Drishti

Major Project Report

Submitted By:

Neetish Kumar	(18103009)
Ankit Goyal	(18103018)
Bishwash Pokhrel	(18103106)
Jaspreet Singh	(18103107)

Under the Supervision of:

Prof. Poonam Saini

Faculty

Department of Computer Science and Engineering

Punjab Engineering College

(Deemed to be University)

Chandigarh

DECLARATION

We hereby declare that the project work entitled “Drishti” is an authentic record of our own work carried out at Punjab Engineering College (Deemed to be University), as a requirement of Major Project for the award of degree of B. Tech (Computer Science and Engineering), under the guidance of Prof. Poonam Saini (Faculty, Department of Computer Science and Engineering) during August to December 2021.

Neetish Kumar, 18103009

Ankit Goyal, 18103018

Bishwash Pokhrel, 18103106

Jaspreet Singh, 18103107

CERTIFICATE

This is to certify that the project entitled Drishti by Ankit Goyal, Neetish Kumar, Bishwash Pokhrel & Jaspreet Singh is an authentic record of our work carried out under the supervision of Prof. Poonam Saini, Faculty, Computer Science and Engineering Department, Punjab Engineering College (Deemed to be University), Chandigarh in fulfilment of the requirements as a part of Major Project for the award of 02 credits in semester 7 of the degree of Bachelor of Technology in Computer Science and Engineering.

Certified that the above statement made by the students is correct to the best of my knowledge and belief.

Prof. Poonam Saini
(Faculty Mentor)
Department of Computer Science and Engineering
Punjab Engineering College
(Deemed to be University)
Chandigarh

Ankit Goyal
18103018

Neetish Kumar
18103009

Bishwash Pokhrel
18103106

Jaspreet Singh
18103107

Dated:

ACKNOWLEDGEMENT

We have taken a lot of deliberations in this venture. But it wouldn't have been possible without the help and backing of numerous people. We want to extend our true appreciation and thank them. We take this opportunity to express our profound gratitude and deep regards to our mentor Prof. Poonam Saini for her exemplary guidance, monitoring and constant encouragement throughout the course of this project.

This project truly wouldn’t have been possible without her mentorship.

ABSTRACT

The challenges faced by blind people and visually impaired in their everyday lives are not well understood. They confront a number of visual challenges every day – from reading the denomination on a currency note to figuring out if there is an obstacle in front of them. There are very limited braille printed books in public libraries and in schools/colleges, which poses a great issue as it puts a constraint on them being educated. Further, The lack of support for them, the limited accessibility to activities and information, the societal stigma and the lack of unemployment, are all factors frequently leading blind or low vision individuals in isolation. We aim to make this world a better place to live in for the blind and the visually impaired. Our project aims to help a visually impaired or a blind person in their everyday life tasks such as Recognizing Currency Notes, reading text from live images, and detecting obstacles during navigation.

Figure 1: Visually impaired people

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Our project aims to help a visually impaired or a blind person in their everyday life tasks such as Recognizing Currency Notes, reading text from live images, and detecting obstacles during navigation. Problems faced by visually impaired in performing daily activities are in great number. They are unable to recognize the paper currencies due to similarity of paper texture and size between different categories. Navigation and text reading (book reading, newspaper reading, etc) are some of the other major problems faced by them. Therefore, a system could be designed that could actually take in live video input from the person and conveys the corresponding audio output for the visually impaired, which will really help them in their day to day chores.

1.1.1 Mobile Application

A mobile application, most commonly referred to as an app, is a type of application software designed to run on a mobile device, such as a smartphone or tablet computer. Mobile applications frequently serve to provide users with similar services to those accessed on PCs. Apps are generally small, individual software units with limited function. The advantage of such an application is that it can be accessed from anywhere and is an anytime application- i.e., it is always available, regardless of the location of the user or other factors. There are many frameworks, languages, tools and paths to create a mobile application. Drishti uses React Native and JavaScript for the front-end and Flask, a Python framework, and GraphQL for the server-side scripting needs. React Native lets us build both Android as well as IOS apps simultaneously, which basically converts the React Native code to their respective native code.

1.1.2 What is Drishti

We aim to help a blind or visually impaired person by helping them in their everyday activities such as recognizing currency notes, informing them about obstacles during navigation, and reading out text from a video input, which can be of a book, newspaper or text written on a product.

Thus, We've developed these features by applying Neural Network on it. The tools used for developing the project are Python, React Native, JavaScript, Flask and GraphQL.

- A mobile application incorporating deep learning techniques.
- Mobile App features:
    - Currency Note Recognition
    - Text to Speech from a video feed
    - Local Navigation Aide
- Output: Voice output for guiding the users.
- Easy to use mobile app for the users.
- An additional feature which we plan to implement is smart glasses, which incorporate all the above mentioned features in wearable glasses.

Figure 2: Drishti App

CHAPTER 2

PROBLEM STATEMENT

Problem Statement Formulation

Blindness does come with its share of challenges, though, considering most of the world is designed with sighted people in mind. As a result, living with limited vision means blind people have to find other ways to handle some things most people take for granted. They confront a number of visual challenges every day – from reading the denomination on a currency note to figuring out if there is an obstacle in front of them. There are very limited braille printed books in public libraries and in schools/colleges, which poses a great issue as it puts a constraint on them being educated.

In such a situation, an app which could assist the blind or the visually impaired person would be of great help. There's a need for a system that could take in live video input from the person and conveys the corresponding audio output for the visually impaired, which will really help them in their day to day chores.

