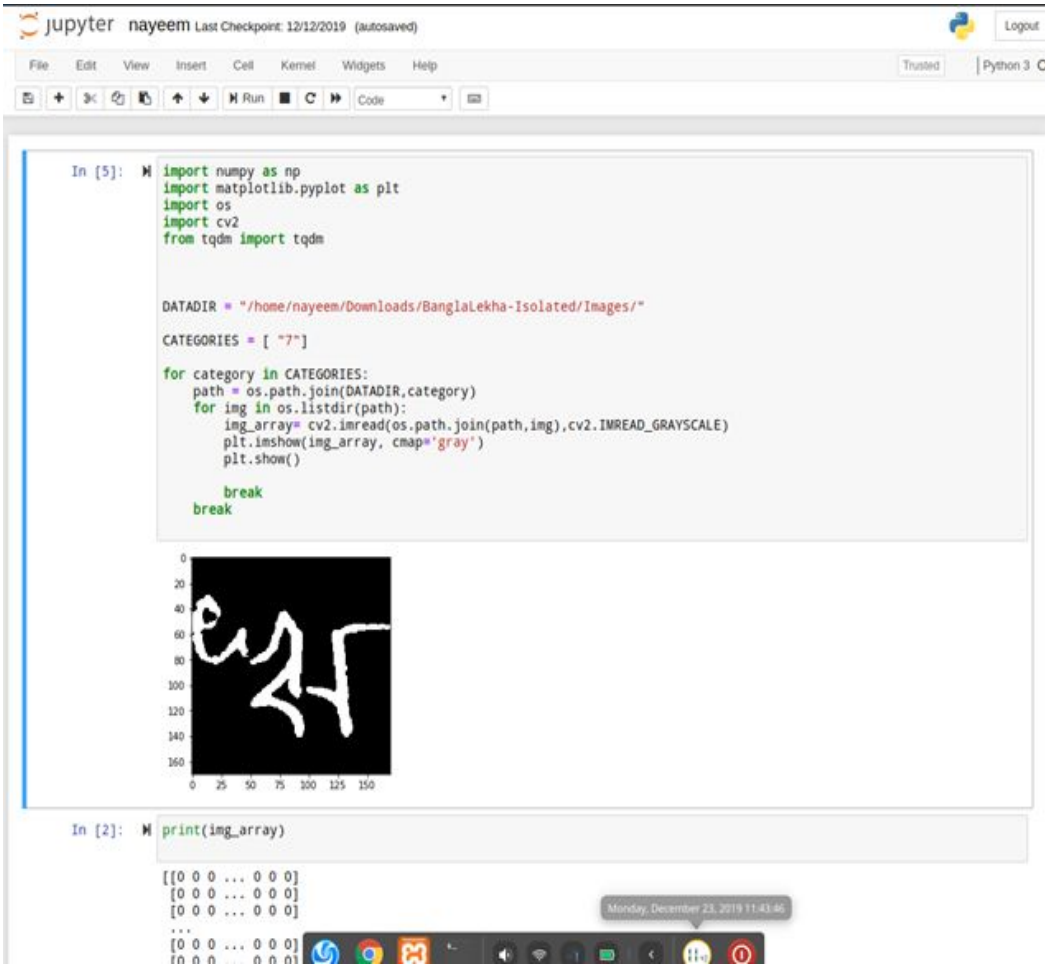


Figure 6.2: Training BanglaLekha-Isolated

We have managed to train our data using the algorithm form . We used Jupyter notebook to complete the process.

