A

DEEP LEARNING MINI PROJECT REPORT

ON

"INDIAN CURRENCY RECOGNITION"

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY IN PARTIAL

FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE

BACHELOR OF COMPUTER ENGINEERING

BY

MR. SINHA KRISHNAGOPAL RAJESH KUMAR
MR. KHAIRNAR GAURAV PRAKASH
MS. BHATJIRE VAISHNAVI DEEPAK

UNDER THE GUIDANCE OF

PROF. DEVIDAS S. THOSAR

DEPARTMENT OF COMPUTER ENGINEERING,

SIR VISVESVARAYA INSTITUTE OF TECHNOLOGY,

NASHIK

A/P. CHINCHOLI, TAL. SINNAR, DIST.NASHIK, MH, INDIA-422102

YEAR 2022-23

Mini Project – Indian Currency Recognition

Title:- Indian Currency Recognition Using Deep learning.

Objectives :-

- Understand the concepts of Deep Learning
- To recognise Currency using Deep Learning.

Software Required :-

- Any Operating System
- Editors: v s studio / sublime / Jupyter Notebook
- Python

Theory:-

Introduction

In the last eight years more than 3.53 lakh cases of counterfeit currency detection in India's banking channels is heighten according to latest government reports. The practice of counterfeiting became more refined with the arrival of paper currency. The Indian Government has taken a astonishing stride of demonetizing 500 and 1000 Rs. notes. Prime Minister Shree. Narendra Modi stated that one of the cognition for this policy was to counter the climbing menace of counterfeit Indian Currency notes. However, the Indian banks acknowledged an all- time peak amount of fake currency and also noticed an over 480% increment in doubtful transactions after demonetization, a first ever report on questioning credits ended in the wake of 2016 notes ban has discovered. The Reserve Bank of India (RBI) is the only one which has the singular authority to issue bank notes in India. The RBI being the highest monetary authority in the country prints the currency notes of all denominations from Rs.2 to 2000. Several security features have been published by the RBI so that the counterfeit notes can be detected by the general public. However, distinguishing a counterfeit note just by visual per lustration is not an easy task. Moreover, an average person is unaware of all the security features. Developing applications which can detect a currency note to be counterfeit by a camera image can help solve this problem. Deep learning models have witnessed a tremendous success in image classification tasks. Our model proposes a binary image classification task with two classes-fake or real. The Deep RNN model we have built helps us detect the counterfeit note without actually

manually extracting the features of images. By training the model on the generated dataset, the model learns on it and helps us detect a counterfeit note.

Working Structure

In this system, we have to just feed image of currency or video, which contain currency, it can detect which currency is this. So now question is that what mechanism is running in behind of this? The answer is very simple we are using Deep learning & Image processing. However, another question is that why we are using Deep learning instead of machine learning? What is difference between Machine learning and Deep Learning? So, let's see all those answers. First, we need to understand what is Machine Learning and Deep Learning.

Strategy of Research

We seen 100 Rupees old note have green color but new note have purple color, 50 rupees new note have light green color but old have dark purple, there are many differences in new & old currency so with that parameters we can classify currencies. We captured images of currency from different angles with different side objects.

We split the images in 80:20 ratio for training & testing. Every Image size is 1200*800. We seen every country have their own pattern, In India every currency notes have different colors, looks and size.

Machine Learning

Machine learning is the technique in which we perform a scientific analysis on different algorithm and statistical models. In this approach we use heuristic algos and approach like Linear Regression, SVM, Naïve Bayes, KNN, KMeans Clustering. These all are the different types of working algorithms of Machine Learning [1]. But the question isthat how Machine learning is working? Simple answer is that it takes data as a input, find a patterns in it using algorithms and heuristics approach and give predictions of our test data based on input training data and algorithms learning pattern. In machine learning we need to do feature extraction by self, then we need to train the model. But this approach is not best and optimal solution for our application so we choose a Deep Learning approach for it. In Machine learning it always go for a solution but that solution could be a optimal or may be not also [2]

