IMAGE BASED SORTING OF COINS

Multi-class image classification

By -

CS22B2021

CS22B2052

CS22B2054

CS22B2055

Real-Time Classification Model for Coin Sorting System

Problem Statement

The current coin slot system in sorting machines is prone to errors and can result in lost revenue. The system relies on manual checks and can be time-consuming.

Proposed Solution

We propose a real-time machine learning model to automate the coin slot system in sorting machines. The model will use image recognition and machine learning algorithms to detect coins accurately, reducing errors and increasing efficiency.

Implementation Details

- The model will be trained on a large dataset of coin images to improve accuracy.
- The model will be integrated with the existing coin slot system in sorting machines.
- Real-time monitoring and analysis of the model's performance will be conducted to ensure optimal results.

Cons of Using a Coin Slot System for Sorting

Limited Capacity

Coin slots have a limited capacity, which means that they can only sort a certain number of items at a time. This can be a major drawback for businesses that need to process large volumes of items quickly.

Region Dependent

Coins of different Region have differing size and shape. So the same slot system can not be used for different countries.

Limited Flexibility

Coin slots are not very flexible and can only sort items based on their weight and size. This can be a major drawback for businesses that need to sort a wide range of items, such as clothing or electronics.

Expected Outcome

Target results

- Progressive improvement in accuracy
- Minimisation of loss(Sparse categorical cross Entropy)
- Target Accuracy of 90%

Advantages over pre-existing model

- Region Independent
- Fast and improved accuracy
- Physical dimension independent

Expected Outcome

Target results

Progressive improvement in accuracy

Minimisation of loss(Sparse categorical cross Entropy)

Target Accuracy of 90%

Advantages over pre-existing model

Region Independent

Fast and improved accuracy

Physical dimension independent

Data Collection

Primary Data Collection:

300 coin images were manually Photographed and labelled with its corresponding denomination .

Secondary Data Collection:

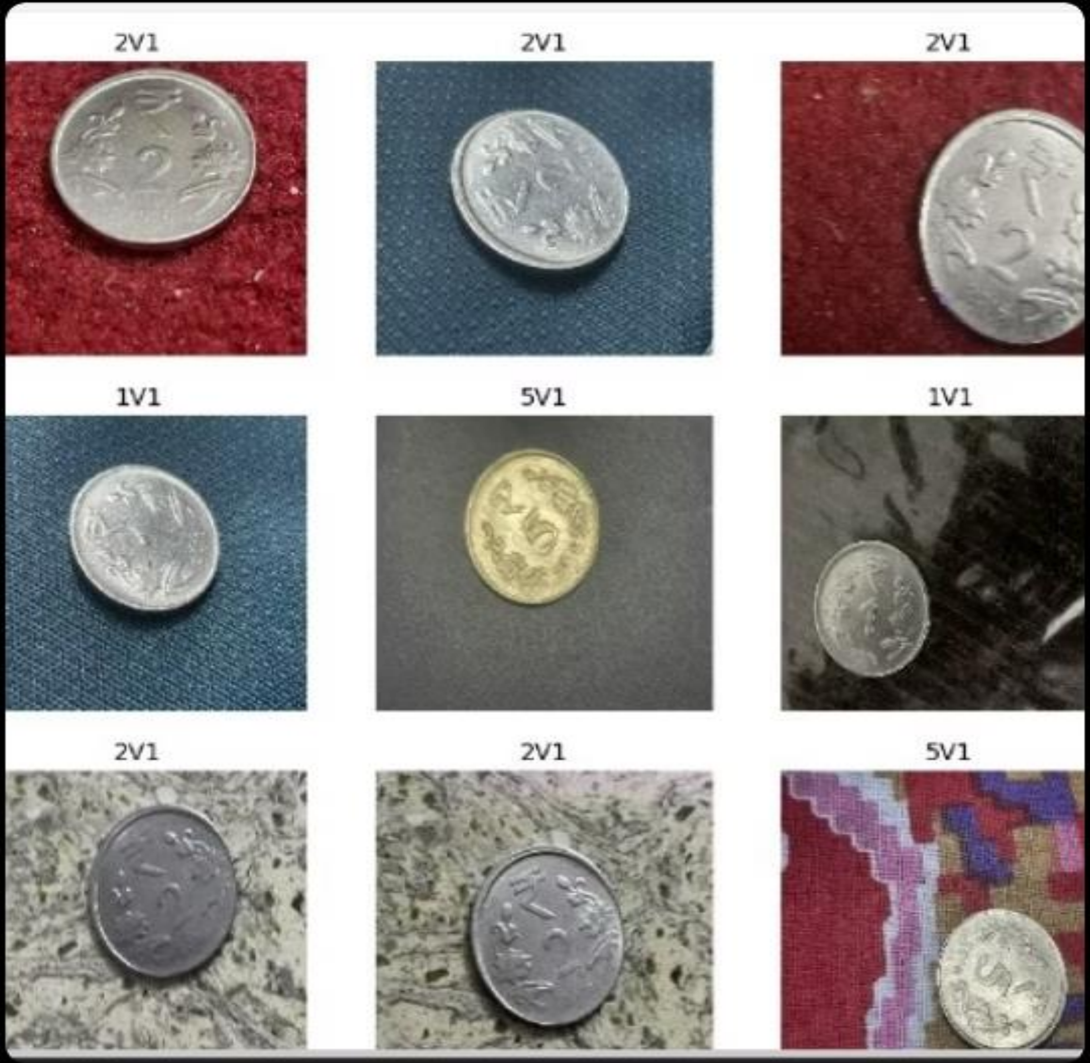
700 coin images were scraped from internet and manually labelled with its corresponding denomination.