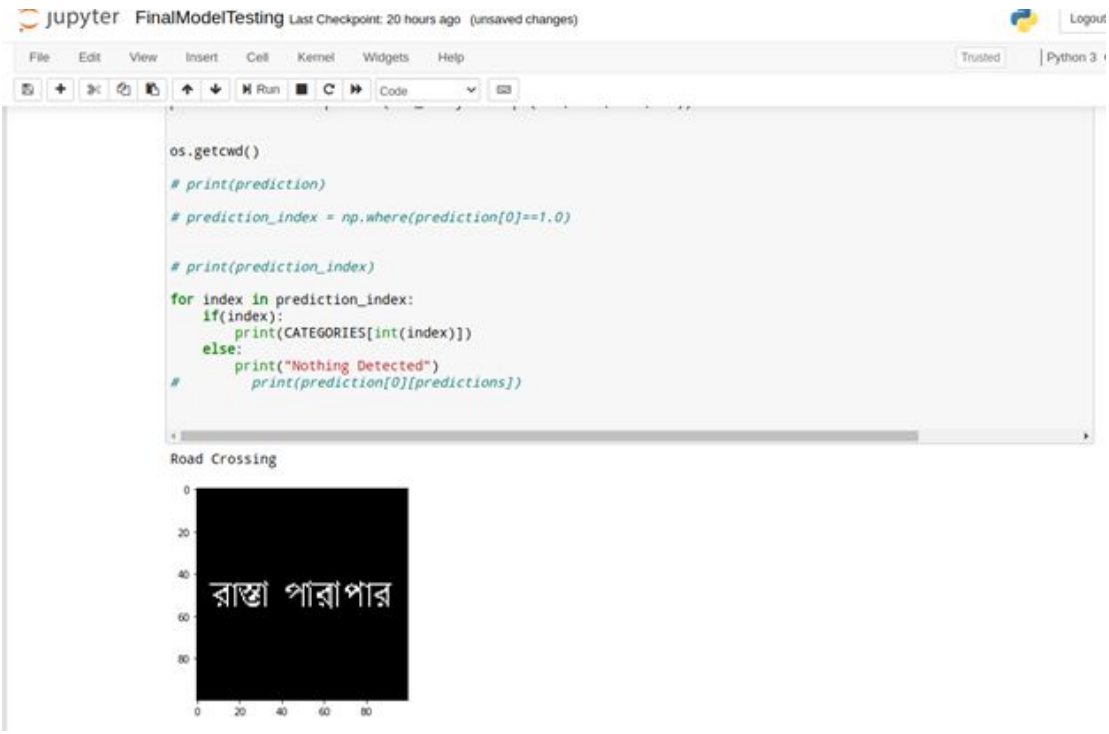


Figure 6.3: Training Road Signs(Kalpurush Font)

Figure 6.4: Training Road Signs(Kalpurush Font)