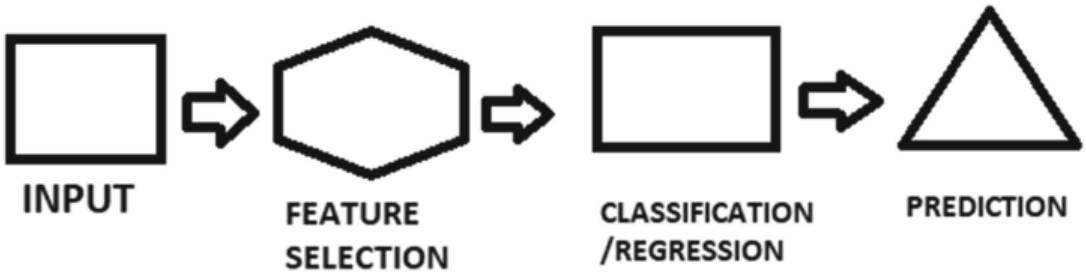


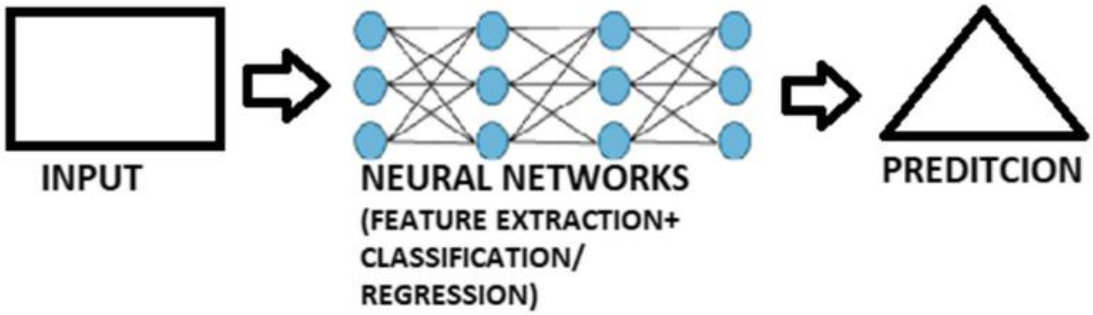
Fig. 1. Machine Learning

Deep Learning

We mimic the human learning pattern and it create the new algorithm, we call it artificial neural network, it has same structure as biological neural network. There are different types of neural network available; we can use it according to our work. However, most important thing is that to understand a structure of artificial neural Deep learning is the approach in that it mimic the humans learning patterns, we humans learn using Biological Neural Network(which is in-build in our body), In deep learning network. Now let see working of deep learning [3].

Working of Deep Learning

Deep learning is becoming the most evolving AI technique in 21st century. In this modern era, we have lots of data, and deep learning needs a data, that's why it becomes so popular in this modern era. We can see the usage of deep learning everywhere like social media,



*** WORKING OF DEEP LEARNING**

Fig. 2. Deep Learning

government, IT sector, Cinemas, Search engines. We humans already implemented Face recognition, self-driving car, auto drones and we are

continuously evolving like we are in infinity loop of mega evolution of AI. Now let us understand some algorithms that we used in our project [4].

Difference Between Machine Learning and Deep Learning

We seen working of Machine Learning & Deep learning. We seen in Machine learning we need to do feature extraction by self, even we are using old algorithms in it, but now Deep Learning is a hot field in AI. In this we are using Neural network concept. In this concept, we create an Artificial Neural Network (ANN), which works same as a Biological Neural Network that is available in human. So, we don't need to do feature extraction in Deep learning, it learns by itself.We have larger amount of Image data. Deep learning can perform better on images rather than Machine learning. Even it provides higher accuracy than machine learning. We can tune different parameters according to our data and model. Just one requirement is that we need a good GPU power for training. Moreover, another thing is that it takes longer time to train but it is worth [4].

