**COIN RECOGNITION AND COUNTING APPLICATION**

J Component Project Report for the course

CSE4019 Image Processing

***by***

**Kartik Narang(18BCE1199)**

**Mansi Parashar(18BCE1048)**

**Anand Bhalotia(18BCE1121)**

**Arpit Gupta(18bce1031)**

**Adarsh Sahu(18BCE1029)**

*Submitted to*

**Dr. Malathi**

****

SCHOOL OF COMPUTER SCIENCE ENGINEERING

VELLORE INSTITUTE OF TECHNOLOGY

CHENNAI - 600127

*October 2020*

**ABSTRACT**

The objective of this project is to create a web application to classify recently released Indian coins of different denomination. The objective is to recognize the coins and count the total value of the coin in terms of Indian National Rupees (INR). In this we will acquire images from the datasets present on the internet as well as make our own datasets by clicking pictures, do the pre-processing and feature extraction and try to recognize the value of the coin from the image. We will be using 1 rupee, 2 rupee, 5 rupee and 10 rupee coins of different designs currently being made and used. The features of old coins and new coins of different denominations are considered for classification.

Indian Coins are released with different values and are classified based on different parameters of coin such as shape, size, surface, weight and so on. Most of the work that has been done on coin recognition is based on mechanical features such as width, weight and diameter. But this paper concentrates on visual features of the coin. We are working with affine transformations such as simple gray level scaling, shearing, rotation etc. The coins are well recognized by zooming processes by which a coin size of the image is increased. We are separating the coins from the image, extracting the features and then using machine learning/deep learning classifiers to classify the coins.

To implement the coin classification, code is written in OPEN CV Python and tested with simulated results. A method is proposed for realizing a simple automatic coin recognition system more effectively. The approach (image processing) being used is new in the field and not much exploration has been done so far.

**Keywords**: Feature extraction, Edge detection, Image Enhancement, Classification, OpenCV, Segmentation, Image pre-processing

**MOTIVATION**

Coin recognition is one of the emerging research fields in modern times. There is vital need of an efficient and robust coin recognition system in our daily life. Coins are an integral part of our life; we cannot imagine our daily life without them. Inspite of daily uses, coin recognition system can prove helpful for recognition purpose in research organizations who deal with ancient coins . Hence, there is an obvious need of coins to be automatically recognized and sorted by computers. The recognition system should be able to recognize coins properly, as the further transactions would depend on the accuracy of recognition. There is problem of false image recognition due to the presence of noise in input data. The coin recognition system must be robust in the manner as it should be able to recognize images efficiently even if noise is present; as all capturing devices, analog or digital, have attributes which make them noise susceptible. There is need of a robust coin identification system, which can recognize images in normal as well as in noisy environment. The system must also be fast and cost effective.

**PROBLEM STATEMENT**

The objective of this project is to create a web application to classify recently released Indian coins of different denomination. The objective is to recognize the coins and count the total value of the coin in terms of Indian National Rupees (INR). To implement the coin classification, code is written in OPEN CV Python and tested with simulated results

**LITERATURE SURVEY**

Some of the recent works done in the topic are:

1. **A Survey on Indian Coin Detection Techniques Using Digital Image Processing**

Date: November, 2016

<https://www.ijarcce.com/upload/2016/november-16/IJARCCE%2092.pdf>

They proposed a method for Indian coin detection under varying lighting conditions. The basic idea behind their method is to extract features of known coins and put in a database. And when a new coin is entered its features are compared with the images of the coin that stored in the database. They trained their system under different lighting conditions. They assumed different lighting intensities, which occurs due to various reasons such as power fluctuations and tried to detect Indian coin value under these conditions. This work basically involved automatic detection and recognition of Indian coin using morphological operations of image processing. Image processing is a process in which an input image is transformed into digital image form and performs some operation on it, In order to get high image property and to extract some useful information from it. This work is a prototype simulation and a proof of concept. The application of this work ranges from simple vending machines to automatic mobile recharging.