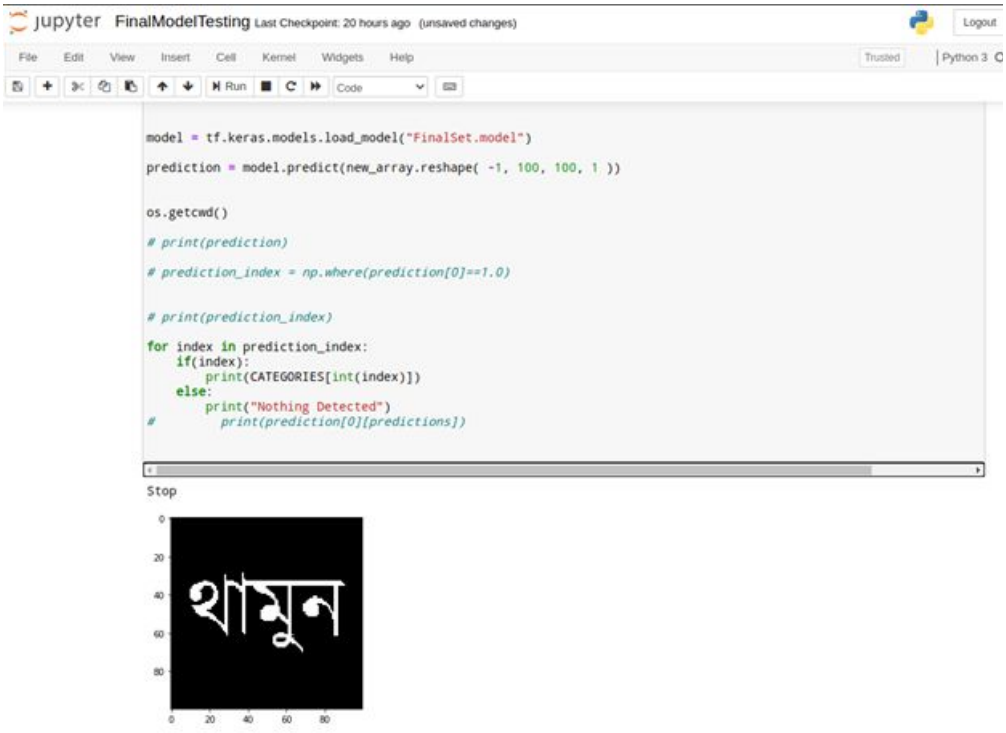


Figure 6.5: Training Road Signs(Kalpurush Font)

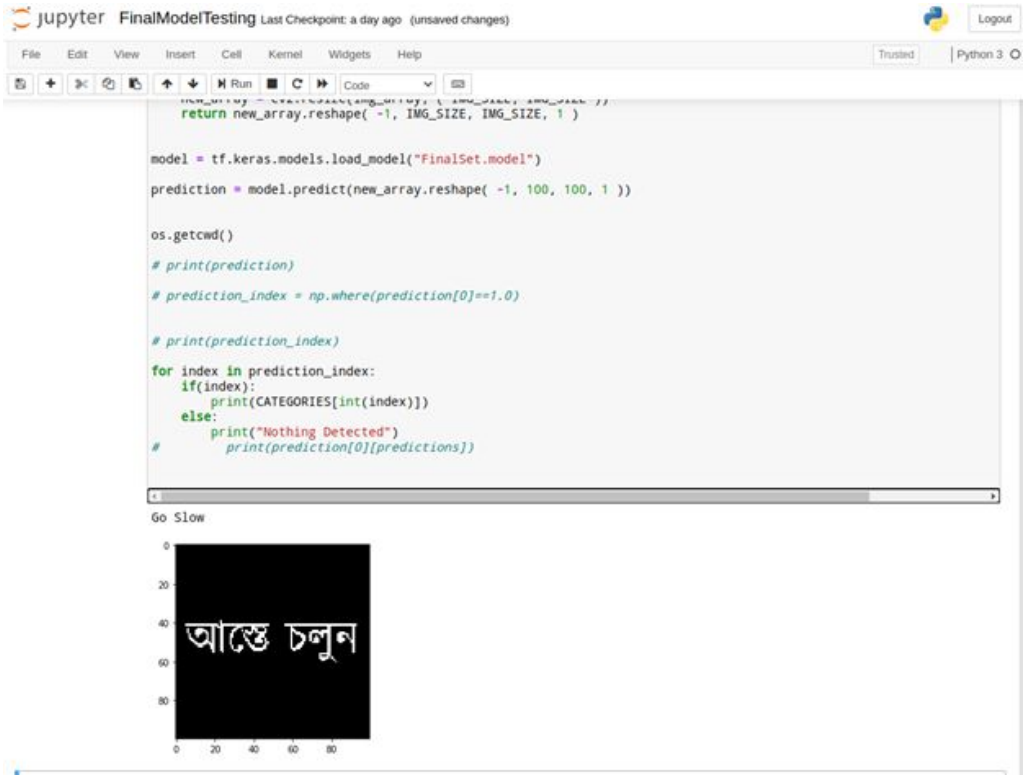


Figure 6.6: Training Road Signs(Kalpurush Font)

