Indian Currency Predictor

Authors

a. S. Sai Mohith (2019501093)
b. N. Vamsi Krishna (2019501054)
c. P. Sharath Chandra (2019501070)
d. I. Jeishnavi Chandra (2019501089)
e. M. Lakshmi Sindu (2019501096)

1. Abstract

The bank notes are the most popular means of value circulation. The present day financial self-service has made currency recognition a key activity of the automated banking process. The process of currency recognition by some extrinsic feature becomes error prone due the pollution and depreciation suffered by the currency during the transactions. Therefore we propose an Indian currency recognition system based on the intrinsic feature of the currency. In the present work we extracted images as intrinsic features for currency recognition and evaluated the class discriminating capability of these features for Indian currency.

2. Introduction

Paper money is still a widely accepted mode of money transaction besides so many alternatives. The attractive features of the paper currency include privacy, simplicity, durability and complete control. But as a means of value transaction it lacks intrinsic value, and mechanism of reversal in case of repudiation, except the credential support by the state. Recent phenomena of financial self service being supported by the banks and other financial institutions have started various services of automated banking systems which have currency recognition as its key activity making automated currency recognition and classification a key problem. So, Our application is made to recognise the denomination of currency and predict it. It recognises the currency through a webcam or by browsing images. The following are the currency denominations that are used in our application are: 10, 20, 50, 100, 200, 500, 2000.

3. Preprocessing

In order to build our data set we use a keras library which is known as keras Image Augmentation.

So, In order to make the most of our few training samples, we will "augment" them via a number of random transformations like zoom_range, horizontal_flip,

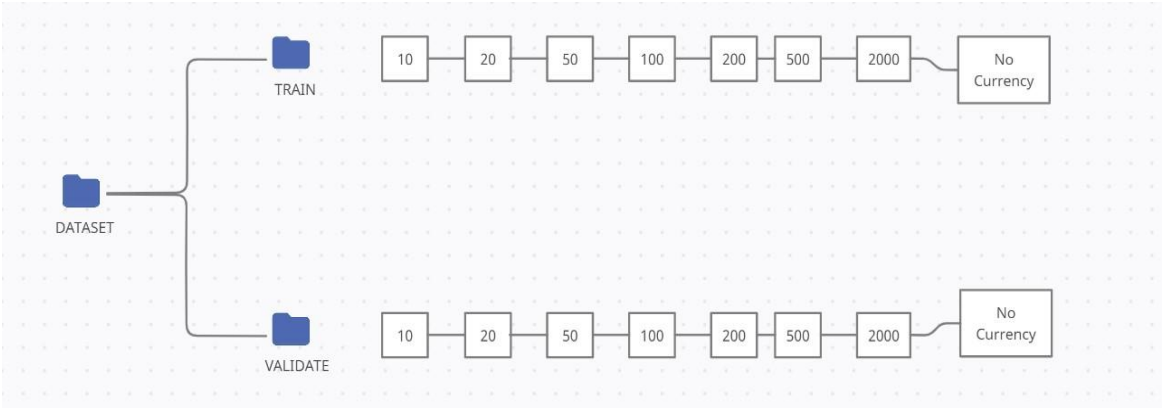
vertical_flip etc., so that our model would never see twice the exact same picture. This helps prevent overfitting and helps the model generalize better.

Input Image:

Augmented images:

4. Data

We have generated our currency dataset by using the above preprocessing technique and generated the training and validation split. The contents inside our dataset looks like the following.



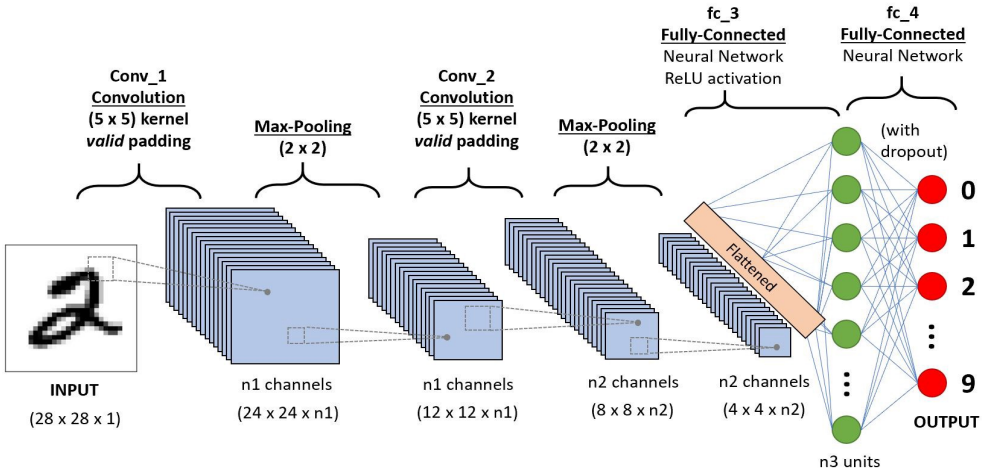
We have split the dataset into Train and Test split where 80% of the images are under the Train folder and 20% of the images are under the validate folder.
After splitting the dataset, we have built our models i.e one by using CNN and the other is by using Google teachable machine and found better accuracy and loss values when we use Google Teachable Machine.