1. **Automated Coin Recognition System using ANN**

Date: July, 2011

<https://www.researchgate.net/publication/317014533_Automated_Coin_Recognition_-_Approaches_and_Techniques>

Developed an ANN (Artificial Neural Network) based Automated Coin Recognition System for the recognition of Indian Coins of denomination `1, `2, `5 and `10 with rotation invariance. They took images from both sides of coin. So, this system is

capable of recognizing coins from both sides. Features are extracted from images using techniques of Hough Transformation, Pattern Averaging etc. Then, the extracted features are passed as input to a trained Neural Network. 97.74% recognition rate has been achieved during the experiments i.e. only 2.26% miss recognition, which is quite encouraging.

**SYSTEM ARCHITECTURE**

**Data Source**

Since, our approach is based on image processing, the data we gathered is in the form of images.

Firstly, we found the one and only dataset present on the internet which contains the images of the Indian coins of denominations Rs. 1, Rs. 2, Rs 5 and Rs 10. This dataset had about 650 images in total. The link is provided below:

<https://www.kaggle.com/vishu53/indian-coins-dataset>

But this was no where enough and the quality of the images was also not very good. So we clicked our own pictures and added to the dataset. As of now, the dataset has about 1500 images.

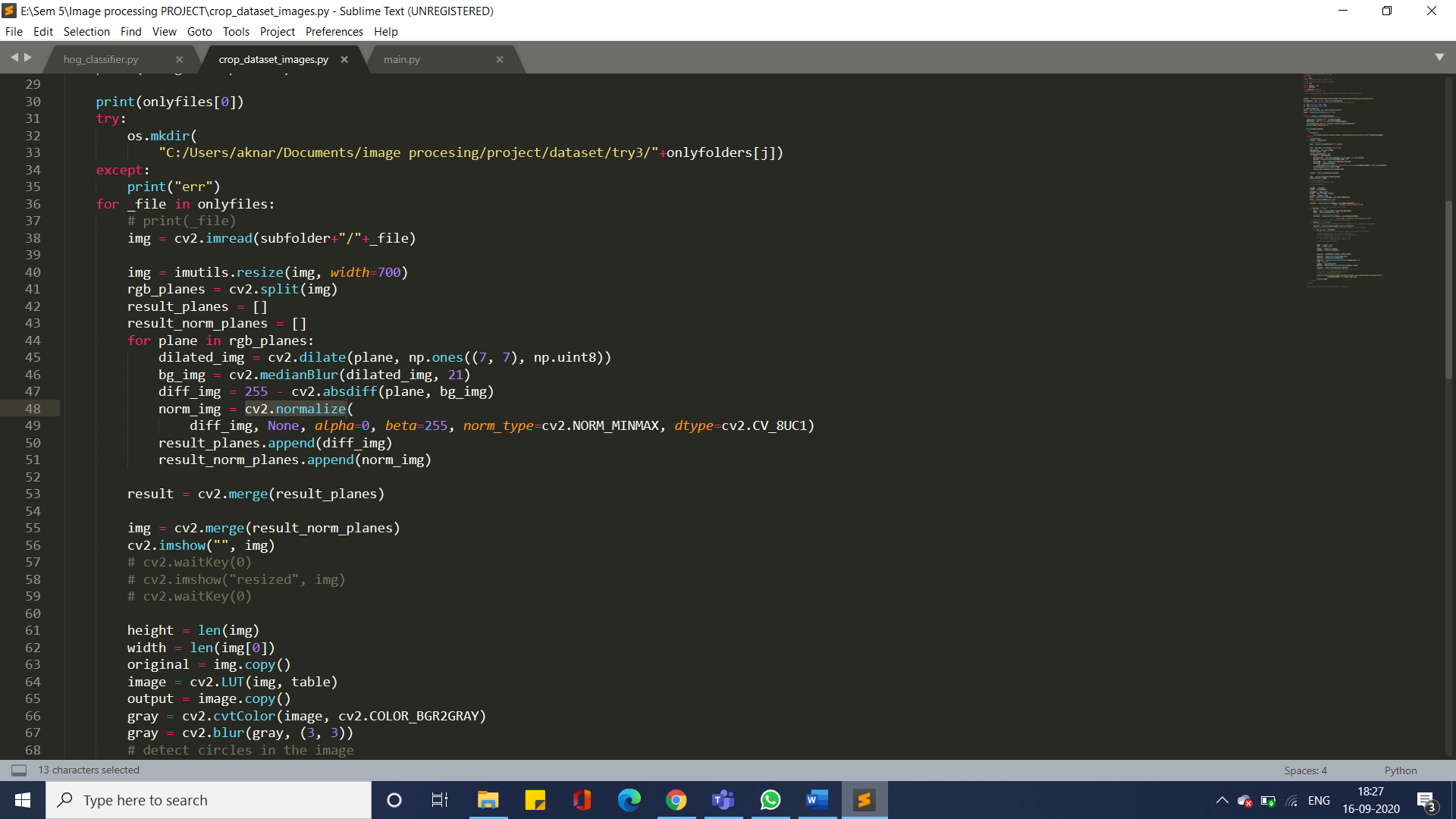
Note that, it contains images of both the sides of coins. Also, images of coins of multiple designs of a coin are taken to increase the practicality of the project. To increase this further, the images vary in aspects like the height, angle, lighting, rotation, etc.