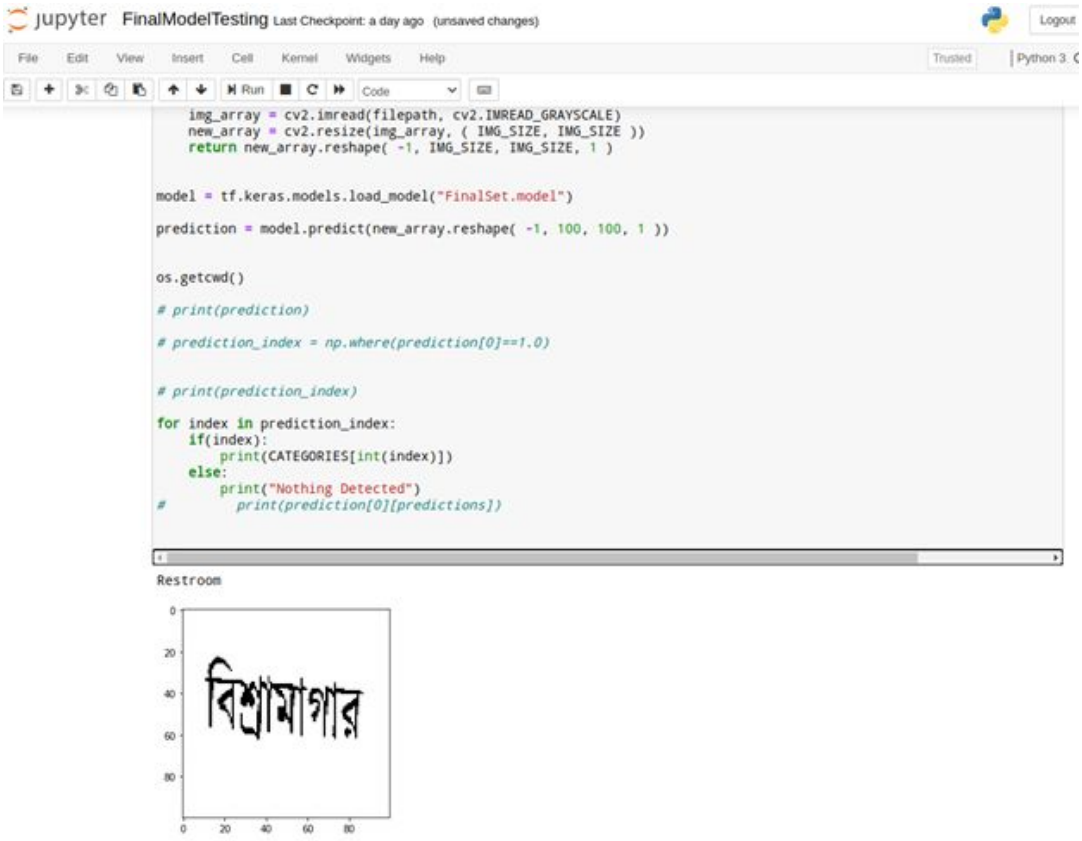


Figure 6.7: Training Road Signs(Kalpurush Font)

6.3 Result

After training our first dataset we have achieved the following result. It was not up to expectation so we created our own dataset for better results. Those datasets performed well but not satisfactory. But our model is working well. So, we hope that if we get enough dataset (which should contain more than 90000 data), we can get a satisfactory result.