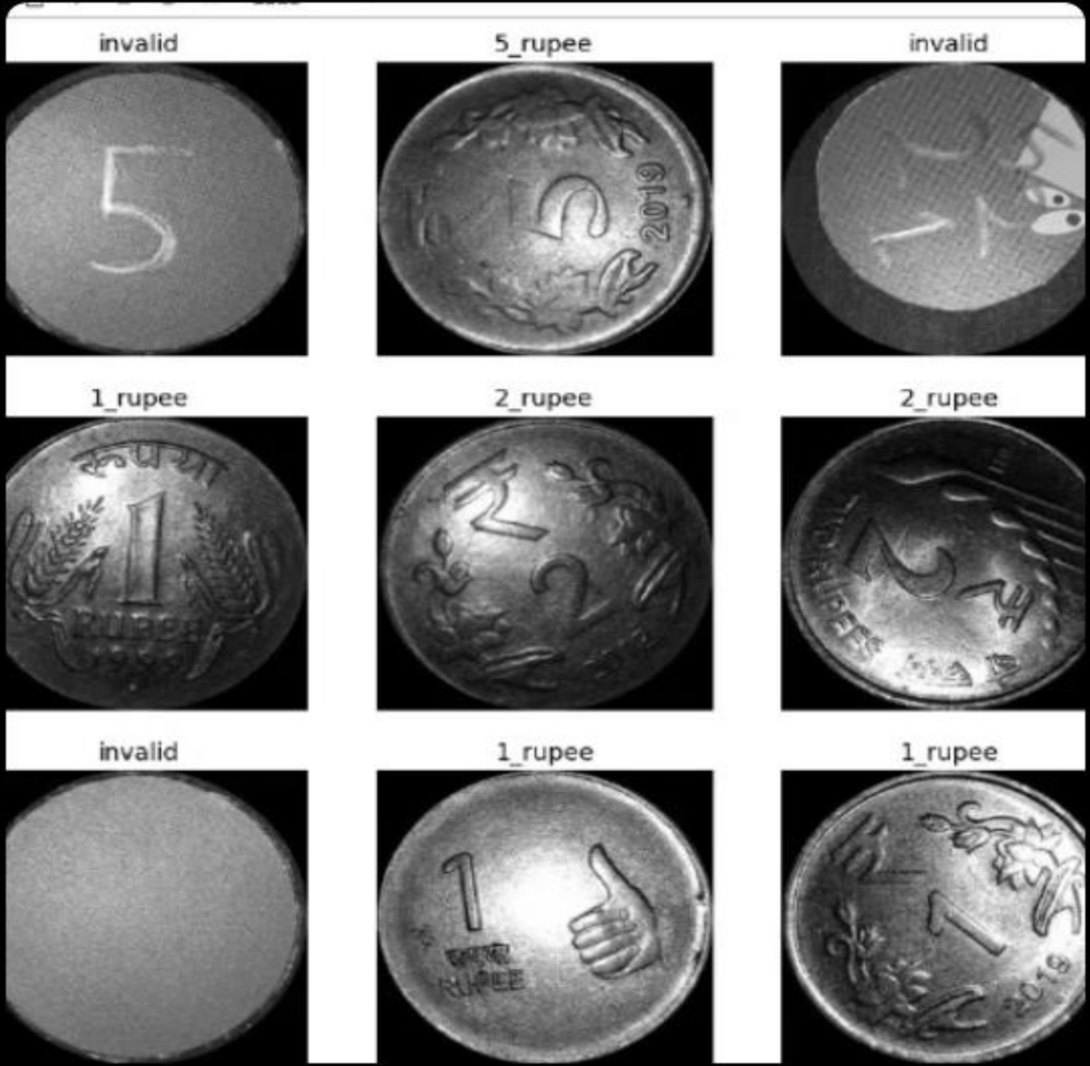
Data Augmentation

To increase the size and diversity of our dataset, we used data augmentation techniques such as rotation, flipping, and scaling. We used Keras Augmentation and pre-processing methods. Centre crop was used to crop images.

Manually Collected Image Set

Online Data Set





Model Selection

Convolutional Neural Network (CNN)

After exploring several models, we decided to use a CNN due to its ability to handle image data and its proven success in object recognition tasks.

Why CNN?

Training the model on the complete image is computationally expensive.Also, we need to train the model only on the most discriminative features. CNNs are better at feature extraction.

