DRISHTI-MOBILE APP FOR VISUALLY IMPAIRED

Priyanka Goel, Achint Khann, Gaurav Sharma, Kritika Ranjan, Abhinav Joshi
Department of Computer Science and Engineering
Moradabad Institute of Technology, Moradabad

Abstract:

The objective of this paper is to help unsighted recognizing the object and Indian currency using an android smartphone. This implementation will be a cost effective system to help the visually impaired. Blind people major problem is to navigate the outdoor region; this system is based on Android technology and designed for trying to solve the impossible situation that afflicts the blind people. Our target for making this app is that now visually impaired people would not be dependent to anyone and with the help of this app and with their own senses they can live with the surrounding better.

Keyword*:* CNN, Machine Learning

I. Introduction

Eyes are the greatest gift to mankind, its disability proves to be tough for people to perform their daily tasks and inability to understand what is around them. With the advancement of Machine Learning and cost effective availability of these technologies in smartphones, we propose the implementation of these algorithms to detect object and use it to aid visually impaired using their Android smart phones.

Our app is very easy to use and does not have complex GUI which provides hassle free accessibility for visually impaired people and it provides text to speech feature which is very useful for our user.

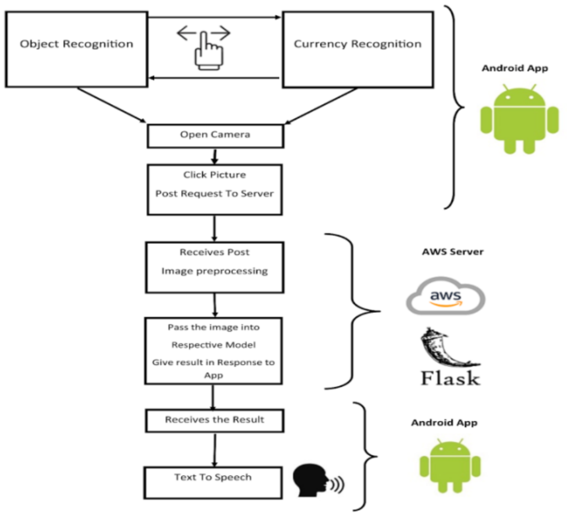


Figure 1: Flow chart of process

Our app has two modes object recognition mode and currency recognition mode and. just swapping the screen left and right user can use currency recognition mode and object recognition mode respectively.

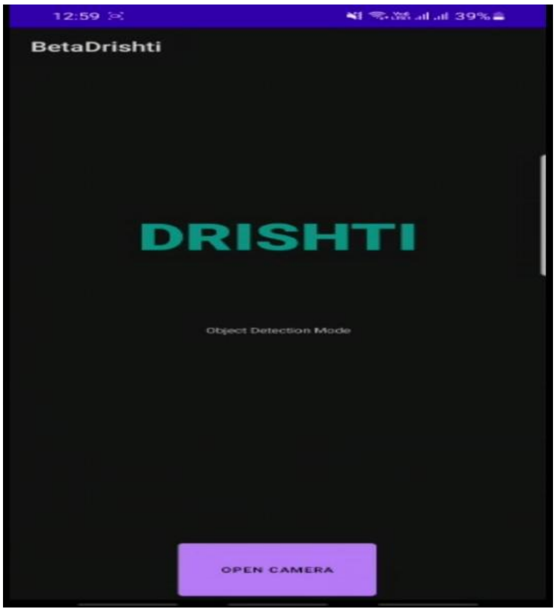


Figure 1: object detection mode by swapping left

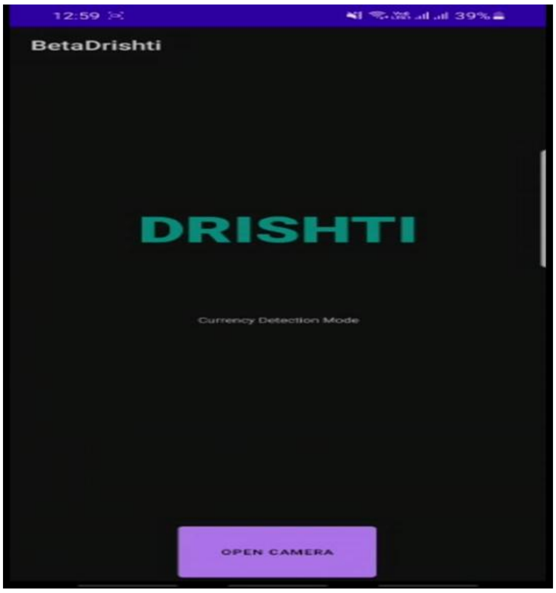


Figure 2: currency detection mode by swapping right

After taking picture, it will post request to server using FLASK and after receiving post image will go into pre-processing in the server and send the result to the app ,now android will make that result into audio output using Texttospeech function in android, the result that will be speech text.

