A Novel Convolutional Neural Network for Classifying Indian Coins by Denomination

Yash Chauhan, Prateek Singh

Galgotias College of Engineering and Technology, Uttar Pradesh, India

Abstract

Coins recognition systems have humungous applications from vending and slot machines to banking and management firms which directly translate to a high volume of research regarding the development of methods for such classification. In recent years, academic research has shifted towards a computer vision approach for sorting coins due to the advancement in the field of deep learning. However, most of the documented work utilizes what is known as ‘Transfer Learning’ in which we reuse a pre-trained model of a fixed architecture as a starting point for our training. While such an approach saves us a lot of time and effort, the generic nature of the pre-trained model can often become a bottleneck for performance on a specialized problem such as coin classification. This study develops a convolutional neural network (CNN) model from scratch and tests it against a widely-used general-purpose architecture known as Googlenet. We have shown in this study by comparing the performance of our model with that of Googlenet (documented in various previous studies) that a more straightforward and specialized architecture is more optimal than a more complex general architecture for the coin classification problem. The model developed in this study is trained and tested on 720 and 180 images of Indian coins of different denominations, respectively. The final accuracy gained by the model is 91.62% on the training data, while the accuracy is 90.55% on the validation data.

Keywords: Convolutional Neural Network; Indian Coin; Classification; Denomination

*Corresponding author

Email address: Ch.yash042@gmail.com

1. Introduction

Coin Classification and recognition systems have seen tremendous application in the past few years due to the surge of machine automation in our society [1]. Such systems are deployed to reduce the odds of human error and fast-track the repetitive process of counting and sorting a large number of coins. While traditionally, such systems make use of the mechanical and electro-magnetic properties [2] of the coins to classify them, it is due to the enormous development in the field of computer vision and artificial intelligence that new systems are coming forth that utilizes robust neural networks to detect, segment and classify coins based on the images of the coins processed in real-time. Although the majority of the application stems from the sectors mentioned above yet due to the increasing reliability of coin classification systems and the increasing availability of computational resources for the average person, various studies have been done aiming to develop a system that can assist visually impaired people in identifying and classifying real currency [[3], [4]]. While a currency classification or identification system would contain both the banknote and coin classifier, this paper confines its scope towards the coin classification part of the system.

Coins are smaller and usually have less color difference between classes (different denominations). Also, coins are more durable than notes, and therefore they survive in the market for a more extensive period; hence the market has more variations of coins currently in use than banknotes. Therefore, a coin classification system must be robust to account for the greater variation and

deficiency of features in comparison to a banknote classification model. In this paper, the authors have proposed a deep learning model that can reliably classify coins based on the coin’s image. For training the model, a dataset was created with 900 images of different denominations (one rupee, two rupees, five rupees, ten rupees, and twenty rupees) Indian coins. It includes 210 images of one rupee coin, 120 images of two rupee coins, 270 images of five rupee coins, 180 images of ten rupee coins, and finally, another 120 images of twenty rupee coins. Each class contains different variations of the denomination to ensure that model has enough for training.

2. Literature Review

2.1. Almisreb, Ali & Saleh, Mohd.A. (2019). [5]

In this study, the authors trained and tuned some of the famous deep learning architecture, namely Alexnet, Googlenet, and Vgg16, using a transfer learning framework on an image dataset that contained 110 images in total of Bosnian banknotes, which included 11 different classes. The main characteristics identified in this study were currency texture, color, and forms. Vgg16 was the most accurate, and Googlenet was the least accurate at 88%.

2.2. Rahman, Mohammad & Poon, Bruce & Amin, Md Ashraful & Yan, Hong. (2014). [6]

The authors have used the Oriented FAST and Rotated BRIEF (ORB) methods developed in the lab of Open Source Computer Vision (OpenCV). The dataset only contained Bangladeshi banknotes. The paper also

describes the method of selecting thresholds for the algorithms. The experiment included test images with white background as well as images with a complex and ‘closer to real’ background to judge the practicality of the model. The proposed system was found to be 89.4% accurate on the white paper background and 78.4% accurate on a complex background.