Convolutional Neural Network

A Convolutional Neural Network is most popular Deep learning algorithm in which it takes an input image, assign weights and biases to various aspect according to the object in the image. In other algorithms we need to do lots of image processing and hand engineering to achieve the accuracy. But in CNN have the ability to learn these all the characteristics of images [5]. So, we don't need to do a lots of hand engineering in images, CNN will do for us. And also, we can achieve a good accuracy in our work.

Working of Convolutional Neural Network (CNN)

A Convolutional Neural Network have a n numbers of layers which can learn to detect different features from an image data, and the output of each processed image is used as the input to the next layer. The filters or we can say processing like edges, increase complexity, adjust brightness. CNN can perform feature identification classification of images, sound, audio, video and text [2].CNN is composed of an input layer, an output layer, and many hidden layers in network. These layers perform learning operation on the given data, Convolution function, Activation function and pooling are hidden layers.

• **Convolution** It have set of convolutional filters, which find features from images, so images pass through these all filters [2].

• **Rectified linear unit (ReLU)** is useful for mapping negative values into zero so it's maintaining positives values, so this is one kind of activation function, we

have many more choices like Sigmoid, hyperbolic tangent, but choosing a layer for a model is a depends on your data. It affects the accuracy [2].

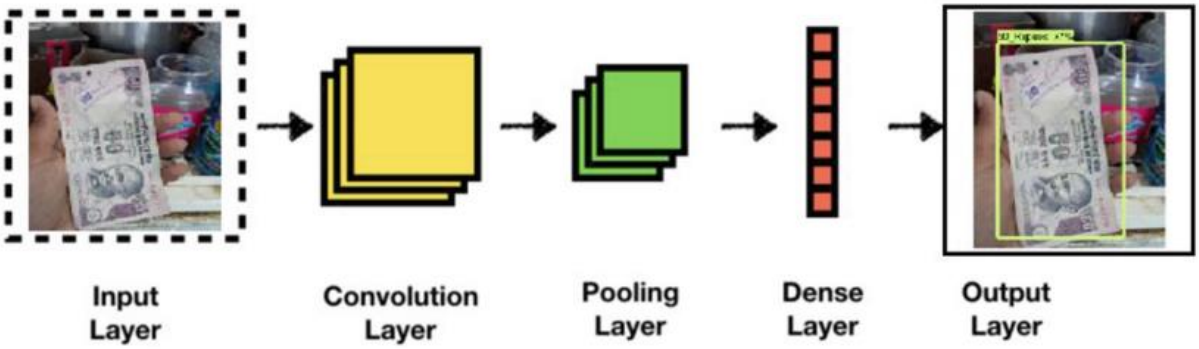


Fig. 3. Layers of CNN [8]

• **Pooling** performs the non-linear down stamping which can reduce the number of parameters then the network needs to learn and simplify the output [8].
• **Dense layer** is collections of neurons. It describes how neurons connected to the next layer of neurons (In short each neuron is connected to every neuron in the next layer).It is also known as Fully Connected layer [8].

These operations iterative on neural network layers, in which each layer learning to identify different features.

Training Time Increasing with GPU

A convolutional neural network is trained on hundreds, thousands, or even millions of images. When we have to work with lots of data then we can use GPUs for processing and computing. It can decrease the model training time and after training we can use our model in real world application.

Pre-trained Model

A pre-trained model, a name defining this term, it means model is trained on large dataset. You can use directly pre-trained model, in this just you have to feed your data and it can train on your data but it is already learnt on the large dataset, now you are re-training model so it will give you a better result. This learning approach is called Transfer learning. For example, you trained network on one lakhs images and now you are retraining it on 500 images for a classification purpose.

Tools and Technology Used

Jupyter Notebook

The Jupyter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at Project Jupyter.

Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

Python Programming Language

Python is programming language like C, C++, C#. It is an interpreted high-level programming language. Guido van Rossum created it, and it was first released in 1991. Python coding style is so comfortable for programmer, it has indentation feature so our code structure always stays good and understandable for other. Python is dynamically type language and also, it's have garbage collection so we don't feel to worry about unnecessary garbage in programming, It supported Procedural programming, functional programming and also object oriented programming. With OOP user can write clear & logical code for small- and large-scale project.

Tensorflow

Tensorflow is a Deep learning library for implement neural network algorithms in our work. In February 2017 version 1 released. It is an open source library. In deep learning, we do lots of math and numerical calculation but with Tensorflow API we can do this thing easily. It is just like feed your data, choose your model, number of layers and activation and starts a training and wait for an outcome. It is developed by Google brain team. It runs on CPUs and GPUs, we can use it in mobile and embedded platforms, it also can process on TPUs, which is hardware to do math on tensors.

Keras

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

Up until version 2.3, Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML. As of version 2.4, only TensorFlow is supported. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was

developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet is also the author of the Xception deep neural network model.[6]

Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

Matplotlib was originally written by John D. Hunter. Since then it has had an active development community and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012 and was further joined by Thomas Caswell. Matplotlib is a NumFOCUS fiscally sponsored project.

Matplotlib 2.0.x supports Python versions 2.7 through 3.10. Python 3 support started with Matplotlib 1.2. Matplotlib 1.4 is the last version to support Python 2.6. Matplotlib has pledged not to support Python 2 past 2020 by signing the Python 3 Statement.

NumPy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The predecessor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. NumPy is a NumFOCUS fiscally sponsored project.

Pandas

pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals. Its name is a play on the phrase "Python

data analysis" itself. Wes McKinney started building what would become pandas at AQR Capital while he was a researcher there from 2007 to 2010.

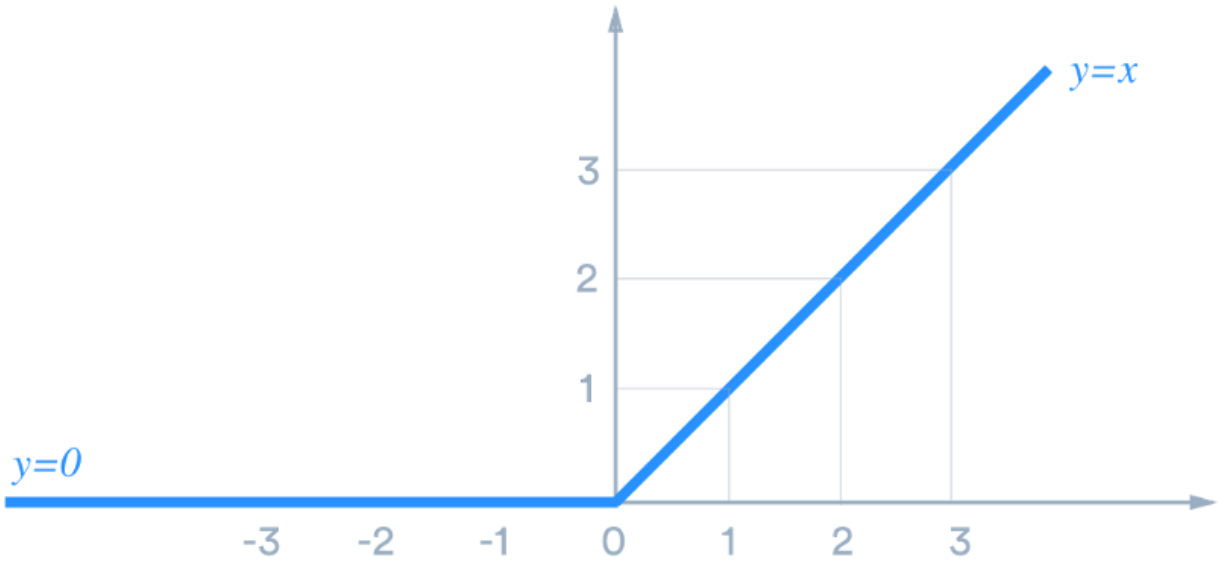
Mathematical Module

ReLu

ReLu is a non-linear activation function that is used in multi-layer neural networks or deep neural networks. This function can be represented as:

$$f(x) = \max(0, x) \qquad (1)$$

where x = an input value



According to equation 1, the output of ReLu is the maximum value between zero and the input value. An output is equal to zero when the input value is negative and the input value when the input is positive. Thus, we can rewrite equation 1 as follows:

$$f(x) = \begin{cases} 0, if\ x < 0 \\ x, if\ x \geq 0 \end{cases} \qquad (2)$$

where x = an input value

Dataset Gathering

Data gathering is the most important part of any research. We gathered number of images for every currency. So, we started gathering dataset with old currency. For a quality images currency should be new. So, I collected currencies from ATM. Because from ATM we get a new note without any fold with clean paper. Then we make a one excel file in which we define what will be our label name & what will be our image count for particular one currency. First, I started with old currencies then click the photos from mobile, compress the images in zip. You can

see structure of imaged below.After gathering of images, we converted our images into one scale of height & width (1200*800) with the help of python code. So now first step is done, now 2nd and most important step is image labelling. So, let see and understand image labelling in brief.

Dataset Labelling

Labelling is terminology or concept in which we give some particular name to a particular object. So in labelling we create a rectangle box which determines the object, then we give a name to that object. We are using labelImg tool for a image labelling. Then we give labels to every image which we gathered for our dataset.

Currency Dataset

In this part, we are discussing how we made our currency dataset, we captured images from 48 MP camera, every image has 4k*3k dimensions, let see some samples of our currency dataset.We used black marble as a background because of image clarity.

Structure of Dataset

For Experiment below are dataset structure with Label, training images and testing images numbers (Table 1).

Table 1. Dataset structure used with model

Label	Number of training images	Number of testing images
10_new _note	201	50
10_old_note	201	52
100_new_note	204	47
100_old_note	201	50
50_new_note	202	51
50_old_note	201	54
200_note	210	55
20_new_note	201	56
20_old_note	200	51
2000_note	201	48

Workflow

After image labelling, we created training and testing subfolder in images main folder, then we spilt our images into training & testing. Our splitting ratio is 80:20.

After that, we created .csv file for both training & testing, it contains image label, name and pixels value of object, which we selected during image labelling, and we did this with our python code. After that, we have created Tensorflow record file using a python code. It created two files one is 'train.record' and 'test.record'. Now we need to define our deep learning model. So we have downloaded a faster RCNN v2 models & config file from TensorFlow's official GitHub page. It is already pre-trained model just we need to feed our images. After that, we need to edit config file of model and need to give a path of our images, record file. Then we did training using 'model_main.py' which is available in 'object_detection' repository. Our training taken almost 8 h, after the getting expected loss we stopped the training. On Tensorboard we observe the measures of loss and iteration continuously. After the training we need to generate a graph file (means model file). From that file machine can see what it is learnt. With code, we generate the inference graph of our training. Now it is time for testing. We created two programs, one for image recognition and another for a recognition from currency. Then we tested our models on different currencies. You can see the results & testing below.