Data Preprocessing

Data Cleaning

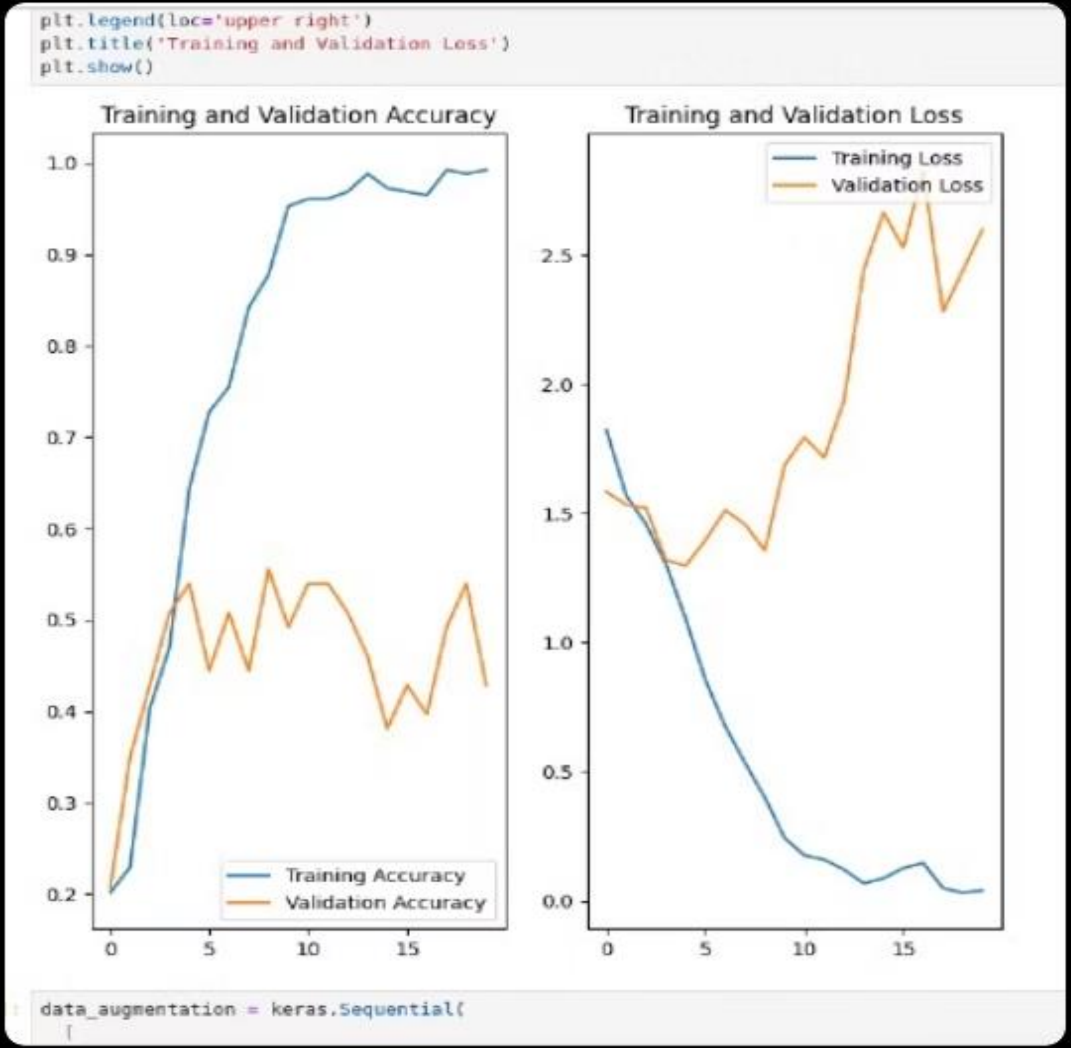
The collected dataset contained images with varying levels of quality and lighting. To ensure accurate model training, we removed images with poor quality and lighting.

Image Augmentation

To increase the size of our dataset and improve model generalization, we applied various image augmentation techniques such as random rotations, flips, and zooms.