5. Models

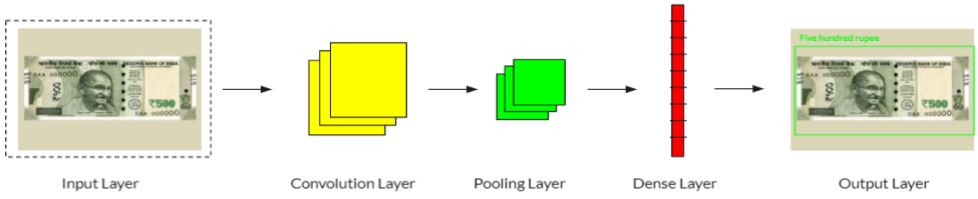
a. CNN (Convolution Neural Network):

Convolutional Neural networks are designed to process data through multiple layers of arrays. This type of neural networks is used in applications like image recognition or face recognition. The primary difference between CNN and any other ordinary neural network is that CNN takes input as a two-dimensional array and operates directly on the images rather than focusing on feature extraction which other neural networks focus on.

Architecture:



CNN structure for currency recognition:



Input Layer:

Input layer in CNN should contain image data. Image data is represented by a three dimensional matrix. Here we need to reshape it into a single column.

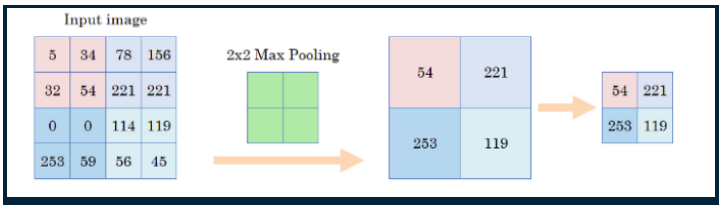
Convolution 2D Layer:

Convolutional layers are the **layers** where filters are applied to the original image. The most important parameters are the number of kernels and the size of the kernels.

For example, if you would apply a **convolution** to an image, you will be decreasing the image size as well as bringing all the information in the field together into a single pixel. The final output of the **convolutional layer** is a vector.

MaxPooling Layer:

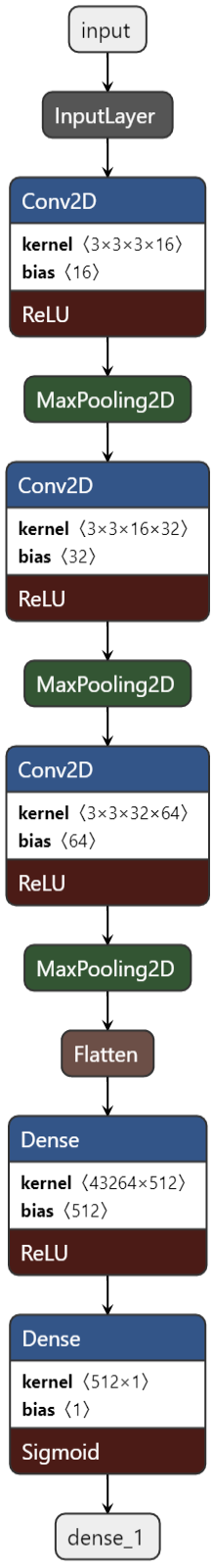
A **pooling layer** is another building block of a CNN. **Pooling**. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. **Pooling layer** operates on each feature map independently. The most common approach used in **pooling** is max **pooling.** Here the image size is reduced to half i.e 2 x 2.



Dense Layer:

Dense layer is the regular deeply connected neural network **layer i.e** each neuron in the dense layer receives input from all neurons of its previous layer. It is the most common and frequently used **layer**. **Dense layer** does the below operation on the input and returns the output.