2.3. A. Chalechale (2007). [7]

The author of this paper proposed a new approach for recognizing coins in an image. The approach is centered around the idea of measuring the similarity between full colour multi-component coin images. The significant advantage of this approach is that image segmentation before the detection is not required, and as segmentation is a very computationally expensive process by using this approach, we save a lot of computational time.

2.4. Ranjendra, P. & Anithaashri, T.P. (2020). [8]

This paper is oriented around Indian currency. The authors followed the same structure as Rahman et al. [2], wherein they created a dataset that contains both Indian banknotes as well as coins of different denominations and three state of the art deep learning frameworks, namely Alexnet, Googlenet, and Vgg16, were trained, tuned and compared. The result showed that Googlenet (with 22 convolution layers) was the least accurate with an accuracy of 88%, while Vgg16 (with 16 convolution layers) was the most accurate.

2.5. Abburu, Vidasamhitha & Gupta, Saumya & Rimitha, S. & Mulimani, Manjunath & Koolagudi, Shashidhar. (2017). [9]

A deep learning model was suggested in this research, which could classify both the country of origin and the denomination or value of a specified banknote. The model works only with paper notes and avoids dealing with coins altogether. The dataset in this study contains paper notes from 20 different countries and their denomination variants, and the model utilizes features like size, texture, color, and text in the image. The models were tuned to classify the notes with about 90% accuracy.

3. Methodology

Most of the recent studies which are focused on showing how the Deep learning model can be used to classify images of currency (banknote or coins) into various classes have used transfer learning framework wherein they use one or more of the popular CNN architectures like Googlenet, Alexnet, Vgg15 or Vgg16 [[10]; [8]]. While there is no apparent downside to using these frameworks, one can argue that such networks are not tailor-made for a specific problem. Instead, they are general in nature can be used in about any situation and the problem with that is that even by using 22 convolution layers, the Googlenet CNN can only get about 88% accuracy [8], since we cannot tweak the architecture of Googlenet without losing all the pre-

trained weights we have to settle for more epochs which in turn would result in higher computational time, or we can resort to collecting more data for more effective training. Both of the solutions are time-consuming and do not guarantee better accuracy.

Hence it is evident that for analyzing the effectiveness of CNNs in classifying currency, a novel architecture is needed that must be created and trained from scratch so that we can get the most accurate solution with a simpler and less computationally expensive neural network. For developing a new architecture, rigorous experimentation is needed, and for benchmarking different architectures, a baseline accuracy of 85% is chosen as this is the average accuracy reported by various studies that incorporated Googlenet CNN architecture.

Figure 1: Methodology for CNN iterations

4. Data

4.1. Accumulation

Most dataset available online does not include images of twenty rupee coins since it was recently approved by the government of India and was first minted in 2019. Hence for this study, we have created a new dataset [11] which includes images of Indian coins of all the currently circulating denominations, namely one rupee (₹1), two rupees (₹2), five rupees (₹5), ten rupees (₹10) and as well as twenty rupees (₹20). Since there are multiple designs for a coin of the same denomination, therefore it must be ensured that the majority of the designs are included in the training data so that we can make a robust model that can identify the denomination of the coin regardless of the change in design, an open access copy of the dataset is available on Kaggle with the same name. Hence, multiple design variations are included in our dataset. Some design variations included in the data are shown in Figures 2(a) and 2(b).

Figure 2: Variation of (a) One Rupee (₹1) & (b) Five Rupee (₹5) Indian Coin and (c) Different Viewing angles

The dataset contains images of 30 different coins, consisting of 7 types of one rupee coins, four types of two rupee coins, nine types of five rupee coins, six types of ten rupee coins, and another four types of twenty rupee coins. Each coin has been imaged from five different viewing angles as shown in Figure 2(c) and in three different lighting conditions i) Artificial Light, ii) Low Light, and iii) Natural Light. Hence, there are 15 possible combinations of viewing angle and lighting condition; furthermore, each combination has been sampled twice. Therefore, each coin has been imaged 30 times with varying light and viewing angles. The whole data set contains 900 such images, out of which 210 images are of one-rupee coins, 120 images are of two-rupee coins, 270 images are of five-rupee coins, 180 are images of ten-rupee coins, and finally, the rest 120 images are of twenty-rupee coins.