- **What already exists:**
  - Currency Note Recognition system (for USD only)
  - PDF text to speech
  - Voice assistants for answering questions
- **Existing Problems:**
  - No Currency Note Recognition system (for INR)
  - No Local Navigation assistant
  - No text to speech from a video feed for blind

- **Our Contribution:**
  - Created dataset for Currency Note Recognition for Indian Rupee(INR).
  - Trained Currency note recognition model using Convolutional Neural Networks.
  - Real time predictions giving currency denominations as output.

Following is our approach divided into different phases about the solution of the problem.

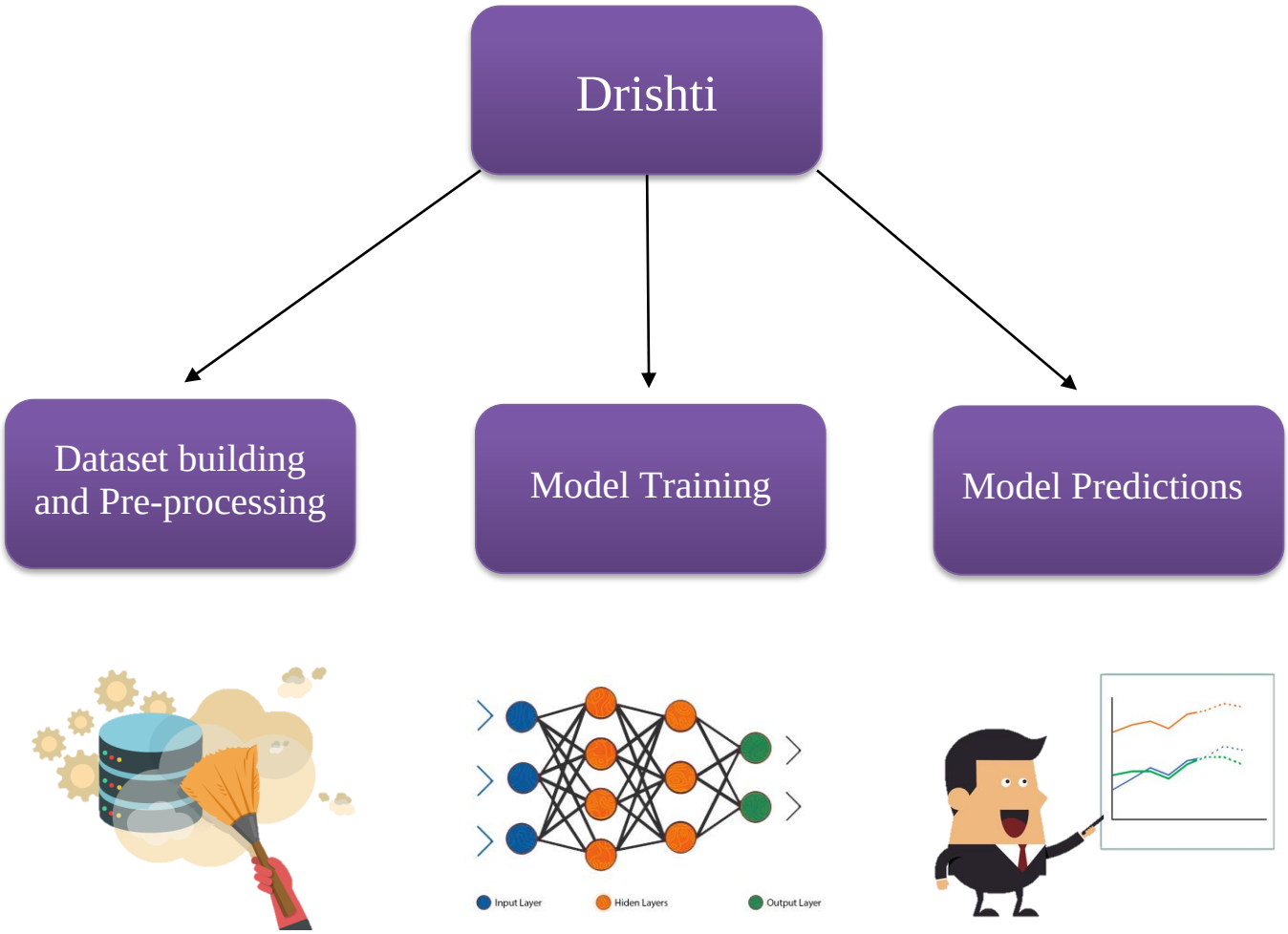


Figure 3: Phases of Project

CHAPTER 3

PROPOSED WORK

(ALGORITHM/MODEL/APPROACH)

3.1 Algorithm & Approach

1. Found small datasets from given websites:
    - Kaggle:
        a. Indian Currency Note images dataset 2020
        b. Indian currency notes
    - Mendeley Data – Indian Currency dataset
2. Preprocessing Dataset:
    - Filtering out irrelevant images from the dataset.
    - Correcting the wrong labels where currency denomination notes are not correctly labeled.
    - Merging datasets into one final dataset.
    - Splitting the dataset into training, validation and testing. Finally left with 8460 images