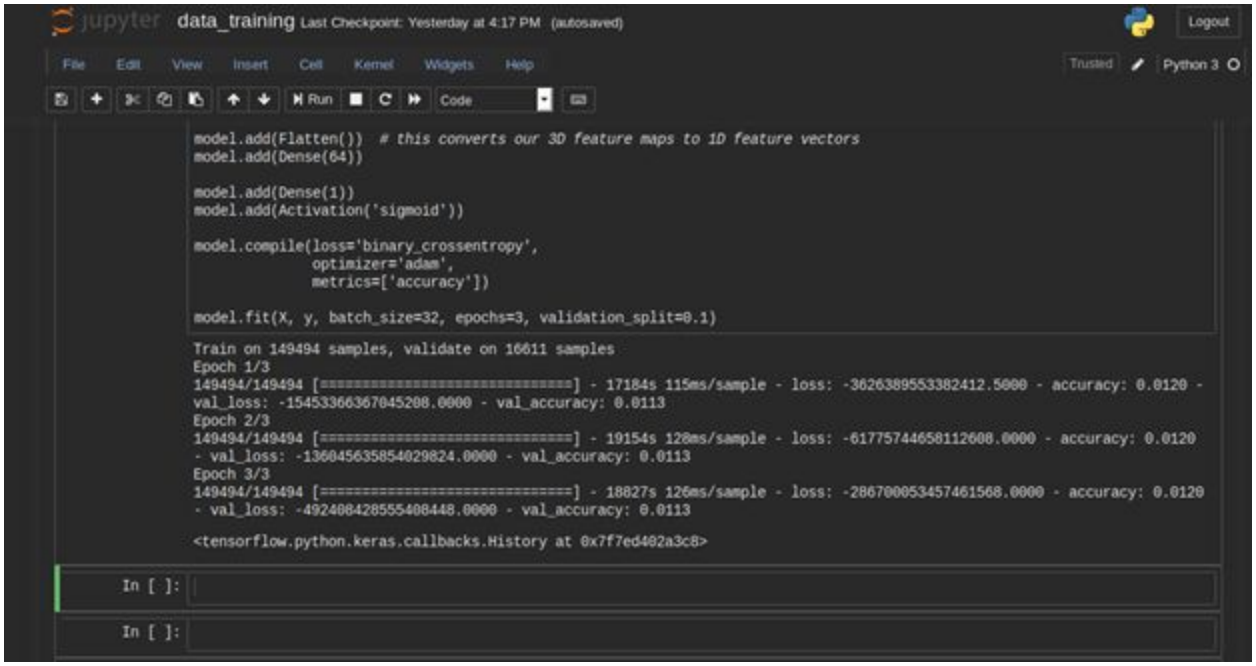


Figure 6.8: Accuracy Result

Chapter 7
Future Works

7.1 Introduction

This chapter discusses the future scope or the implementation of this system. As our system is only in the beginning phase various forms of new features and improvements can be incorporated to this system as per the requirements.

7.2 Future Scope of Work

The main objective of developing this system is to provide a simple solution for the user to translate Bangla language to English. As such projects hardly exist we had too little data to start the project with. So we had to create a dataset based on other projects manually for better results. Which took a lot of time to get perfect. We also had to modify our own model to run the dataset. As we have got a hold of the resource we would like to extend our dataset so that we can get a more perfect result. Moreover the website can be up running after we perfectly get the model training our whole dataset.

7.3 Summary

This chapter has described the possible future applications of the design. But there are a lot of other possibilities to improve our design. The project may need some research for improvements, though the principle of the designed system will remain as it is.

Conclusion

As the image processing site was totally new to us, we needed to research a lot. Also, we had to give a spin to every path that we had found and we did. In example we can say about Matlab, we tried the process in Matlab but as Matlab is an offline platform and a bit complicated, we had to look for new ways. This is how we got introduced with TensorFlow and started to work with it. TensorFlow is less time consuming and based on Python coding. So, it's less complicated. These research and trial and error phases are time consuming. Then we successfully created two models to data train. But we gathered a lot of knowledge out of it. After much trial and error we get to know the limitations and to get over them we made several datasets. We managed to create our own dataset of printed fonts of common street signs which was used to achieve our final result. In this way, we succeeded in training our datasets. But as it was not perfect we didn't achieve our total goals. But, we got familiar with image processing, learned it. We are hopeful that if we manage to get the sufficient and large dataset, we can improve our model which will be then implemented to our proposed web application for actual use.