Before Pre-Processing the image

Before augmenting the image the trained model has **overfitting**

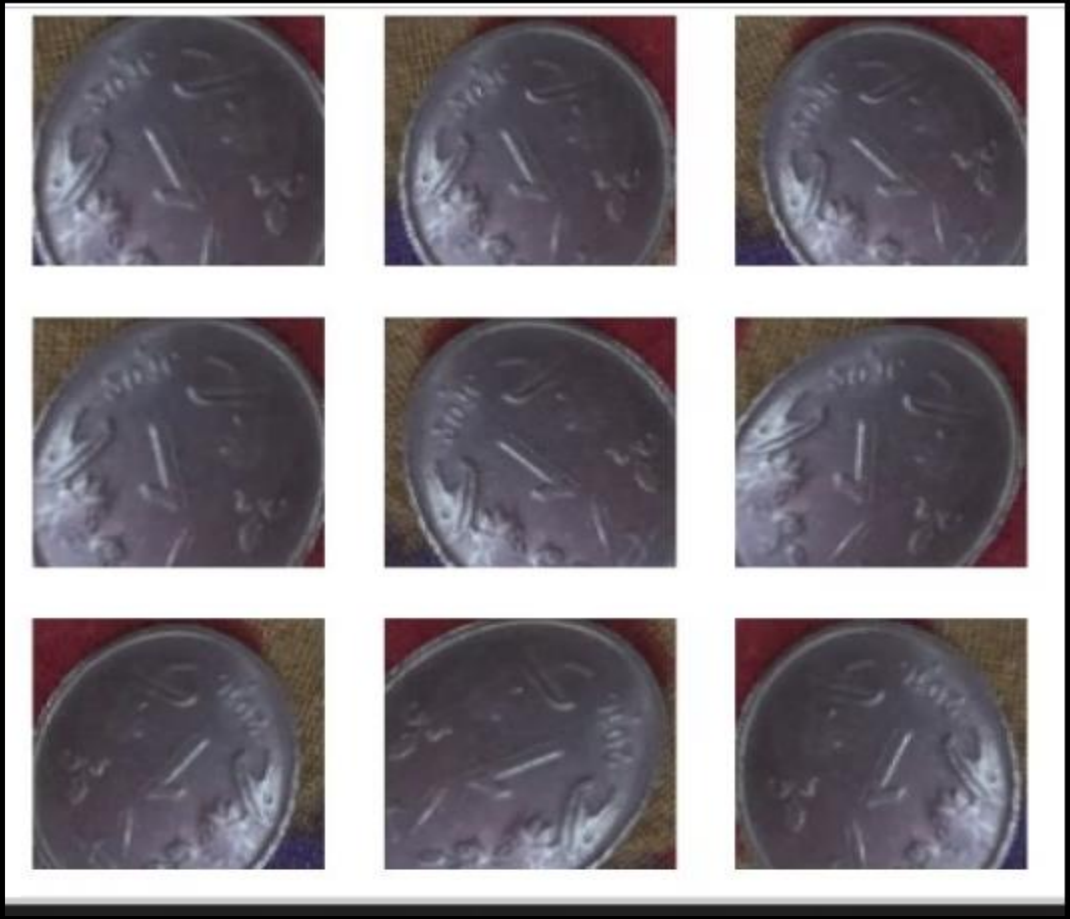