The data is then categorized for training.

**Data Pre-processing**

For data pre-processing, which is image preprocessing in this case we have utilized various open cv tools for various steps. Image preprocessing contains steps like image enhancement, edge detection and segmentation.

For image enhancement, we have resized all the images. We have split the rgb images into different planes and then applied enhancement methods like cv2.dilate(), cv2.medianBlur(), cv2.normalize(), etc and then merged the planes.



Then, we have applied histrogram equalization as well as contrast limited adaptive histrogram equalization (CLAHE) to increase the contrast of the images.

We have used the open cv function cv2.HoughCircles() for edge detection and segmentation.

For feature extraction, we have used HOG i.e Histogram of Oriented Gradients.

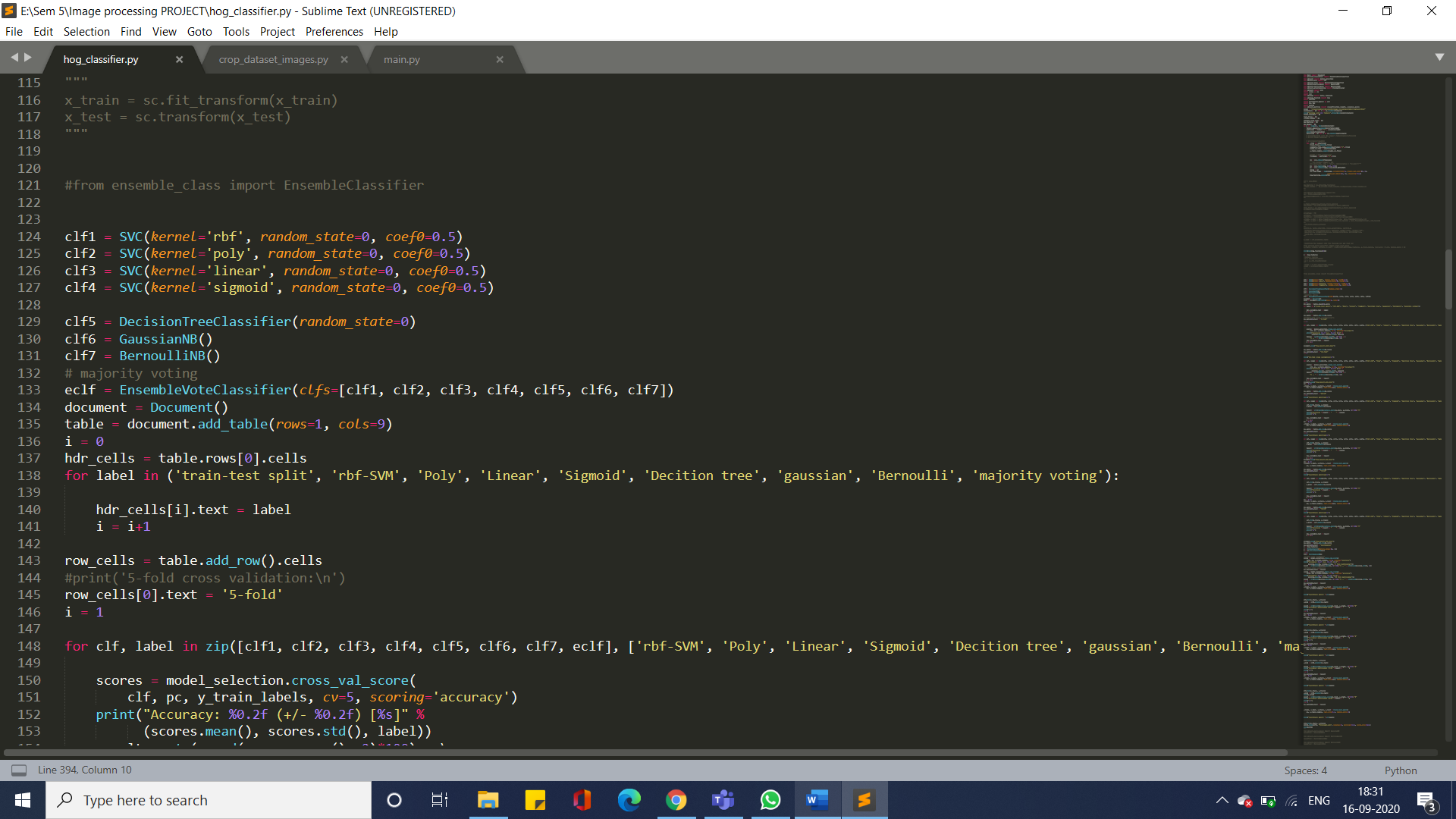
**Models**

We have used about 7 classification models to classify the joins after creating the feature vectors. And we have chosen the best results from the models.

The models we have used are:

1. Rbf support vector classifier
2. Poly support vector classifier
3. Linear support vector classifier
4. Sigmoid support vector classifier
5. Decision tree classifier
6. Gaussian naïve bayes classifier
7. Bernoulli naïve bayes classifier

And then we have used Ensemble Vote Classifier to compare all the results of all the seven and get the best outcome.