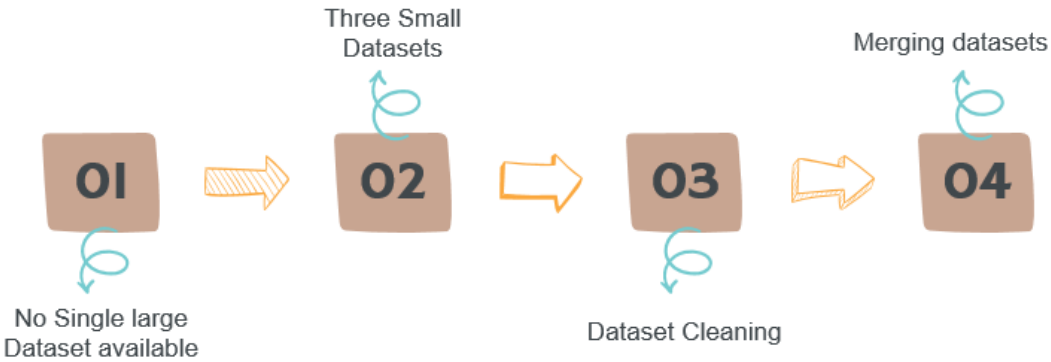


Figure 4: Dataset creation

3. Requirements of the Currency Note Recognition feature:
    - High Accuracy – since our feature involves financial transactions
    - Lightweight Model – for faster predictions
4. We have used Convolutional Neural Networks (CNNs) with transfer learning. We performed the following steps:
    - Selected a pre-trained base model namely EfficientNet, which is trained over millions of images of imagenet dataset
    - We removed the last few layers of the architecture and added our custom fully connected layers to it.
    - We finally froze the parameters of our base model layers.

Figure 5: Transfer learning

5. We have tried using the following base models during our training phase:
    - InceptionV3
    - VGG16
    - EfficientNet
6. Applying Image augmentation is a really part for our feature, since a visually impaired person might hold a currency note in any orientation. So before starting the training phase, we applied the following augmentations to our images while reading the images from the directory:

- Flipping images
- Rotations
- Shifts
- Shear
- Zoom

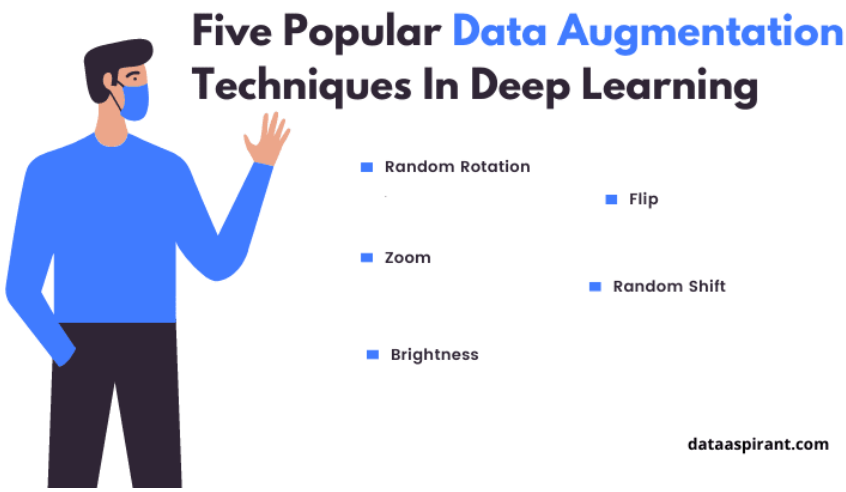


Figure 6: Data Augmentation Techniques

7. We have used different optimizers and learning rates:
    - Optimizers: Adam, RMSProp & SGD (Stochastic Gradient Descent).
    - Learning rate: ranging from 0.0001 to 0.001
8. After completing all the above steps, we started the training process.
9. We were successfully able to achieve a best validation accuracy of 99.81% and training accuracy of 99.50% using Adam Optimizer, sparse_categorical_crossentropy as the loss function and 0.0001 as the Learning Rate.
10. Once our deep learning model is ready, we need to embed our model in the mobile application.
11. For the Mobile application part, we have used React Native, as it helps us develop multiple apps simultaneously:
    - IOS Apps
    - Android Apps

Figure 7: React Native

12. Our app currently only has three basic windows:

- Splash Screen: the loading window of our app
- Home Page: It currently only has a button which opens the camera for us to take the video stream as input for currency recognition.
- Camera window: On the camera window, we do have a close camera button which takes us back to the homepage.

Figure 8: Drishti App

3.2 Model

The system aims to help a blind or visually impaired person by helping them in their everyday activities such as recognizing currency notes, informing them about obstacles during navigation, and reading out text from a video input, which can be of a book, newspaper or text written on a product.
The Design requirements for the project are:

3.2.1 Knowledge Requirement for Modelling

a) Knowledge of App Development: To develop the application with the currency note recognition feature, local navigation assistant and text to speech from a video feed, the knowledge of App Development was required. The team has knowledge of App Development, and Drishti uses React Native and JavaScript for the front-end and Flask, a Python framework, and GraphQL for the server-side scripting needs.
b) Knowledge of Deep Learning: Since the application required designing a model which could help predict currency notes, the knowledge of Deep learning was required. Our team had knowledge of Neural networks, and our application specifically used CNN's (Convolutional Neural Network) and Transfer learning.
c) Knowledge of Needs of Visually Impaired: Knowledge of the actual needs of the blind and the visually impaired was really essential. The scope of this project was defined by meeting the teachers of Institute for The Blind, Sector 26.

3.2.2 Modelling the Architecture

a) Server: The system requires servers to process applications and to host databases, to and from which the querying is done. For development purposes the team has used their local machines.
b) App Hosting: The application and databases are required to be hosted on the web, to provide anytime access to them. For development, the team has used the localhost to host applications.
c) IDEs: Since various technologies are being used, specialized IDEs and tools are used for them. The team has used:
    I. VS Code and Atom for Mobile App Development.
    II. Jupyter Notebook and Google Colab for Development of Python Script.