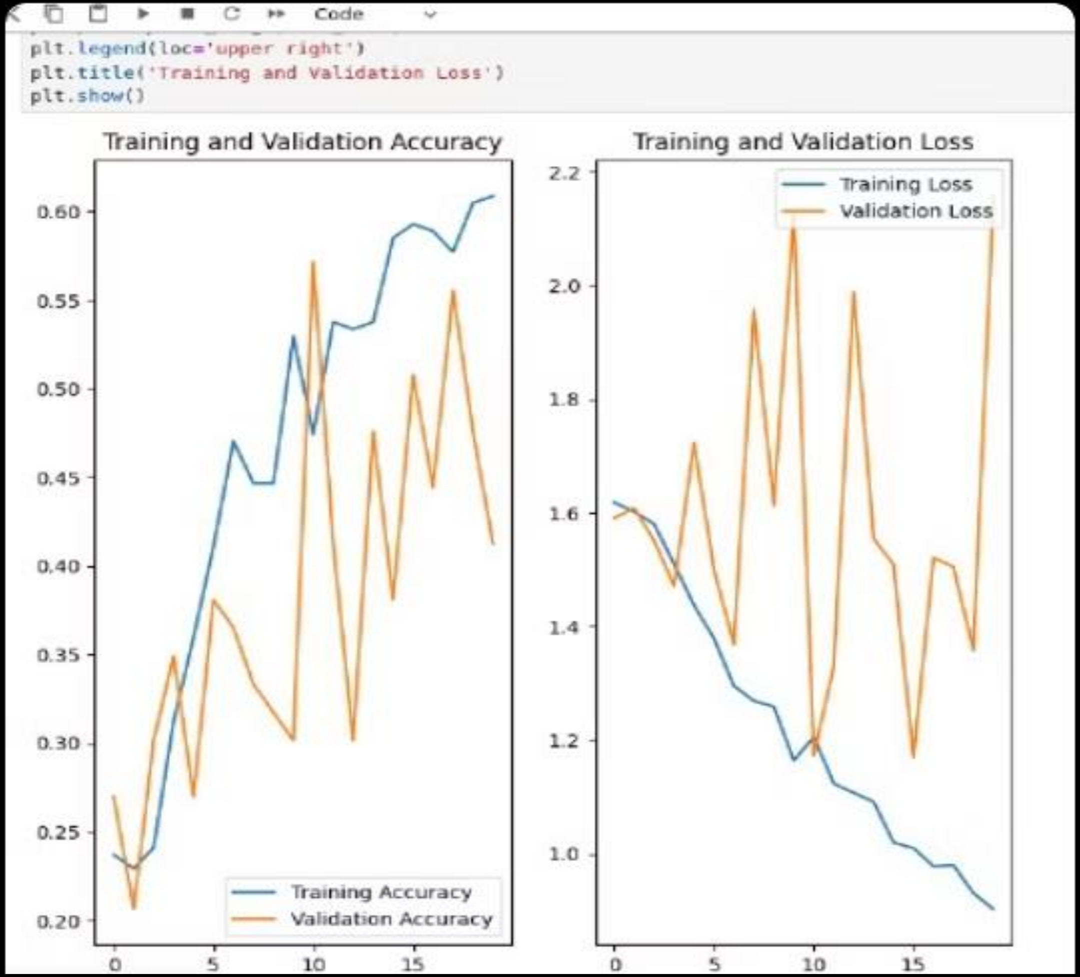
```
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```