**RESULTS FOR TAILS DATASET**:

Starting off with this project, we were not sure which classifiers will give us the best results. So, we decided to take an alternative path.

In order to find what method gives out best results, we tried out various classifiers in various Train-Test data splits.

Following are the results we achieved.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| train-test split | rbf-SVM | Poly | Linear | Sigmoid | Decision tree | gaussian | Bernoulli | majority voting |
| 5-fold | 53.0% -- 0.13 | 63.0% -- 0.16 | 63.0% -- 0.16 | 16.0% -- 0.01 | 34.0% -- 0.05 | 47.0% -- 0.12 | 22.0% -- 0.05 | 56.00000000000001% -- 0.12 |
| 10-fold | 61.0% -- 0.17 | 69.0% -- 0.14 | 68.0% -- 0.15 | 17.0% -- 0.04 | 32.0% -- 0.11 | 53.0% -- 0.14 | 22.0% -- 0.03 | 64.0% -- 0.14 |
| 85:15 | 61.0% | 78.0% | 78.0% | 4.0% | 37.0% | 55.00000000000001% | 24.0% | 67.0% |
| 80:20 | 63.0% | 78.0% | 78.0% | 8.0% | 35.0% | 56.99999999999999% | 28.000000000000004% | 66.0% |
| 75:25 | 56.00000000000001% | 74.0% | 74.0% | 7.000000000000001% | 31.0% | 48.0% | 22.0% | 62.0% |
| 70:30 | 54.0% | 77.0% | 77.0% | 9.0% | 39.0% | 48.0% | 16.0% | 62.0% |

Accuracy Results Table

As it is clear from diagram, we received best results from Poly SVC and Linear SVC.

The worst results were given by Sigmoid SVC and Bernoulli NB.

Hence going forth, we will be focusing on Poly SVC as our main classifier for the project.