CHAPTER 4

IMPLEMENTATION

4.1 MOBILE APPLICATION

4.1.1 Splash Screen

The mobile app first opens up the splash screen, which is the first screen when the user opens the app. Drishti is seen written on this page, along with an image denoting visually impaired people.

Figure 9: Drishti Splash Screen

4.1.2 Drishti Homepage

This page contains a button which says “Recognize Currency” and when we press the button, the camera opens. If the app doesn’t have camera permissions, it requests for the same from the user. The page currently looks like as shown in below figure.

Figure 10: Drishti homepage

Figure 11: Drishti currency Recognition

4.1.3 Recognize Currency window:

After we press the button present on the which says “Recognize Currency” and when we press the button, the camera opens. We plan to take camera video feed from here, and then send it to our backend for the further processing of input and give the corresponding output value as audio. The screen looks like as shown in the figure above.

4.2 LANGUAGES AND OTHER TOOLS USED

4.2.1 React Native

Figure 12: React Native

React Native is a JavaScript framework for writing real, natively rendering mobile applications for iOS and Android. It's based on React, Facebook's JavaScript library for building user interfaces, but instead of targeting the browser, it targets mobile platforms. In other words: web developers can now write mobile applications that look and feel truly "native," all from the comfort of a JavaScript library that we already know and love. Plus, because most of the code you write can be shared between platforms, React Native makes it easy to simultaneously develop for both Android and iOS. React Native currently supports both iOS and Android, and has the potential to expand to future platforms as well.

4.2.2 Flask

Figure 13: Flask

Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common frameworks related tools.

4.2.3 Tensorflow

Figure 14: Tensorflow

TensorFlow is a Python library for fast numerical computing created and released by Google.
It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow. TensorFlow is an open source library for fast numerical computing. It was created and is maintained by Google and released under the Apache 2.0 open source license. The API is nominally for the Python programming language, although there is access to the underlying C++ API.

4.2.4 GraphQL

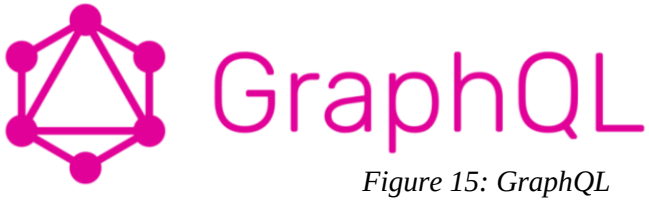


Figure 15: GraphQL

GraphQL is a query language and server-side runtime for application programming interfaces (APIs) that prioritizes giving clients exactly the data they request and no more. GraphQL is designed to make APIs fast, flexible, and developer-friendly. It can even be deployed within an integrated development environment (IDE) known as GraphiQL. As an alternative to REST, GraphQL lets developers construct requests that pull data from multiple data sources in a single API call.
Additionally, GraphQL gives API maintainers the flexibility to add or deprecate fields without impacting existing queries. ways to clients.

.

4.2.5 Sublime Text

Sublime Text is a shareware cross-platform source code editor with a Python application programming interface (API). It natively supports many programming languages and markup languages, and functions can be added by users with plugins, typically, community-built and maintained under free-software licenses.

Figure 16: Sublime text

4.2.7 JUPYTER NOTEBOOK

Figure 17: Jupyter Notebook

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, statistical modeling, data visualization, machine learning, and much more. We used Jupyter Notebook for writing code, compiling it and getting the results in modern fashion.

4.2.8 Atom

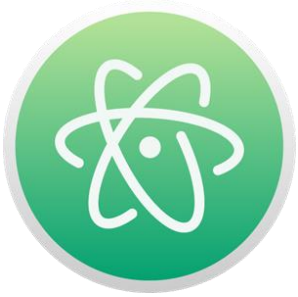


Figure 18: Atom

Atom is a free and open-source text and source code editor for macOS, Linux, and Microsoft Windows with support for plug-ins written in JavaScript, and embedded Git Control, developed by GitHub. Atom is a desktop application built using web technologies. Most of the extending packages have free software licenses and are community-built and maintained. Atom is based on Electron (formerly known as Atom Shell), a framework that enables cross-platform desktop applications using Chromium and Node.js. It is written in CoffeeScript and Less.

4.3 ANALYSIS OF DATASET

Found small datasets from given websites:

1. Kaggle:
    - Indian Currency Note images dataset 2020
    - Indian currency notes
2. Mendeley Data:
    - Indian Currency dataset