4.2. Pre-processing

4.2.1. Resize

Resizing is the first and arguably the most critical step in image pre-processing for training a neural network as a larger image size will have more information and can produce better correlation inside the neurons, but at the same time, a large size means that the time taken by the neural network to train will also be significantly high. Therefore, a particular size should be selected that can retain the maximum amount of information, yet it should be small enough to have a practical training time.

In their original form, all the images inside the dataset have a pixel size of 4000 x 3000 (12000000 or 12MP); this is a considerable size for an image that is not suitable for neural networks. So, after experimenting with different sizes, an image size of 320 x 240 was

selected, and all the images were resized while keeping the aspect ratio intact.

4.2.2. Normalization

Normalization is the process of rescaling the pixel intensity of an image to a predefined scale from the original scale. All images are in RGB format in the dataset, with pixel intensity ranging from 0 to 255 for each color channel. Image normalization was used to transform the pixel intensities from a scale of 0-255 to a scale of 0-1.

$$I_n = I_o \times \left(\frac{1}{255} \right) \quad (1)$$

Where I_n and I_o are the pixel intensity after and before normalization, respectively.

4.3. Augmentation

Training a Convolutional neural network requires a lot of data to achieve optimum accuracy. While our dataset contains a decent number of images, increasing the dataset further can effectively enhance the performance of the trained model. Data augmentation techniques are usually implemented to develop robust image classifiers using limited data [12] that can artificially generate images by employing various transformations such as random rotation, shift, flips, etc. Some transformations used in the study are shown in Figure 3.

5. CNN Architecture

5.1. Proposed Model

The architecture of the CNN model proposed in this paper consists of five convolution layers and five pooling layers while decreasing the size of the image and increasing the number of feature maps.

Figure 3: (a) Rotation Transformation and (b) Brightness Transformation

The output of the image processing part of CNN is then fed into a fully connected layer from which we get the final output decision of the model. The overall architecture and the positioning of the convolution and pooling layers are shown in Figure 4.

5.2. Convolution Layer

Each Convolution layer is a 2D convolution layer followed by a Rectified Linear Unit (ReLU) activation function, which has been shown to have good convergence properties, for example, scale invariance and 1-Lipschitz continuity [13]. A ReLU function takes the incoming signal from the neuron. It passes on only if the signal is positive. This means that if the incoming signal is negative, the signal will not proceed further in the neural network, and if it is positive, it will move forward to the next neuron unchanged.

The ReLU Activation Function:

$$f(x) = x^+ = \max(0, x) \quad (2)$$

Where x is the input to the neuron, the kernel size or the dimension of feature maps was fixed as $3 \times 3 \times 3$ (3×3 map with three color channels), and a $1 \times 1 \times 1$ stride was selected. Furthermore, feature maps increased from 16 (2^4) to 256 (2^8).

5.3. Pooling Layer

Max pooling layers are used between some of the convolution layers to reduce the dimensionality of the feature maps. The stride of the Max Pooling layer was of 2 pixels that were selected, and finally, a global average pooling layer was used instead of a flattening layer at the junction of the CNN and the fully connected layer to reduce the shape of the final output tensor coming from the last convolution layer to shorten the time required to train the fully connected layer.

5.4. Fully Connected Layer

The low dimensional feature maps are fed into the fully connected layer through a global average pooling layer that reduces the parameter's size and works as a flattening layer, converting the feature map to separate

neurons that make up the fully connected input layer. The ReLU activation function also follows this layer.

The last dense layer computes the final probabilities given by the model for each class. This layer utilizes the SoftMax function, also known as the normalized exponential function that normalizes the neural network's output to a probability distribution over the prediction classes.

The SoftMax Function:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}, \quad (3)$$

for $i = 1, 2, \dots, K$ and $z = (z_1, z_2, \dots, z_K) \in \mathbb{R}^K$

Using the SoftMax function in the last layer, we convert the output of the fully connected layer into an equivalent set of probabilities that belongs to each prediction class.

5.5. Training

The model weights were updated using the Adam (Adaptive Moment Estimation) Stochastic Gradient Descent optimization algorithm [14] so that the output corresponds with the given labels. At the same time, an adaptive learning rate was adopted that started with a value of 0.01 ($1e-02$) and would get reduced by a factor of 0.1 every time the training got hit by a learning plateau (with a patience level of 12 epochs). The Categorical cross-entropy function was selected as our model's optimization criteria or as the loss function.