**RESULTS FOR BOTH TAILS AND HEADS TOGETHER:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| train-test split | rbf-SVM | Poly | Linear | Sigmoid | Decition tree | gaussian | Bernoulli | majority voting |
| 5-fold | 40.0% -- 0.08 | 50.0% -- 0.1 | 50.0% -- 0.09 | 12.0% -- 0.0 | 25.0% -- 0.05 | 40.0% -- 0.08 | 17.0% -- 0.02 | 43.0% -- 0.05 |
| 10-fold | 44.0% -- 0.07 | 52.0% -- 0.08 | 52.0% -- 0.08 | 12.0% -- 0.0 | 24.0% -- 0.05 | 44.0% -- 0.1 | 19.0% -- 0.04 | 46.0% -- 0.06 |
| 85:15 | 45.0% | 62.0% | 60.0% | 13.0% | 22.0% | 49.0% | 21.0% | 50.0% |
| 80:20 | 45.0% | 56.99999999999999% | 56.99999999999999% | 13.0% | 24.0% | 47.0% | 20.0% | 47.0% |
| 75:25 | 46.0% | 56.00000000000001% | 56.00000000000001% | 13.0% | 20.0% | 44.0% | 22.0% | 47.0% |
| 70:30 | 43.0% | 59.0% | 57.99999999999999% | 12.0% | 24.0% | 40.0% | 22.0% | 47.0% |

**System Flow**

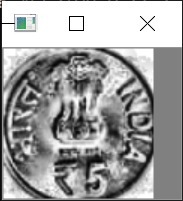
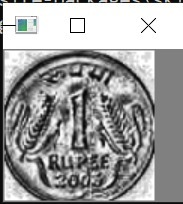
**Backend Processing**

1. **Input**: The input is an image with multiple or single coin pictures.

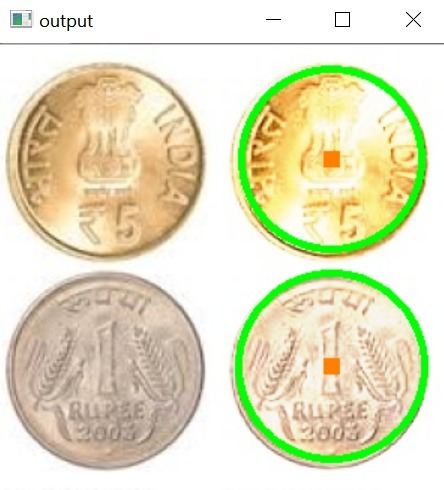


*Fig: Input Image*

1. The input image goes through multiple image processing techniques for image enhancement and object detection. As a result, segmented coin images are retrieved from the input image.

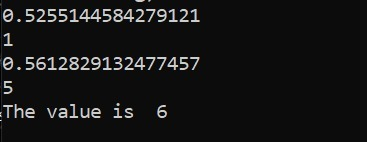


*Fig: Segmented and detected separate coins from single input image*



*Fig: Input Image and output image after detection*

1. Features are extracted from each segmented coin image, and the prediction function is called to predict the class the coin lies in.From the classification, the value of the coin is extracted, and is added to a global counter which is initialised to keep track of the value of coins in the image. The **OUTPUT** is the value after coin counting.