Output = activation(**relu**)
Output = activation(dot(input, kernel) + bias)



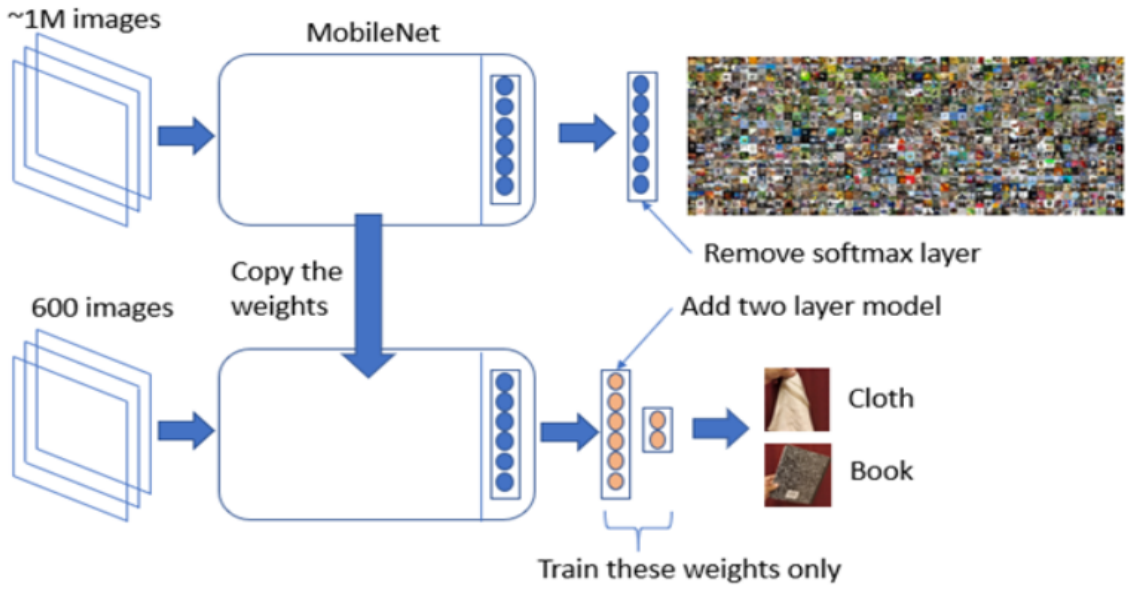
Model Visualization

b. Google Teachable Machine:

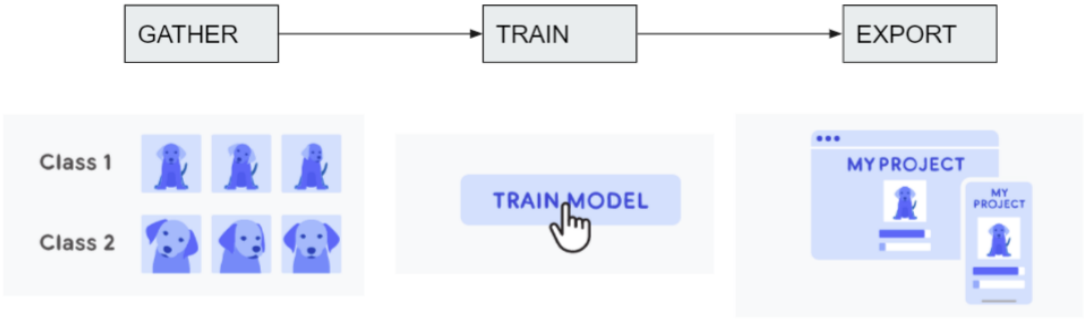
The teachable machine uses transfer learning; A method that uses transferable knowledge from another model and refines it using data available for the task at hand.

In the case of neural network models, the first approach uses a base model that does very well at some general tasks like classifying all types of images into a thousand different classes. We just strip out the last classifier layer, fit our own classification layer and train only this layer with new images. In this case, our base model simply functions as a 'fixed feature extractor' by creating a representation of an image that has captured generally relevant features.

Let's take a look at the following image for better understanding of how a teachable machine works.