Steps:

1. Importing Required Libraries

```
In [1]: from tensorflow.keras.layers import Input, Lambda, Dense, Flatten
        from tensorflow.keras.models import Model
        from tensorflow.keras.preprocessing import image
        from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
        from tensorflow.keras.models import Sequential
        from tensorflow.keras.layers import MaxPooling2D
        from tensorflow.keras.layers import Conv2D

        import numpy as np
        from glob import glob
```

2. Loading Data

Let's See The Data

```
In [2]: IMAGE_SIZE = [224, 224]

        train_path = 'Train'
        valid_path = 'Test'

In [3]:
        folders = glob('Train/*')

In [4]: folders
Out[4]: ['Train\\1Hundrednote',
         'Train\\2Hundrednote',
         'Train\\2Thousandnote',
         'Train\\5Hundrednote',
         'Train\\Fiftynote',
         'Train\\Tennote',
         'Train\\Twentynote']
```

3. Train Datagen

```
In [7]: from tensorflow.keras.preprocessing.image import ImageDataGenerator

        train_datagen = ImageDataGenerator(rescale = 1./255,
                                           height_shift_range=0.2,
                                           featurewise_center=True,
                                           rotation_range=0.4,
                                           horizontal_flip = True)

        test_datagen = ImageDataGenerator(rescale = 1./255,)
```

4. Training Set

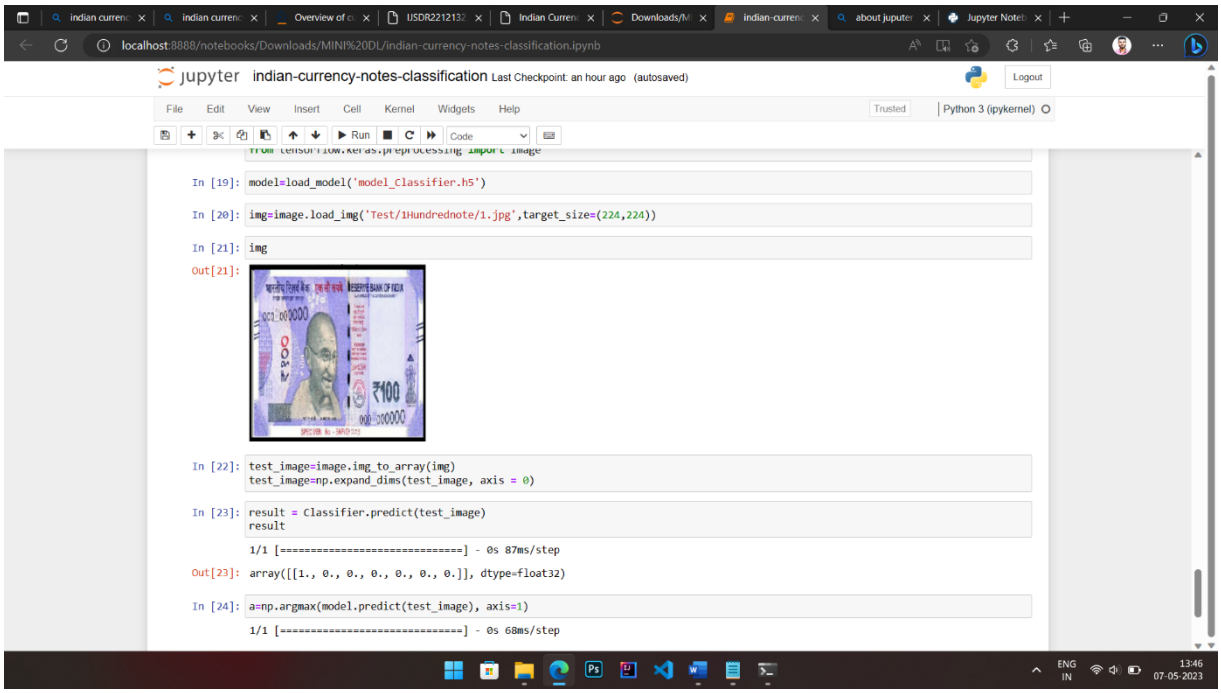
```
In [8]: training_set = train_datagen.flow_from_directory('Train',
                                                         target_size = (224, 224),
                                                         batch_size = 32,
                                                         class_mode = 'categorical')
Found 153 images belonging to 7 classes.

In [9]: test_set = test_datagen.flow_from_directory('Test',
                                                    target_size = (224, 224),
                                                    batch_size = 32,
                                                    class_mode = 'categorical')
Found 42 images belonging to 7 classes.
```