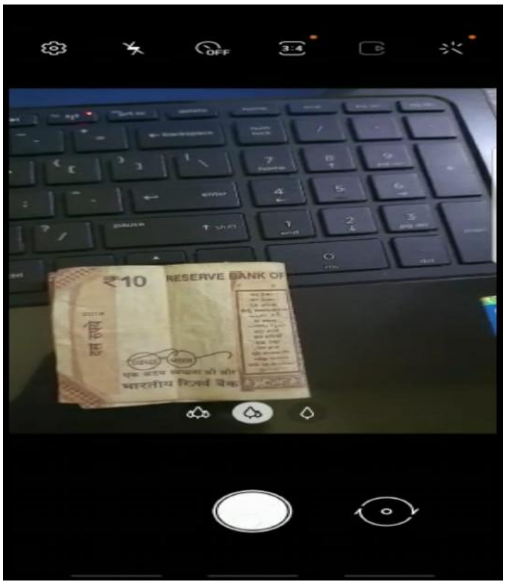
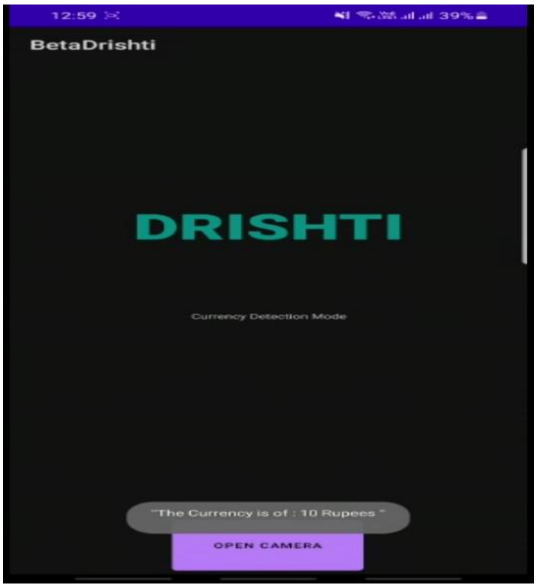


Figure 3: taking picture of a 10 rupee note



"The Currency is of : 10 Rupees "

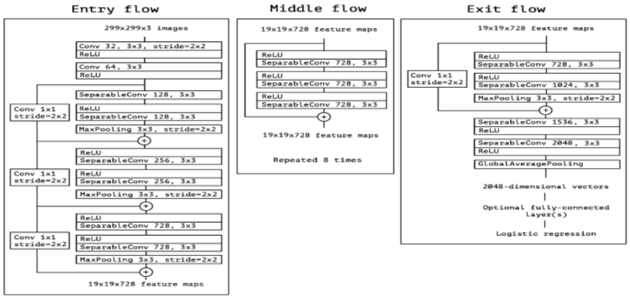
Figure 4: speech text of a picture as a result

II. Methodology

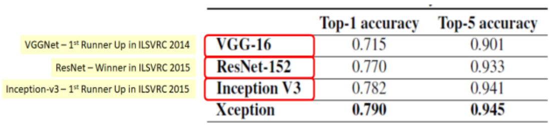
As this project contains some machine learning models so not to make mobile application heavy as it will consume more processing and memory, to overcome this problem the model will be running on a server which is hosted over cloud. We have divided the whole project into 4 sections which are listed below:

CURRENCY RECOGNITION MODULE:

In this module we have made the model that will recognize the currency denomination so to make this model we have used Deep Learning and Transfer Learning. First let us talk about the dataset , we have downloaded the dataset from kaggle in which it contains the seven classes denoting the different types of denomination notes such as 10,20,50..etc.Each class contains up to 80 images .So to train the model firstly we have used 'Xception' as the pre trained model