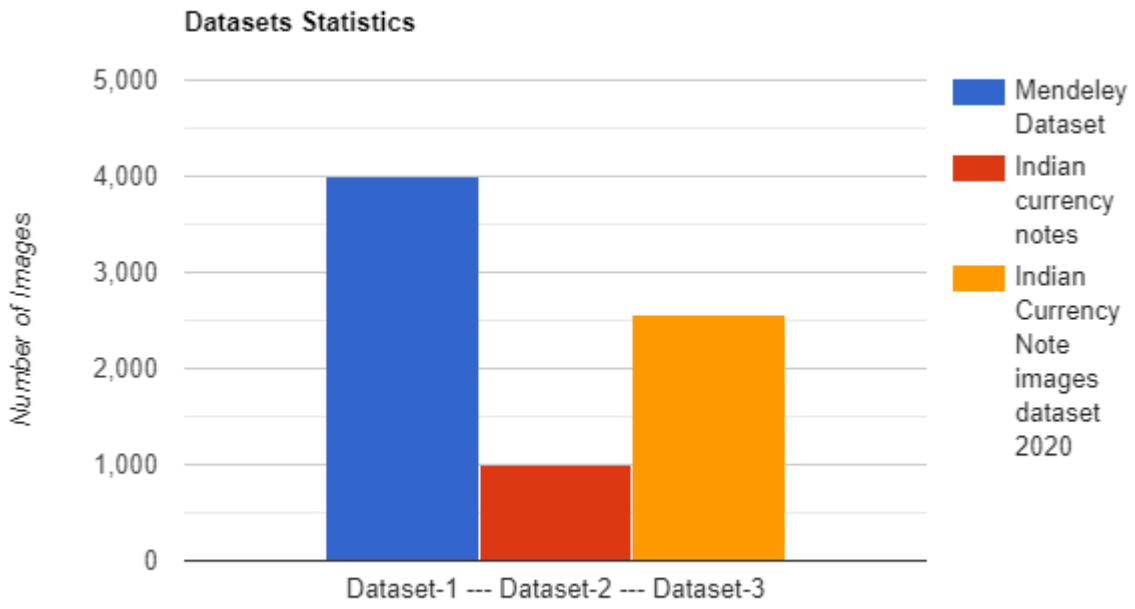


Figure 19: Dataset Statistics

Statistics Analysis:

The division of the datasets is as follows:

1. Total number of Image examples: 8460
2. Total number of Images in the training set: 7833
3. Total number of Images in the validation set: 536
4. Total number of Images in the testing set: 91

The Pie chart for the same is shown below:

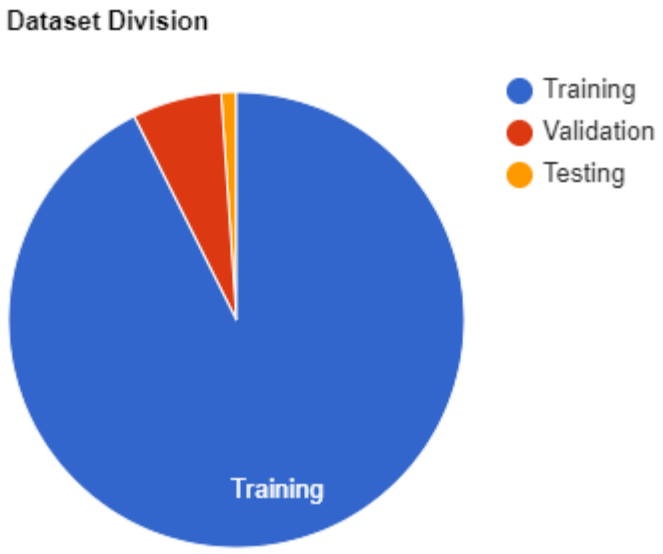


Figure 20: Dataset division

Dataset Class Analysis:

Datasets had the following classes of currency denominations:

- 10
- 20
- 50
- 100
- 200
- 500
- 2000
- Background

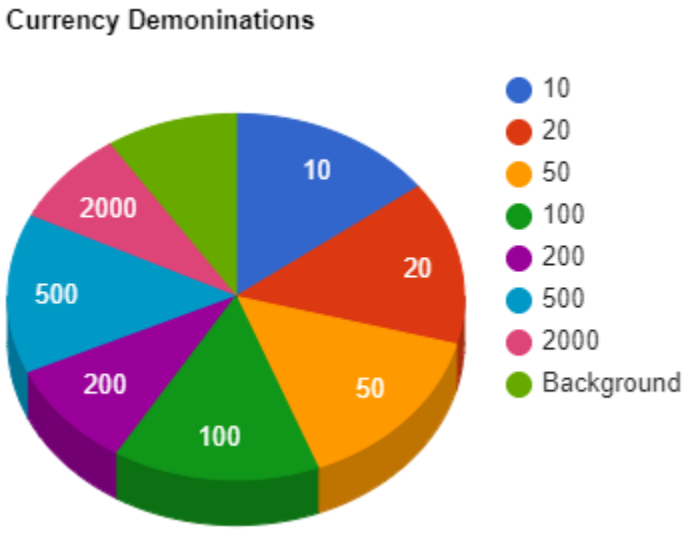


Figure 21: Dataset Class Analysis

CHAPTER 5

RESULTS AND DISCUSSION

5.1 RESULTS

The following is the structure of our trained neural network:

- Base EfficientNet Input Layer
- EfficientNet Model architecture
- Final custom Fully Connected layers