Thus, development could open up boundless possibilities and new applications that can be very useful and have a great impact on people's lives.

Bibliography

1. Mohammed, Nabeel; Momen, Sifat; Abedin, Anowarul; Biswas, Mithun; Islam, Rafiqul; Shom, Gautam; Shopon, Md. (2017), "BanglaLekha-Isolated", Mendeley Data, v2
2. Pythonprogramming.net
3. Historical review of OCR research and development
4. B.B.Chowdhury and U.Pal, "A complete Bangla OCR System", Computer Vision and Pattern Recognition
5. Bangla optical character recognition through segmentation using curvature distance and multilayer perceptron algorithm
6. Image Processing Basics
7. Python Image Processing (Using OpenCV)
8. Activation Function in Neural Networks
9. Deep Learning Model to Perform Binary Classification
10. Implementing a Neural Network from Scratch in Python

Appendices
Codes for training data

Code for Model 1

Pre-Processing

```
import numpy as np
import matplotlib.pyplot as plt
import os
import cv2
from tqdm import tqdm



DATADIR = "/home/nayeem/Downloads/BanglaLekha-Isolated/Images/"

CATEGORIES = [
"1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20","21","22","2
3","24","25","26","27","28","29","30","31","32","33","34","35","36","37","38","39","40","41","42","43
","44","45","46","47","48","49","50" ]

for category in CATEGORIES:
    path = os.path.join(DATADIR,category)
    for img in os.listdir(path):
        img_array= cv2.imread(os.path.join(path,img))
        img_gray= cv2.cvtColor(img_array, cv2.COLOR_BGR2GRAY)
        (thresh, img_bw) = cv2.threshold(img_gray, 128, 255, cv2.THRESH_BINARY |
cv2.THRESH_OTSU)
        img_bw = cv2.threshold(img_gray, thresh, 255, cv2.THRESH_BINARY)[1]
#         img_bw = cv2.threshold(img_gray, 222, 255, cv2.THRESH_BINARY)
        plt.imshow(img_bw, cmap='gray')
        plt.show()

        break
    break
```

```
print(img_bw)

for row in img_bw:
    for item in row:
        print('%3d ' % item, end="")
print()



IMG_SIZE = 64

new_array = cv2.resize(img_array,(IMG_SIZE, IMG_SIZE))
plt.imshow(new_array,cmap='gray')
plt.show()



training_data = []

def create_training_data():
    for category in CATEGORIES:
        path = os.path.join(DATADIR,category)
        class_num = CATEGORIES.index(category)
        for img in os.listdir(path):
            try:
                img_array= cv2.imread(os.path.join(path,img),cv2.IMREAD_GRAYSCALE)
                new_array = cv2.resize(img_array,(IMG_SIZE, IMG_SIZE))
                training_data.append([new_array, class_num])

            except Exception as e:
                pass

create_training_data()


print(len(training_data))


import random

random.shuffle(training_data)
```

```
X = []

y = []

for features, label in training_data:
    X.append(features)
    y.append(label)

X = np.array(X).reshape(-1, IMG_SIZE, IMG_SIZE, 1)



import pickle

pickle_out = open("X.pickle","wb")
pickle.dump(X, pickle_out)
pickle_out.close()

pickle_out = open("y.pickle","wb")
pickle.dump(y, pickle_out)
pickle_out.close()
```

Model Saving

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
# more info on callbakcs: https://keras.io/callbacks/ model saver is cool too.
from tensorflow.keras.callbacks import TensorBoard
import pickle
import time
```

```
pickle_in = open("X.pickle","rb")
X = pickle.load(pickle_in)

pickle_in = open("y.pickle","rb")
y = pickle.load(pickle_in)