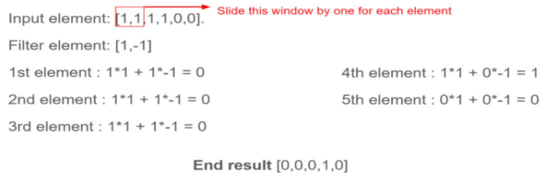
Xception will help the basic model to extract the useful features from the dataset. We have used Xception model here as it has the best accuracy to use it for feature extraction in similar kind of dataset. Then we have add Dense and pooling layer in the end of Xception model and finally the output layer which contains seven nodes ,each node for each class, each node will give the likelihood of image belonging to that and whichever node has maximum value the image will belong to that class or the image is of that currency denomination.



		Top-1 accuracy	Top-5 accuracy
VGGNet – 1st Runner Up in ILSVRC 2014	VGG-16	0.715	0.901
ResNet – Winner in ILSVRC 2015	ResNet-152	0.770	0.933
Inception-v3 – 1st Runner Up in ILSVRC 2015	Inception V3	0.782	0.941
	Xception	**0.790**	**0.945**

Figure 7: Xception v/s other model

In this module we have used CNN (Convolutional Neural Networks) model, exception model is also a type of CNN. CNN is a neural network model which ought to extract higher representations for the image content. In a classical image recognition its id required to define all the features, whereas in CNN it does the job automatically, it trains the model by extracting the raw pixel data. It then go through the image and then the dot product of the data is calculate with the extracted inputs. This allows convolution to emphasize the relevant features.



Input element: [1,1,1,1,0,0].

Slide this window by one for each element

Filter element: [1,-1]

1st element : 1*1 + 1*-1 = 0

2nd element : 1*1 + 1*-1 = 0

3rd element : 1*1 + 1*-1 = 0

4th element : 1*1 + 0*-1 = 1

5th element : 0*1 + 0*-1 = 0

End result [0,0,0,1,0]

Figure 8: 1D Convolution Operation with features(filter)

We will enclose the window elements with a tiny window, dot multiplies it with the filter elements, and save the output. We will iterate the operations to derive 5 output elements as [0,0,0,1,0]. From this output, we get to know that the feature change (1 becomes 0) is in sequence 4, from this we can claim that the operation performed was successful. Similarly, this happened for 2D Convolutions.

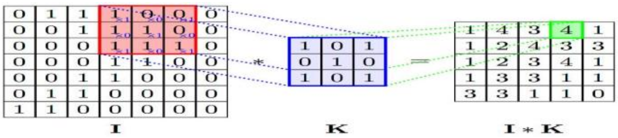


Figure 9: 2D Convolution Operation with features(filter)

B. OBJECT RECOGNITION MODEL

This module is the main or most important section of the project as in this we have to describe the image with a caption .So come over the Dataset, The Dataset which we have used is commonly known as ‘Flickr 8k’ Dataset which basically Contain the general image of some common things and for each image we have 5 captions that describes the image. Every image has 5 caption so as to give a good vocabulary to the model and don’t make model over fit to the dataset or make model more generalized. So we will break the whole process into following step**:**

A: Preprocess the images using Resnet50: So First to extract the features of image we have used the ‘Resnet50’ pre trained model to extract the feature vector of the image

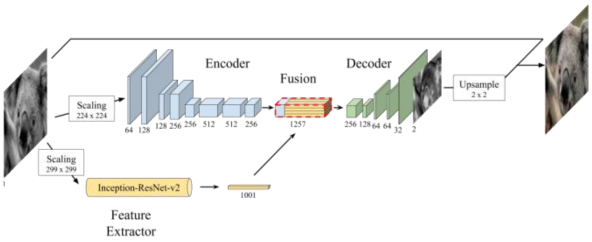


Figure 10: Feature Extraction using Resnet

It will produce the feature vector of size 2048 that will describe the features in the image. We will the save the feature vector of each image using a dictionary like structure and save it to disk so as to use it while generating captions.