Results

Resnet v2 (Pre-Trained model) gives me 0.87 accuracy and 0.201 loss. We have trained model many times on prepared dataset, even we got a lower loss of 0.115 but at that loss I got very bad prediction ratio. Means my model is overfitted.
At current scenario, we are trying Restnet_50_coco model, but it is given memory error because of less GPU power, but we are trying to implement resnet50 for a better accuracy. Figure shows output of our trained model for different amount notes

```
Epoch 1/50
5/5 [==============================] - 9s 1s/step - loss: 7.1952 - accuracy: 0.1503 - val_loss: 3.4068 - val_accuracy: 0.1429
Epoch 2/50
5/5 [==============================] - 5s 998ms/step - loss: 2.5273 - accuracy: 0.1438 - val_loss: 1.8535 - val_accuracy: 0.1
667
Epoch 3/50
5/5 [==============================] - 5s 994ms/step - loss: 1.8741 - accuracy: 0.1503 - val_loss: 1.7927 - val_accuracy: 0.2
143
Epoch 4/50
5/5 [==============================] - 5s 970ms/step - loss: 1.7852 - accuracy: 0.3137 - val_loss: 1.6924 - val_accuracy: 0.2
857
Epoch 5/50
5/5 [==============================] - 6s 1s/step - loss: 1.5445 - accuracy: 0.4444 - val_loss: 1.4199 - val_accuracy: 0.4048
Epoch 6/50
5/5 [==============================] - 5s 1s/step - loss: 1.3659 - accuracy: 0.4771 - val_loss: 1.1477 - val_accuracy: 0.5714
Epoch 7/50
5/5 [==============================] - 5s 967ms/step - loss: 1.0876 - accuracy: 0.5882 - val_loss: 1.2231 - val_accuracy: 0.5
000
Epoch 8/50
```



Conclusion :

This project work prompts to an end that CNN model helps in identifying the currency denomination over a dataset. The prepared dataset comprises of Indian currency with the denominations 10, 20, 50, 100, 500 and 2000. And these images are obtained from [14] and other few were captured through smartphone. So it gives the most possible accuracy because we are concentrating on the innovations that can be conceptualized on the robustness and computation for recognizing the denomination of Indian Currency as a framework. The work is progressively effective to understand the methodologies and calculations engaged with the framework

References

1. Zhang, Q., Yan,W.Q., Kankanhalli,M.: Overview of currency recognition using deep learning.
J. Banking Financ. Technol. 3(1), 59–69 (2019)

2. Xu, L., Ren, J.S., Liu, C., Jia, J.L.: Deep convolutional neural network for image deconvolution. In: Advances in Neural Information Processing Systems, vol. 2, pp. 1790–1798 (2014)

3. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Neural Information Processing Systems, pp. 91–99 (2015)

4. Kurkova, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I.: Artificial neural networks and machine learning—ICANN 2018. In: Proceedings of 27th International Conference on Artificial Neural Networks (part II), vol. 11141, pp. 4–7 (2018)

5. Saha, S.: A comprehensive guide to convolutional neural networks — the ELI5 way, 15 December 2018. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional neural-networks-the-eli5-way-3bd2b1164a53. Accessed 2 Feb 2020

6. Xu, Y.: Faster R-CNN (object detection) implemented by Keras for custom data from Google's Open Images Dataset V4 (2018). https://towardsdatascience.com/faster-r-cnn-obj ect-detection-implemented-by-keras-for-custom-data-from-googles-open-images-125f62 b9141a. Accessed 2 Feb 2020

7. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)

8. Convolutional neural network. https://www.mathworks.com/solutions/deep-learning/convol utional-neural-network.html#howitworks. Accessed 2 Feb 2020