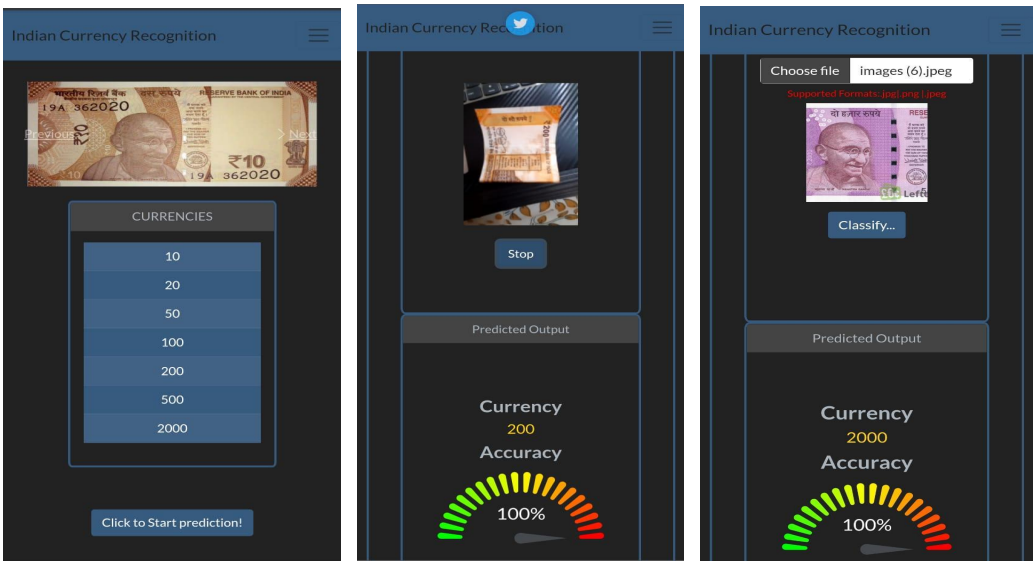
In the above figure we can notice that a teachable machine takes the copies of the weights from a pretrained model using transfer learning and will train only those particular weights. By doing this it reduces the complexity of training a dataset and also produces better accuracy in prediction.



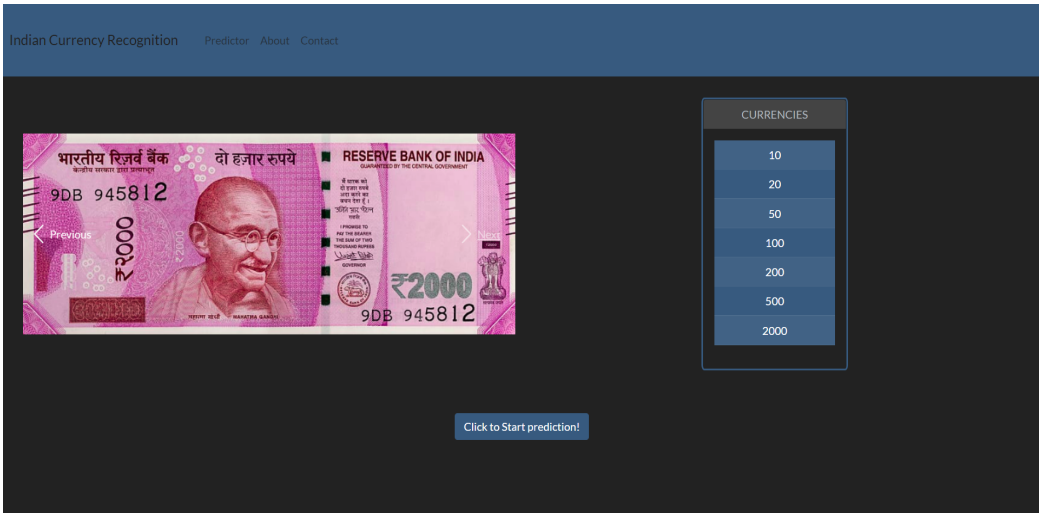
- ☐ **Gather:** We have gathered and organized our dataset into classes, or categories, at this point.
- ☐ **Train:** We trained our model and then immediately tested it to see if it could classify new examples correctly.
- ☐ **Export:** Finally, we exported our model for Indian Currency Prediction, which included three files: model.json, metadata.json, and weights.bin, which we used to make predictions via webcam or browse an image using the react app that we created.