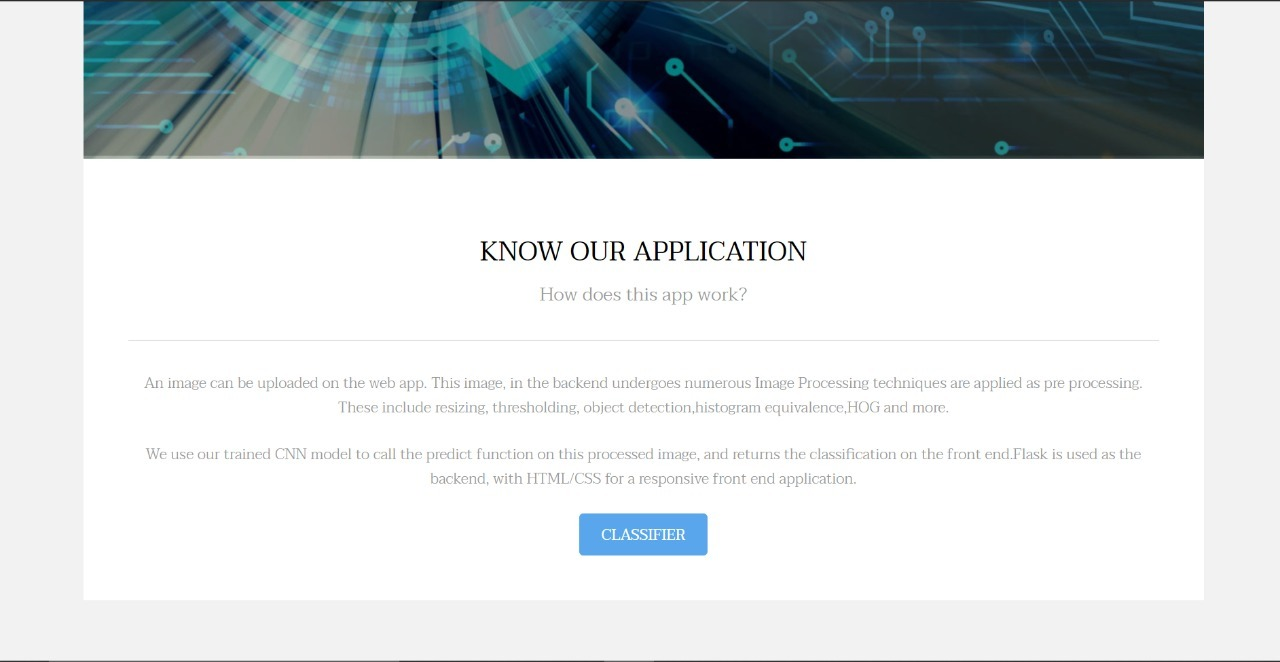


*Fig: Counting coin values*

**Web Application**

1. The web application was deployed with a **FLASK** backend and a responsive **HTML/CSS** frontend.

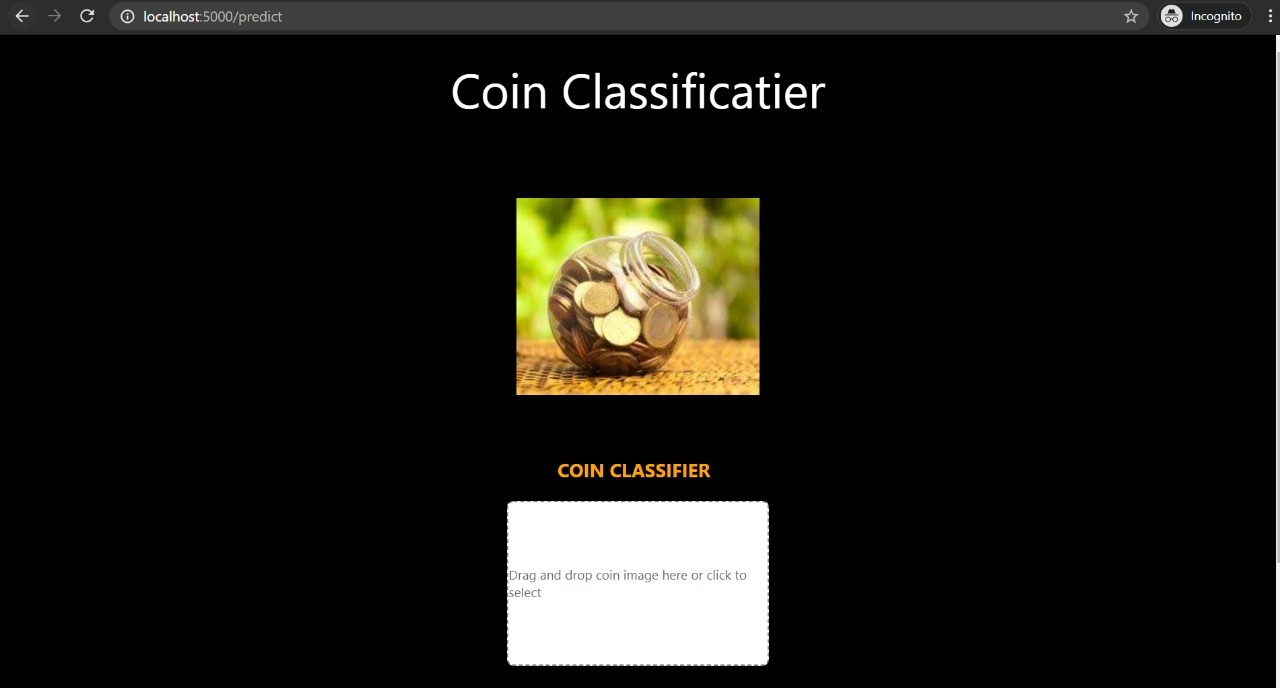






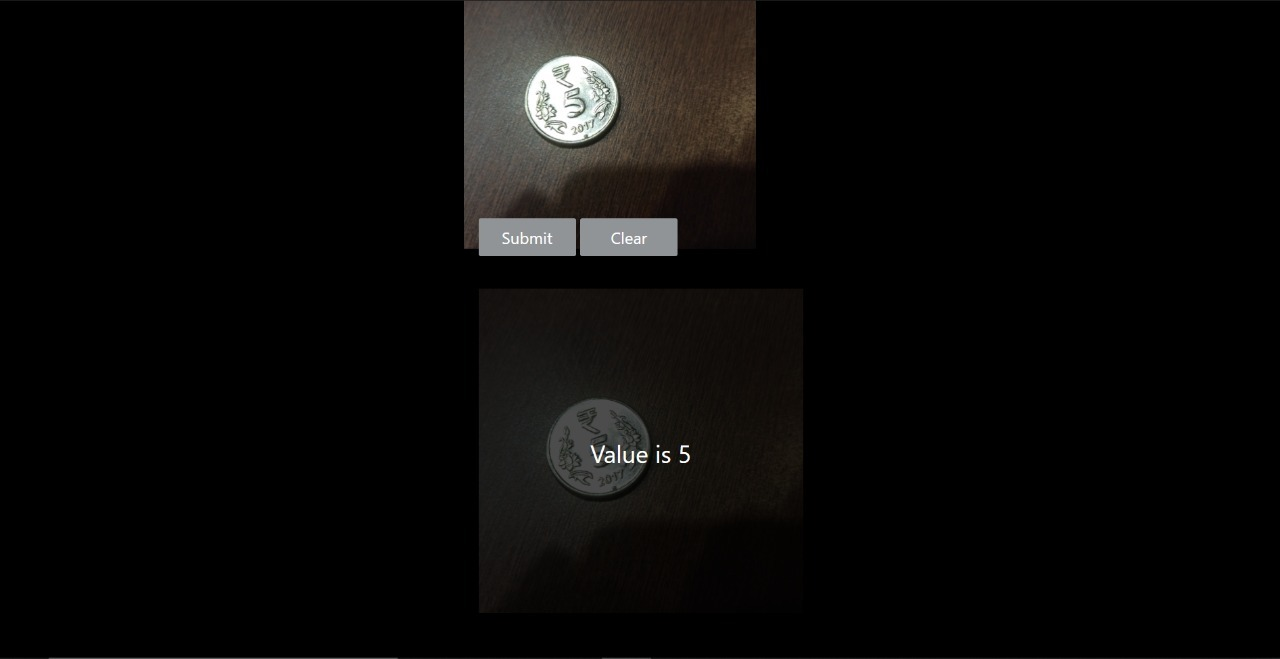
*Fig: Application UI*

1. The **input** for the web application is a coin image which can be selected from the user’s desktop, or can be dragged and dropped onto the web application.



*Fig: Classification input page*

1. On recieving the input image, image processing and coin value detection and calculation occurs in the backend.
2. The resultant **OUTPUT** is the value of the coin displayed on the web application, as recieved from the backend operations.



*Fig: Output*

**CONCLUSION**

We were able to successfully deploy our model as a web application by pipelining the image acuiring, pre-processing,coin recognition and value counting operations on the flask backend. The app inputs a coin image from the user and outputs the value of the Indian coins in the image.

We are able to recognize coins from an image with an accuracy of 78%. This accuracy is lower that expected as it is based on a model trained with a dataset of around 600 images taken from a pre-existing database, as mentioned above. To increase the accuracy, we have created our own database extending our current database to around 1700 images, hence expecting a significant increase in accuracy.