B. Text Preprocess: Now we will come over to captions. Here we have created the Dictionary where the image name is key of the dictionary and the captions of the image will be the value of that key. Now we will create the vocabulary of the model by taking the each word present in the captions. Now after creating Vocabulary we will encode the words into numbers as in model it will use number so to make working of the model less complex.

C. RNN Model: This is LSTM (Long Short Term Memory) type of RNN (Recurrent Neural Network) model which will use the caption dictionary and image feature vector to predict the caption correctly by predicting word by word by caption.

D. Predicting the Caption: So we will be predicting the caption using the caption dictionary and image feature vector.so basically we will use the above RNN model to predict the next word in the caption, to predict the next word the model will produce the probability of each word and use the word as the next word which have the maximum probability and check about the prediction with the original caption while training.

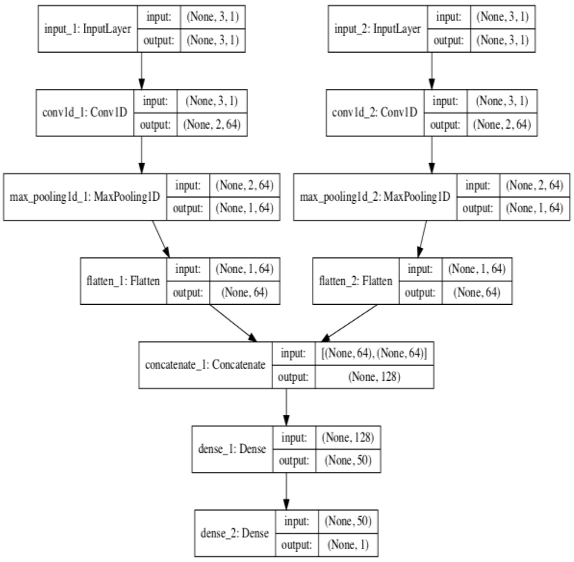

input_1: InputLayer | input: (None, 3, 1) | output: (None, 3, 1)
conv1d_1: Conv1D | input: (None, 3, 1) | output: (None, 2, 64)
max_pooling1d_1: MaxPooling1D | input: (None, 2, 64) | output: (None, 1, 64)
flatten_1: Flatten | input: (None, 1, 64) | output: (None, 64)
input_2: InputLayer | input: (None, 3, 1) | output: (None, 3, 1)
conv1d_2: Conv1D | input: (None, 3, 1) | output: (None, 2, 64)
max_pooling1d_2: MaxPooling1D | input: (None, 2, 64) | output: (None, 1, 64)
flatten_2: Flatten | input: (None, 1, 64) | output: (None, 64)
concatenate_1: Concatenate | input: [(None, 64), (None, 64)] | output: (None, 128)
dense_1: Dense | input: (None, 128) | output: (None, 50)
dense_2: Dense | input: (None, 50) | output: (None, 1)

Figure 11: RNN model

C . HOSTING THE SERVER ON CLOUD:

We have generated two machine learning models and embedding them in the android app will create a dependency of processor and the results may vary for various devices thus, we have uploaded the models to an AWS cloud and the outputs are generated upon the data passed. The cloud is hosted with a windows server on the cloud (AWS) and will make the server using the Flask (python web framework) – It is micro framework because it does not require particular tools or libraries.

D. MOBILE APPLICATION:

The mobile application is an Android application should have a very simple UI, which will be responsible for capturing an image and sending it to the destined server using HTTP POST request and the output for the same received will be displayed as TOAST message and as an audio output.

III. Conclusion

In this paper we proposed a system which can be used to assist the visually impaired person in understanding the environment by narrating the objects in the surrounding. The developed system is based on using a website which on loading takes the image from the back camera of the phone and pass that image to the server. On server side, a trained machine learning model is deployed to detect the objects in that image. The result of detection is passed to the client app where a voice library narrates the results so that the visually impaired person can hear. The results showed that the accuracy is varying depending on phone camera quality and the light effects, the average accuracy comes out is to be 75%.

References

1. Image Captioning with visual attention using Tensorflow

(https://www.tensorflow.org/tutorials/text/image_captioning).

2. XceptionModel (https://towardsdatascience.com/review-xception-withdepthwise-separable-convolution-better-than-inception-v3-image)
3. Understanding ResnetModel (https://towardsdatascience.com/ understandingand-coding-a-resnet-in-keras)