Categorical cross-entropy (Loss Function):

$$\ell(x_i) = - \sum_{q=1}^Q 1\{y_i = c_q\} \times \log(P(y_i = c_q | x_i)) \quad (4)$$

Where Q represents the number of classes, $1\{y_i = c_q\}$ is the indicator that equates to 1 if y_i belongs to the class c_q and 0 otherwise. And $P(y_i = c_q | x_i)$ is the probability predicted by the model of x_i belonging to the class c_q .

Figure 4: Proposed CNN Architecture

6. Model Evaluation

The model was developed and trained on the Google Collaboratory environment with a GPU backend support (NVIDIA Tesla K80). The model was trained for a total of 150 Epochs and was found to achieve an accuracy of 91.62% on the training data, while on the validation or test data, the accuracy was found to be 90.55%. The high level of agreement between the accuracy on training and test data signifies that the model is not overfitting. Figure 5(a) and (b) show the accuracy and loss curve over training epochs, respectively, while Figure 5(c) shows the adaptive learning rate during training.

The accuracy metric is not sufficient to evaluate the performance of a multiclass classifier; rather, the performance of each class should be judged separately. Figure 6 shows the confusion matrix for the classifier, and it is evident from the confusion matrix that the model performed best at classifying Five rupees (₹5) and Ten Rupees (₹10) coins while it struggles the most to classify Twenty Rupee (₹20) coins, this phenomenon is confirmed by looking at the accuracy scores for each class separately shown in Table 1.

Given that evaluation metrics like positive predictive value (PPV) or precision and negative predictive value (NPV) can only be used in binary classification, therefore using the ‘one vs. rest’ approach, the multiclass confusion matrix can be converted into five binary confusion matrix (one for each class). For example, considering class 1 (One rupee), all the correctly classified images will be termed as True Positives (TP), all the images that were correctly classified as not One rupee will be termed as True Negative (TN), and all the images that were incorrectly classified as One rupee will be termed as False Positive (FP). Lastly, all incorrectly classified images as not One rupee will be termed False Negative (FN). Using this approach, TP, TN, FP, and FN can be calculated for each class, which will be used to calculate the positive predictive value (PPV) and the negative predictive value (NPV). The calculated values for TP, TN, FP, FN, PPV, and NPV for each class are shown in Table 2.

Figure 5: (a) Accuracy, (b) Loss, and (c) Learning Rate

Table 1:

Accuracy Score for each class				
Class	Total Samples	Correct Predictions	Wrong Predictions	Acc. (%)
One	42	39	3	92.8
Two	24	20	4	83.3
Five	54	51	3	94.4
Ten	36	34	2	94.4
Twenty	24	19	5	79.1
Total	180	163	17	90.5

Confusion Matrix					
True Class Label	One	Two	Five	Ten	Twenty
	39 21.67%	0 0.00%	3 1.67%	0 0.00%	0 0.00%
	4 2.22%	20 11.11%	0 0.00%	0 0.00%	0 0.00%
	3 1.67%	0 0.00%	51 28.33%	0 0.00%	0 0.00%
	0 0.00%	0 0.00%	0 0.00%	34 18.89%	2 1.11%
	0 0.00%	0 0.00%	0 0.00%	5 2.78%	19 10.56%
Predicted Class Label					

Figure 6: Confusion Matrix

Table 2:

Binary Classification Metrics using 'one vs rest' approach

Class	True Positive (TP)	True Negative (TN)	False Positive (FP)	False Negative (FN)	PPV (%)	NPV (%)
One Rupee (₹1)	39	130	8	3	82.98	97.74
Two Rupee (₹2)	19	156	0	5	100	96.89
Five Rupee (₹5)	51	123	3	3	94.44	97.62
Ten Rupee (₹10)	34	140	4	2	89.47	98.59
Twenty Rupee (₹20)	20	154	2	4	90.91	97.47