**REFERENCES**

1. [http://emis.impa.br/EMIS/journals/IJOPCM/Vol/09/IJOPCM(vol.2.2.8.J.9).pdf](https://meet.google.com/linkredirect?authuser=0&dest=http%3A%2F%2Femis.impa.br%2FEMIS%2Fjournals%2FIJOPCM%2FVol%2F09%2FIJOPCM(vol.2.2.8.J.9).pdf)
2. [https://arxiv.org/ftp/arxiv/papers/1312/1312.6599.pdf](https://meet.google.com/linkredirect?authuser=0&dest=https%3A%2F%2Farxiv.org%2Fftp%2Farxiv%2Fpapers%2F1312%2F1312.6599.pdf)
3. [http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.643.9252&rep=rep1&type=pdf](https://meet.google.com/linkredirect?authuser=0&dest=http%3A%2F%2Fciteseerx.ist.psu.edu%2Fviewdoc%2Fdownload%3Fdoi%3D10.1.1.643.9252%26rep%3Drep1%26type%3Dpdf)
4. [https://www.ijarcce.com/upload/2016/november-16/IJARCCE%2092.pdf](https://meet.google.com/linkredirect?authuser=0&dest=https%3A%2F%2Fwww.ijarcce.com%2Fupload%2F2016%2Fnovember-16%2FIJARCCE%252092.pdf)
5. [https://www.ijert.org/research/automated-coin-recognition-and-counting-by-image-processing-techniques-IJERTCONV6IS15024.pdf](https://meet.google.com/linkredirect?authuser=0&dest=https%3A%2F%2Fwww.ijert.org%2Fresearch%2Fautomated-coin-recognition-and-counting-by-image-processing-techniques-IJERTCONV6IS15024.pdf)
6. [https://www.researchgate.net/publication/317014533\_Automated\_Coin\_Recognition\_-\_Approaches\_and\_Techniques](https://meet.google.com/linkredirect?authuser=0&dest=https%3A%2F%2Fwww.researchgate.net%2Fpublication%2F317014533_Automated_Coin_Recognition_-_Approaches_and_Techniques)
7. [https://patentimages.storage.googleapis.com/ef/df/09/a9783b0cb08205/US9053595.pdf](https://meet.google.com/linkredirect?authuser=0&dest=https%3A%2F%2Fpatentimages.storage.googleapis.com%2Fef%2Fdf%2F09%2Fa9783b0cb08205%2FUS9053595.pdf)
8. [https://inf.u-szeged.hu/~ssip/2006/projects/team6/Dagobert.pdf](https://meet.google.com/linkredirect?authuser=0&dest=https%3A%2F%2Finf.u-szeged.hu%2F~ssip%2F2006%2Fprojects%2Fteam6%2FDagobert.pdf)
9. [https://www.sice.jp/e-trans/papers/E1-23.pdf](https://meet.google.com/linkredirect?authuser=0&dest=https%3A%2F%2Fwww.sice.jp%2Fe-trans%2Fpapers%2FE1-23.pdf)
10. <https://www.ijmter.com/papers/volume-2/issue-2/algorithm-to-recognize-countsaperate-coin-from-image-using-matlab.pdf>
11. <https://pdfs.semanticscholar.org/d53a/8e7f7257e4a206929cae52fc8f5123ce83c8.pdf>
12. <https://ieeexplore.ieee.org/document/4555308>
13. <https://books.google.co.in/books/about/Image_Processing_Analysis_and_Machine_Vi.html?id=DcETCgAAQBAJ&redir_esc=y>
14. <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.259.1869&rep=rep1&type=pdf>
15. <https://www.researchgate.net/publication/317014533_Automated_Coin_Recognition_-_Approaches_and_Techniques>
16. <https://www.ijcaonline.org/archives/volume26/number4/3093-4244>
17. <https://link.springer.com/chapter/10.1007/978-3-540-74272-2_68>
18. <http://docplayer.net/40333157-Indian-coin-recognition-and-sum-counting-system-of-image-data-mining-using-artificial-neural-networks.html>
19. <https://ijesc.org/upload/6d1d273d2b56d9a325f1062a8699aa39.Coin%20Recognition%20System%20using%20Artificial%20Neural%20Network%20on%20Static%20Image%20Dataset.pdf>
20. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.672.2322&rep=rep1&type=pdf>