X = X/255.0

dense_layers = [0]
layer_sizes = [50]
conv_layers = [3]

for dense_layer in dense_layers:
    for layer_size in layer_sizes:
        for conv_layer in conv_layers:
            NAME = "{}-conv-{}-nodes-{}-dense-{}".format(conv_layer, layer_size,
dense_layer, int(time.time()))
            print(NAME)

            model = Sequential()

            model.add(Conv2D(layer_size, (5, 5), input_shape=X.shape[1:]))
            model.add(Activation('relu'))
            model.add(MaxPooling2D(pool_size=(2, 2)))

            for l in range(conv_layer-1):
                model.add(Conv2D(layer_size, (3, 3)))
                model.add(Activation('relu'))
                model.add(MaxPooling2D(pool_size=(2, 2)))

            model.add(Flatten())

            for _ in range(dense_layer):
                model.add(Dense(layer_size))
                model.add(Activation('relu'))

            model.add(Dense(50))
```

```
        model.add(Activation('sigmoid'))

        tensorboard = TensorBoard(log_dir="logs/{}".format(NAME))

        model.compile(loss='sparse_categorical_crossentropy',
                      optimizer='adam',
                      metrics=['accuracy'],
                      )

        model.fit(X, y,
                  batch_size=32,
                  epochs=5,
                  validation_split=0.1,
                  callbacks=[tensorboard])

model.save('ShoroBorno.model')
```

Model Testing

```
import cv2
import pandas as pd
import tensorflow as tf
import numpy as np

CATEGORIES =
["অ","আ","ই","ঈ","উ","ঊ","ঋ","এ","ঐ","ও","ঔ","ক","খ","গ","ঘ","ঙ","চ","ছ","জ","ঝ","ঞ","ট
","ঠ","ড","ঢ","ণ","ত","থ","দ","ধ","ন","প","ফ","ব","ভ","ম","য","র","ল","শ","ষ","স","হ","ড়","
ঢ়","য়","ৎ","ং","ঃ","ঁ"]
```

```
def prepare(filepath):
    IMG_SIZE = 64
    img_array = cv2.imread(filepath, cv2.IMREAD_GRAYSCALE)
    new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))
    return new_array.reshape(-1, IMG_SIZE, IMG_SIZE, 1)



model = tf.keras.models.load_model("ShoroBorno.model")

prediction = model.predict([prepare('test.jpg')])



# print(prediction)

prediction_index = np.where(prediction[0]==1.0)

# print(prediction_index)

for index in prediction_index:
    if(index):
        print(CATEGORIES[int(index)])
    else:
        print("Nothing Detected")
#     print(prediction[0][predictions])
```

Code for Model 2

Pre-Processing

```
import numpy as np
import matplotlib.pyplot as plt
import os
import cv2
from tqdm import tqdm



DATADIR = "/home/nayeem/Downloads/Kalpurush/"

CATEGORIES = [ "1", "2", "3" , "4" , "5" , "6" , "7" , "8", "9" , "10" ]

for category in CATEGORIES:
    path = os.path.join(DATADIR,category)
    for img in os.listdir(path):
        img_array= cv2.imread(os.path.join(path,img))
        img_gray= cv2.cvtColor(img_array, cv2.COLOR_BGR2GRAY)
        (thresh, img_bw) = cv2.threshold(img_gray, 128, 255, cv2.THRESH_BINARY |
cv2.THRESH_OTSU)
        img_bw = cv2.threshold(img_gray, thresh, 255, cv2.THRESH_BINARY)[1]
#         img_bw = cv2.threshold(img_gray, 222, 255, cv2.THRESH_BINARY)
        plt.imshow(img_bw, cmap='gray')
        plt.show()

        break
    break


print(img_bw)

for row in img_bw:
```

```
    for item in row:
        print('%3d ' % item, end="")
print()