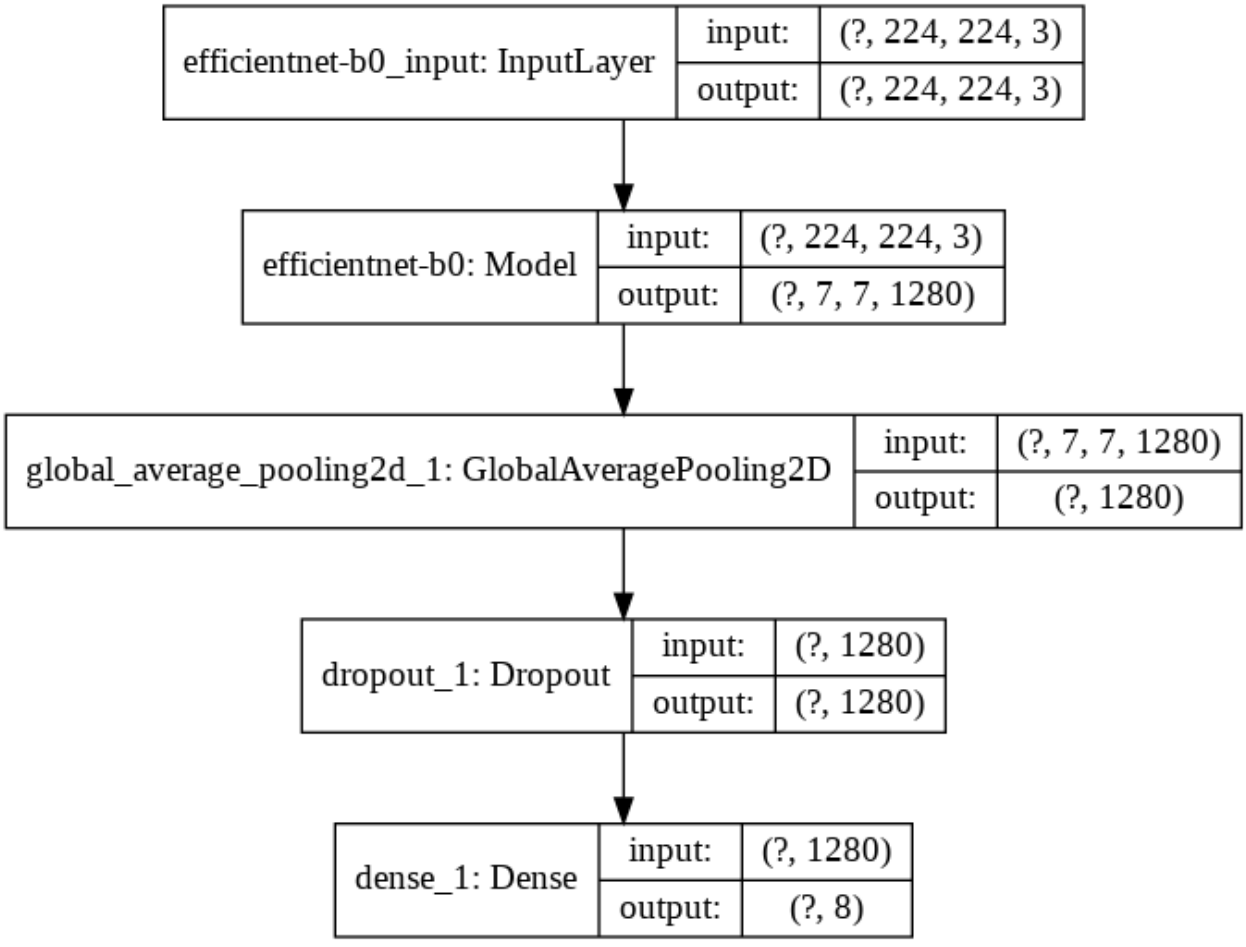


Figure 22: Neural Network Structure

The results achieved so far are:

- We trained and tested our model on Google Colab.
- The training time for the most efficient model was around 5-6 hours for around 20 epochs.
- We tried retrieving the model over three optimizers, i.e. Adam, RMSProp and SGD (Stochastic Gradient Descent).
- Besides this, we did hyper parameter tuning by changing the learning rate and data augmentation.

Finally, we were successfully able to achieve a best validation accuracy of 99.81% and training accuracy of 99.50% using Adam Optimizer, sparse_categorical_crossentropy as the loss function and 0.0001 as the Learning Rate.

The following table shows the results of our training phase:

Base Model	Augmentation	Size	Train-Acc.	Validation-Acc.
InceptionV3	low	155.9 MB	90.05%	87.5%
	high	228.1 MB	69.79%	69.4%
EfficientNet	low	15.8 MB	99.58%	99.44%
	high	**46.7 MB**	**99.5%**	**99.81%**

Table 1: Training and validation accuracy over different hyperparameters

The model training phase was also performed by using VGG16 and InceptionV3 as our base models, but they were rejected because of the following reasons:

- VGG16: This model is a very heavyweight model, which slowers down our predictions. Since we require our predictions to be as fast as possible, so we decided to discard using this as our base model.
- InceptionV3: This model stopped learning after reaching certain percentage accuracy (very low accuracy value). Since the achieved accuracy wasn't satisfactory for our application, we decided to discard the same.

We have used different optimizers, learning rates and augmentations:

1. Optimizers: Adam, RMSProp & SGD (Stochastic Gradient Descent).
2. Learning rates: varying from 0.0001 to 0.001
3. Data Augmentation: Flips, Zoom, Shifts, Shear, Rotations

5.2 Model Predictions:

Our predictions had all bases covered. The model was found to predict correctly on most of the edge test cases. Some of the cases are listed below:

1. Normal case of currency prediction:

```
File Name: images\IMG_20211121_174820.jpg
Prediction: 20
```

Figure 23: Prediction case 1

2. Folded currency notes:

```
File Name: images\IMG_20211121_175500.jpg
Prediction: 50
```

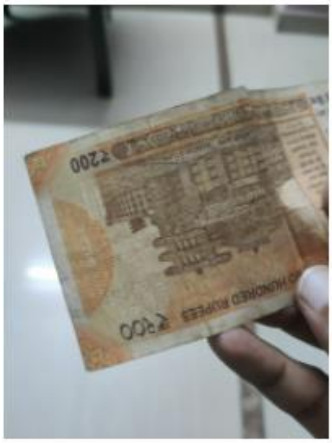
Figure 24: Prediction case 2

3. Cropped currency notes:

```
File Name: images\IMG_20211121_230112.jpg
Prediction: 500
```

Figure 25: Prediction case 3

4. Flipped and Cropped currency notes:



```
File Name: images\IMG_20211121_230151.jpg
Prediction: 200
```

Figure 26: Prediction case 4

5. Image without currency note:

Figure 27: Prediction case 5

Hence, our model performed really well on all the different scenarios.