$$\text{Positive Predictive Value (PPV)} = \frac{TP}{TP + FP} \quad (5)$$

$$\text{Negative Predictive Value (NPV)} = \frac{TN}{TN + FN} \quad (6)$$

7. Conclusion and Future Prospect

In this study, we have created a new dataset of 900 images that can be used to train a machine learning model to classify Indian coins based on their denomination. Out of the 900 images, 720 images were used for training the model, and 120 images were set aside for validation. A CNN architecture was developed from scratch that contains five convolution layers. The accuracy gained on the training and validation sets was 91.62% and 90.55%, respectively. The validation accuracy (90.55%) was greater than the average accuracy of Googlenet CNN (88%) reported by various previous studies. We also confirmed that a relatively simple (4 convolution layers) CNN architecture specifically made for a particular problem (Coin Denomination Classification) was more efficient than a more complex (22 convolution layer) general-purpose architecture (Googlenet).

The model developed in this study was later evaluated with both multiclass classification metrics like Over-All-Accuracy and Per Class Accuracy as well as binary classification metrics like Positive Prediction Value (PPV) and Negative Prediction Value (NPV)). It was observed that the model performed best at classifying Five rupees (₹5) and Ten Rupee (₹10) coins while it was worst at classifying Twenty Rupee (₹20) coins. This phenomenon can be explained by the fact that the dataset contains only 120 images of twenty rupee coins while there were 270 and 180 images of five and ten rupee coins, respectively.

In this study, the class imbalances in the dataset were neglected; therefore, for future improvement of the model, applying thresholding to compensate for prior class probabilities is advised for increasing the overall accuracy of the model. Moreover, since the model's architecture is relatively shallow with only five convolution layers adding more convolution blocks might yield more accurate results.

References

- [1] Chen, Chi & Chen, Bo & Chang, Anthony Y. (2020). A Coin Recognition System Towards Unmanned Stores Using Convolutional Neural Networks.
- [2] Muppalaneni, Naresh & Prathima, Ch & Sekhar, Akula. (2021). Ensemble Deep Learning for Brazil

- Currency Coin Prediction. IOP Conference Series: Materials Science and Engineering.
- [3] Yousry, Ahmed & Taha, Mohamed & Selim, Mazen. (2018). Currency Recognition System for Blind people using ORB Algorithm. International Arab Journal of Information Technology.
 - [4] Reddy, Jagan & Rao, K. (2020). Identification of Indian Currency Denomination Using Deep Learning. Journal of Critical Reviews.
 - [5] Almisreb, Ali & Saleh, Mohd.A.. (2019). Transfer Learning Utilization for Banknote Recognition: a Comparative Study Based on Bosnian Currency. Southeast Europe Journal of Soft Computing.
 - [6] Rahman, Mohammad & Poon, Bruce & Amin, Md Ashraful & Yan, Hong. (2014). Recognizing Bangladeshi Currency for Visually Impaired. Commun. Comput. Inform. Sci.
 - [7] A. Chalechale (2007), "Coin recognition using image abstraction and spiral decomposition" 9th International Symposium on Signal Processing and Its Applications, 2007, pp. 1-4.
 - [8] Rajendran, P. & Anithaashri, T.P. (2020). CNN-based framework for identifying the Indian currency denomination for physically challenged people. IOP Conference Series: Materials Science and Engineering.
 - [9] Abburu, Vidasamhitha & Gupta, Saumya & Rimitha, S. & Mulimani, Manjunath & Koolagudi, Shashidhar. (2017). Currency recognition system using image processing.
 - [10] A. U. Tajane, J. M. Patil, A. S. Shahane, P. A. Dhulekar, S. T. Gandhe and G. M. Phade (2018). "Deep Learning Based Indian Currency Coin Recognition," 2018 International Conference On Advances in Communication and Computing Technology (ICACCT).
 - [11] Yash Chauhan, Prateek Singh. (2021). Indian Coin Denomination Dataset (ICDD). IEEE Dataport. <https://dx.doi.org/10.21227/t63t-5j24>.
 - [12] Perez, L., & Wang, J. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. ViXra.
 - [13] Suzuki, T. (2018). Fast generalization error bound of deep learning from a kernel perspective. Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics.
 - [14] Kingma, D.P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. CoRR, abs/1412.6980.
-