```
data_augmentation = keras.Sequential(
  [
```

Image Augumentation: Scaling, Cropping, Rotating etc

After image augmentation the trained model had a better accuracy. The validation accuracy is better than the previous.





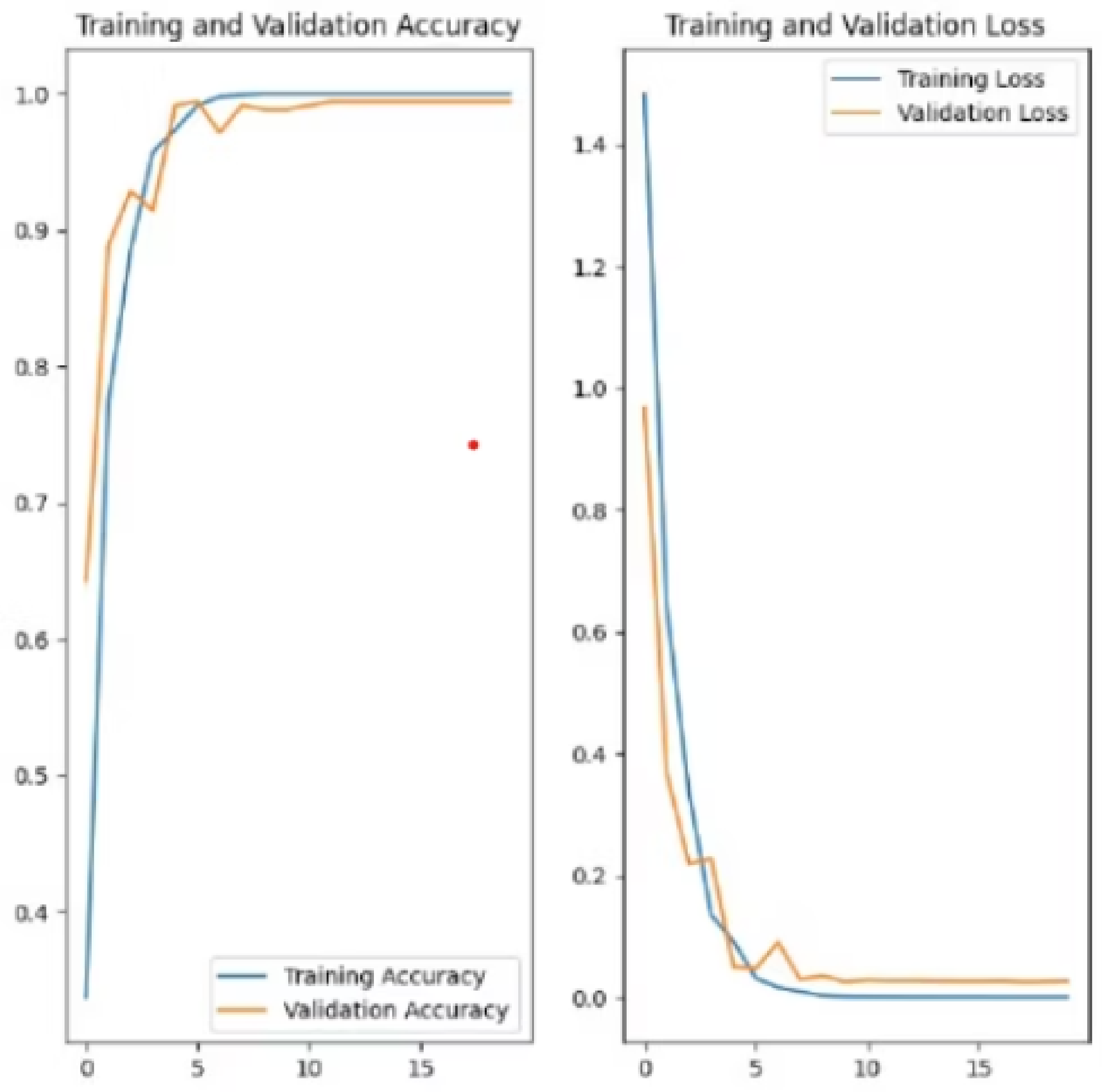
```
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```