The model training and validation accuracy on the first 10 epochs is shown in the below graph:

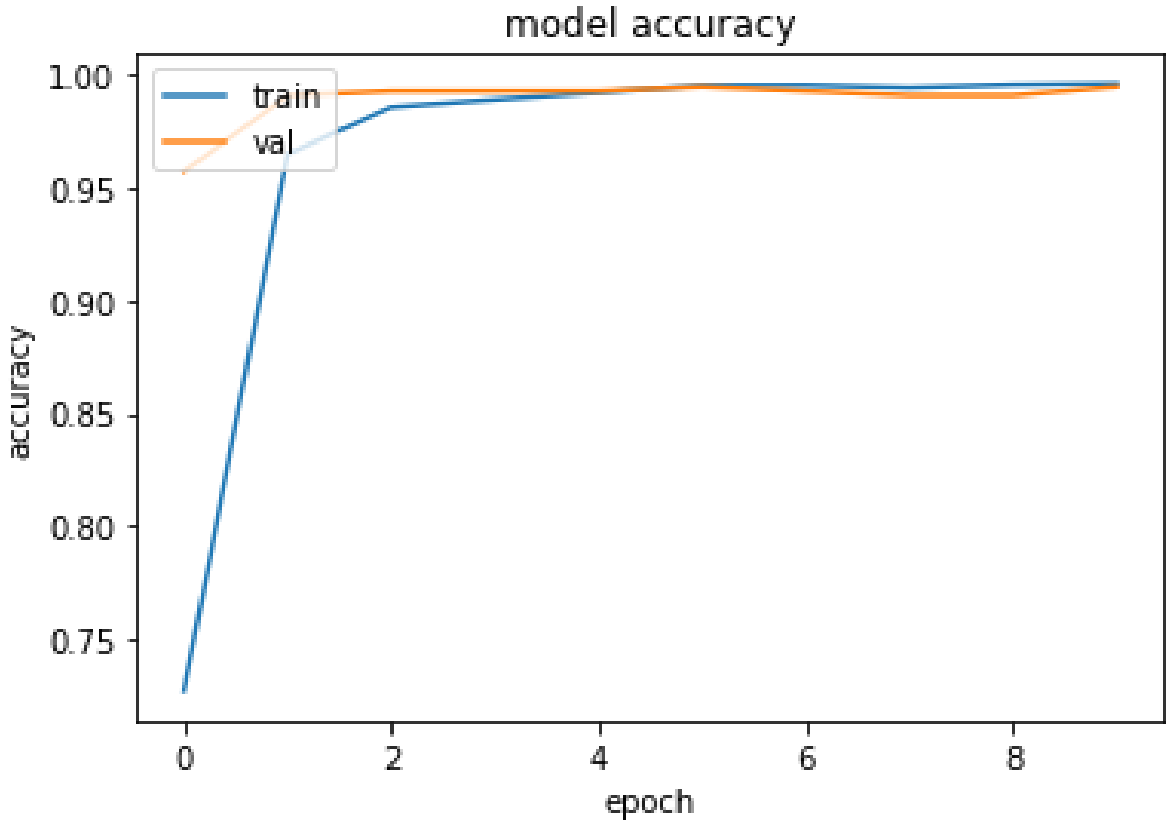


Figure 28: Accuracy curve

5.3 DISCUSSION

Through this project, we help a blind or visually impaired person by helping them in their everyday activities such as recognizing currency notes, local navigation aide, and text to speech from a video feed. Also, this project has the potential to get implemented in different parts of the world helping users with their day to day chores.

The currency recognizer will really add a lot of value to the daily life of a blind or a visually impaired person as it involves financial transactions and helps them easily recognize currency notes, because recognizing it just through size and texture of notes is extremely difficult, as they are almost the same.

Also, the text to speech feature can be widely used for reading out text from a video input, which can be of a book, newspaper or text written on a product. This feature can be further expanded to more applications.

Also, as travelling and navigation are part of any person's daily life, and for a visually impaired person, direction-finding is something where they find a lot of difficulty. Our navigation assistant informs them about obstacles during navigation, and makes their process of travelling and moving around pretty easy and seamless.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

Drishti is a mobile application made using React Native, JavaScript, Flask and GraphQL incorporating Deep learning techniques.

- A model trained to predict currency notes in a variety of situations, taking into account that the users of the app will be blind and visually impaired people.
- Mobile App features:
    - Currency Note Recognition
- Output: Voice output for guiding the users.
- Easy to use mobile app for the users.

We have made a App Application, which helps a visually impaired or a blind person in their everyday life tasks such as Recognizing Currency Notes.