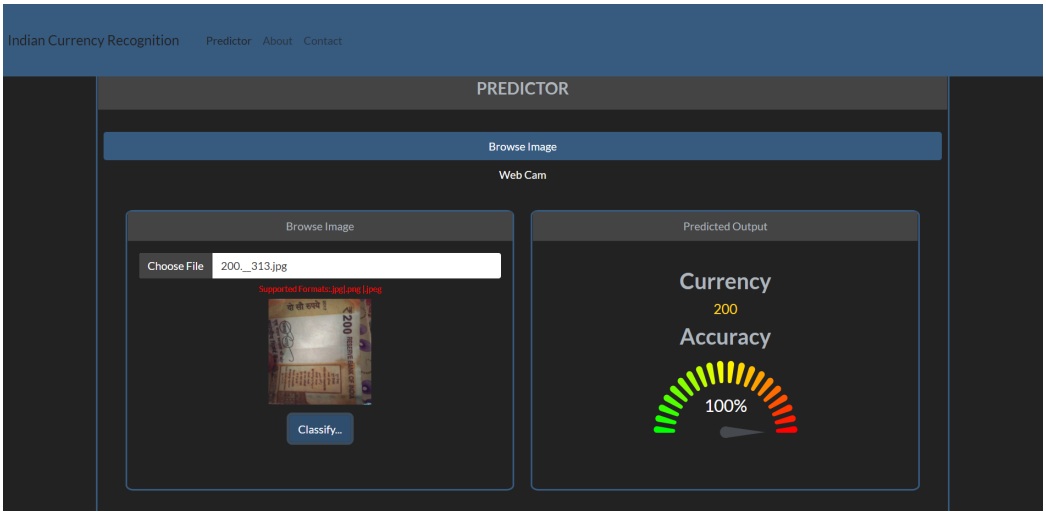
6. Project Outlook

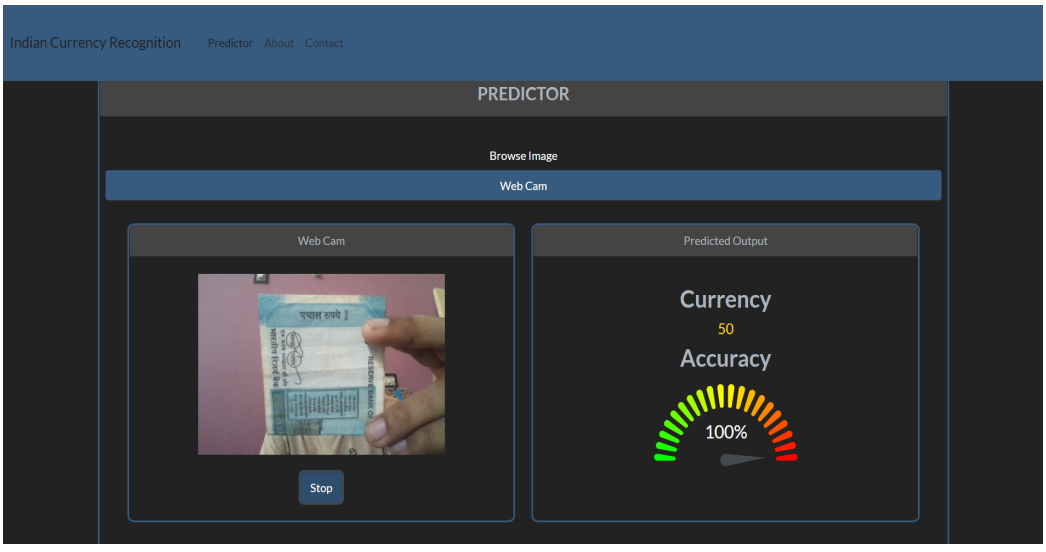
a. Mobile APK UI



b. Website UI


Indian Currency Recognition
Predictor
About
Contact
CURRENCIES
10
20
50
100
200
500
2000
Previous
Next
Click to Start prediction!


Indian Currency Recognition
Predictor
About
Contact
PREDICTOR
Browse Image
Web Cam
Browse Image
Choose File
200__313.jpg
Classify...
Predicted Output
Currency
200
Accuracy
100%


Indian Currency Recognition
Predictor
About
Contact
PREDICTOR
Browse Image
Web Cam
Web Cam
Stop
Predicted Output
Currency
50
Accuracy
100%

7. Conclusion

After building and observing the datasets of various models, we chose a model which gets more accuracy for the prediction. The Teachable machine helps us to train and analyse the data and gives the outcome as the model file. So, the future scope of the project would be to try these models on different kinds of currencies all over the world and to implement them with more accuracy.

8. References

https://ieeexplore.ieee.org/abstract/document/6153231
https://towardsdatascience.com/have-you-taught-your-machine-yet-45540b7e646b
https://blog.keras.io/building-powerful-image-classification-models-using-very-little-data.html