IMG_SIZE = 100

new_array = cv2.resize(img_array,(IMG_SIZE, IMG_SIZE))
plt.imshow(new_array,cmap='gray')
plt.show()



training_data = []

def create_training_data():
    for category in CATEGORIES:
        path = os.path.join(DATADIR,category)
        class_num = CATEGORIES.index(category)
        for img in os.listdir(path):
            try:
                img_array= cv2.imread(os.path.join(path,img),cv2.IMREAD_GRAYSCALE)
                new_array = cv2.resize(img_array,(IMG_SIZE, IMG_SIZE))
                training_data.append([new_array, class_num])

            except Exception as e:
                pass

create_training_data()



print(len(training_data))
```

```
A = []

b = []


for features, label in training_data:
    A.append(features)
    b.append(label)

A = np.array(A).reshape(-1, IMG_SIZE, IMG_SIZE, 1)


import pickle

pickle_out = open("A.pickle","wb")
pickle.dump(A, pickle_out)
pickle_out.close()

pickle_out = open("b.pickle","wb")
pickle.dump(b, pickle_out)
pickle_out.close()
```

Model Saving

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
# more info on callbakcs: https://keras.io/callbacks/ model saver is cool too.
from tensorflow.keras.callbacks import TensorBoard
import pickle
import time

pickle_in = open("A.pickle","rb")
A = pickle.load(pickle_in)
```

```
pickle_in = open("b.pickle","rb")
b = pickle.load(pickle_in)

A = A/255.0

dense_layers = [2]
layer_sizes = [50]
conv_layers = [3]

for dense_layer in dense_layers:
    for layer_size in layer_sizes:
        for conv_layer in conv_layers:
            NAME = "{}-conv-{}-nodes-{}-dense-{}".format(conv_layer, layer_size, dense_layer,
int(time.time()))
            print(NAME)

model = Sequential()

model.add(Conv2D(64, (5, 5), input_shape=A.shape[1:]))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))


model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(16, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())


# model.add(Dense(10))
# model.add(Activation('relu'))

model.add(Dense(10))
model.add(Activation('softmax'))
```

```
tensorboard = TensorBoard(log_dir="logs/{}".format(NAME))

model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics=['accuracy'])

model.fit(A, b,batch_size=5,epochs=3,validation_split=0.1,callbacks=[tensorboard])

model.save('FinalSet.model')
```

Model Testing

```
import cv2
import pandas as pd
import tensorflow as tf
import numpy as np
import os
import matplotlib.pyplot as plt

CATEGORIES = [ "Go Right", "No Parking" , "Stop",  "Go Slow", "Road Crossing", "School Ahead",  "Closed Road",  "Restroom",  "Waiting Room",  "Bus Stop"]



img_array= cv2.imread('T74.png')



img_gray= cv2.cvtColor(img_array, cv2.COLOR_BGR2GRAY)
(thresh, img_bw) = cv2.threshold(img_gray, 128, 255, cv2.THRESH_BINARY | cv2.THRESH_OTSU)
img_bw = cv2.threshold(img_gray, thresh, 255, cv2.THRESH_BINARY)[1]
```

```
new_array = cv2.resize(cv2.bitwise_not(img_bw), (100, 100))
#   img_bw = cv2.threshold(img_gray, cv2.bitwise_not(ig_bwmHRESH_BINARY)
plt.imshow(new_array, cmap='gray')
#plt.show()

def prepare(filepath):
    IMG_SIZE = 100
    img_array = cv2.imread(filepath, cv2.IMREAD_GRAYSCALE)
    new_array = cv2.resize(img_array, ( IMG_SIZE, IMG_SIZE ))
    return new_array.reshape( -1, IMG_SIZE, IMG_SIZE, 1 )


model = tf.keras.models.load_model("FinalSet.model")

prediction = model.predict(new_array.reshape( -1, 100, 100, 1 ))


os.getcwd()

# print(prediction)

# prediction_index = np.where(prediction[0]==1.0)


# print(prediction_index)

for index in prediction_index:
    if(index):
        print(CATEGORIES[int(index)])
    else:
        print("Nothing Detected")
#         print(prediction[0][predictions])
```

----- The End -----