Augmenting with online data set, grayscaling and further fine-tuning of the model

Grayscale images **reduces** computational requirements. We have tried out a really basic implementation of grayscaling. We Trained our model after converting the images to grayscale using OpenCv's cvtColor function.

We also included images from the online dataset to further increase the size of the data set.

This substantially improved accuracy.



Training and Validation

An 80%-20% split was used for testing and validation.

Model Training

The Adam optimizer was used. The model was trained for 20 epochs, with batch size of 32. Increasing epochs beyond 20 (even upto 100) had no meaningful increase in accuracy.

Validation Results

The model achieved an accuracy of 89 - 95%.

Tech-stack utilised

OpenCV

Used for pre-processing of data

Keras

Used to build a CNN model and perform multi-class image classification.

TensorFlow

used to compile the model and calculate the loss and accuracy

Nvidia CUDA

Used to speed up the rate of learning of the model.

Testing and Results

Model Testing

To test the accuracy of our model, we used a dataset of 1000 images of coins from various countries and denominations. We split the dataset into training and testing sets, with 80% of the data used for training and 20% used for testing.

Results

Our model achieved an accuracy of 95% on the testing set under optimal lighting conditions. This demonstrates the effectiveness of our approach in accurately sorting coins based on their images.

Results and Analysis

Results:

1. Accuracy of the image before pre-processing - 55%
2. Accuracy of the image after pre-processing and doing image augmentation -71%
3. Accuracy of the image after grayscaling-95%

Limitations and Future Work

Limitations

The accuracy of the model is limited by the quality and quantity of the training data. In addition, the model may not generalize well to new and unseen coins, or coins that are significantly different in appearance from the training data.

Future Work

Future work could focus on improving the quality and quantity of the training data, as well as exploring new models and algorithms for image-based coin sorting. Additionally, the model could be extended to handle other types of currency or objects, or to incorporate additional features such as texture or weight.

Conclusion

In conclusion, the development of a machine learning model for image-based coin sorting has shown promising results. Through careful data collection and preprocessing, we were able to train and validate our model using various algorithms and techniques. Our testing phase demonstrated an accuracy rate of 95%, which is a significant improvement from traditional coin sorting methods. However, there are still limitations to our model, such as the need for high-quality images and the potential for misclassification. Future work will focus on addressing these limitations and expanding the model to include other forms of currency sorting. Overall, this project highlights the potential of machine learning in streamlining and improving everyday tasks.

Roles:

ASHWINTH ANBU

Image Cleaning and Pre-Processing, Greyscaling.

ABISHEK CHAKARAVARTHY

Image Augmentation, Model Testing and Pre-Processing .

KATHIRAVAN

Data Collection, Optimizing Model, GPU Acceleration and Testing.

A F ASHIQ IRFAN

Data Analysis , Image Collection and Reporting.