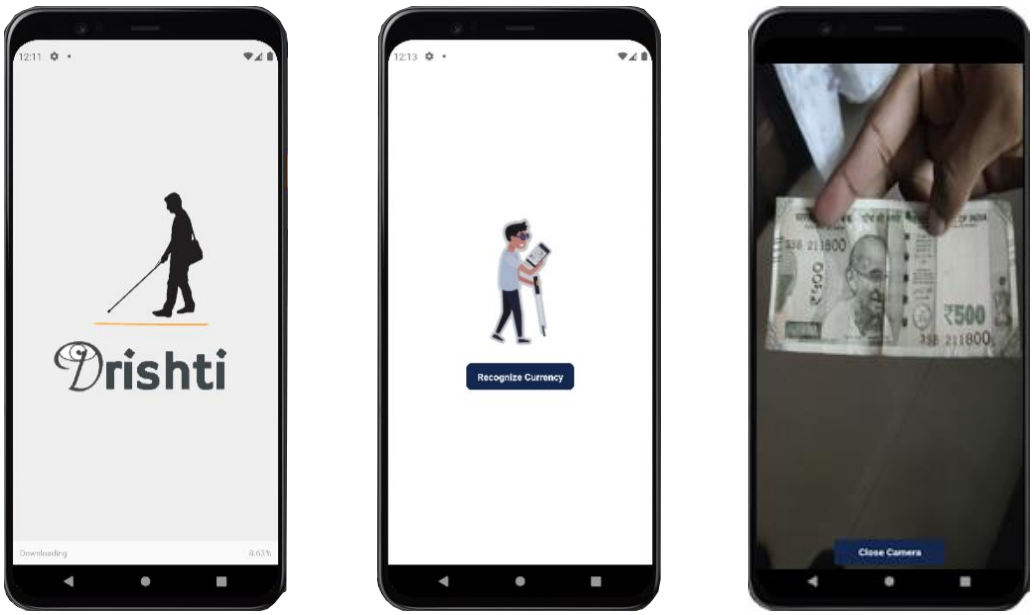


Figure 29: Drishti

6.2 Future Scope

CURRENT CHALLENGES:

- The Currency note recognition model currently needs to be integrated with React Native, having flask and GraphQL as our backend.
- Although the output speed is quite decent, we still aim to make it faster, so that we are able to provide even better experience to our users.
- Accuracy can be made even better having a larger dataset consisting of currency notes, and then training our model over that.

FURTHER SCOPE:

- Local Navigation Aide: We aim to guide the visually impaired people and inform them about the obstacles around them.

Figure 30: local navigation aide

- Text to speech from a video feed: We plan to convert a video feed containing text to speech. Basically, the user will have the text content underneath the camera, and then using a short video clip, we will process the text in it, and then give the processed text as an audio output for the user.
- Smart Wearable Glasses: An additional feature which we plan to implement is smart glasses, which will incorporate all the above mentioned features in wearable glasses.

The glasses will have a camera attached to it, which will directly take in input feed. The output for the same can be given using the glasses itself.

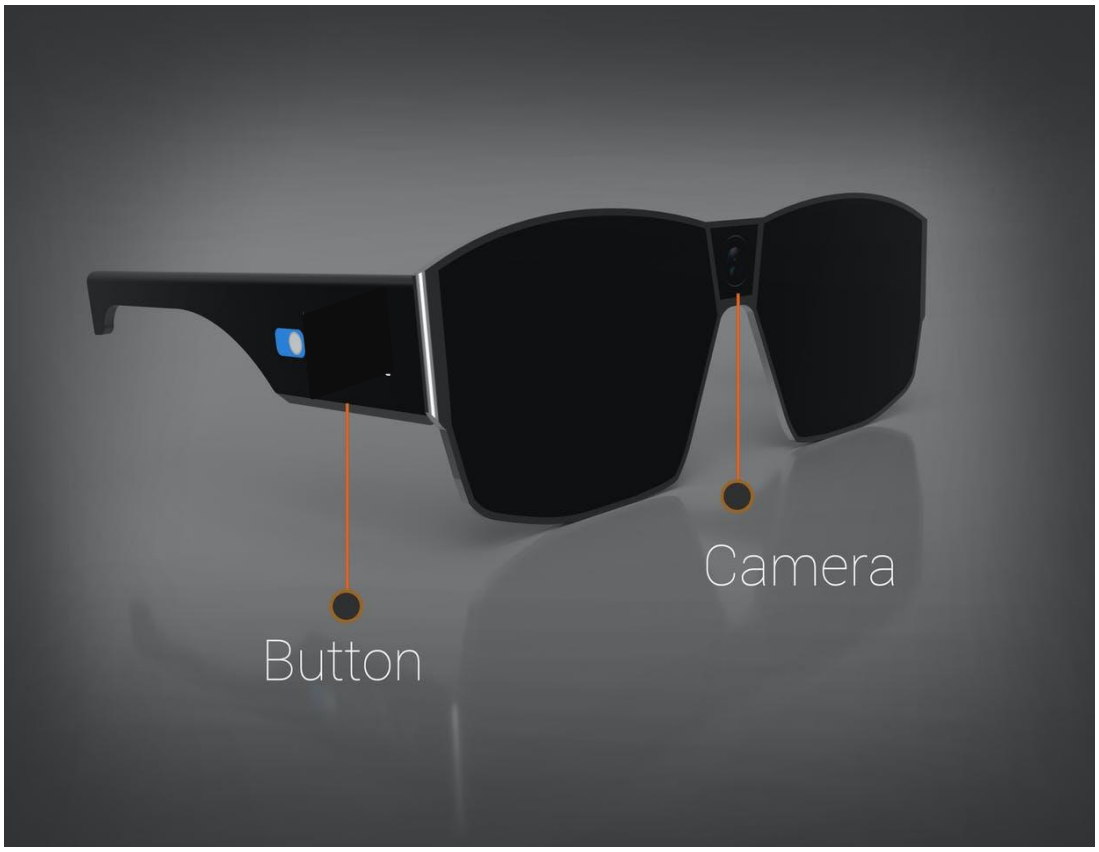


Figure 31: Smart wearable glasses

CHAPTER 7

REFERENCES

[1]. CNN based framework for identifying the Indian currency denomination for physically challenged people, P.Selvi Rajendran , Dr. T.P. Anithaashri (Hindustan Institute of Technology and Science, Chennai)

[2]. Indian Currency Detection using Image Recognition Technique, Kalpna Gautam GNA University, Punjab, India

[3]. Indian Currency Note Recognition System using YOLO v3 Methodology, Wafia Rarani, Vrushabh Rode, Chaitali Mahatme, Dhiraj Chavhan, Prof. Kalyani Gholap