PROJECT REPORT

Submitted in fulfilment for the J-Component of

ITA6004-Soft Computing

Image Recognition using CNN for Indian Currency

Submitted To

Dr. Anitha A.

Associate Professor Sr.

School of Information Technology & Engineering

Vellore Institute of Technology

Vellore, India

Submitted By

Priyanka Sarkar

22MCA0082

School of Information Technology & Engineering

Vellore Institute of Technology

Vellore, India

Abstract

The categorization of Indian money is critical for a variety of applications such as automated teller machines (ATMs), vending machines, and financial transaction systems. The project presents a deep learning-based strategy for automated categorization of Indian money denominations using image analysis techniques in this paper.
This study's dataset contains a complete collection of high-resolution photographs of Indian currency notes, including recently announced designs and variants. To boost the classification system's performance, data augmentation techniques including as rotation, scaling, and noise addition were used to expand and diversify the dataset.
We used a convolutional neural network (CNN) architecture for the classification model, which has shown great performance in image classification tasks.
The CNN model was trained on the supplemented dataset employing a large-scale computer infrastructure, allowing for fast training and parameter optimisation.
Extensive tests were carried out on a different test dataset to assess the performance of the suggested technique. The results show that the CNN-based model is successful at reliably categorising Indian rupee notes, with good accuracy and recall rates. The Keras library and Tensorflow’s other modules have been used in order to perform the classification. The dataset of images has been extracted and compiled from multiple free sources available online such as from Kaggle.
This study contributes to the field of image classification by proposing a strong and accurate solution for recognising and categorising Indian cash denominations. The suggested technique has the potential to improve the efficiency and reliability of financial systems, allowing for speedier and error-free transactions in a variety of scenarios. However, the model is not 100% accurate in its predictions. Value accuracy reaches as high as 93%.

Keywords: Image classification, CNN, VGG16, Indian Currency

I. Introduction

Deep learning for image classification is on the increase, and it is also a lucrative area. Convolutional neural network (CNN) is the most extensively used structure among them. Since CNN's success in classification, people have begun to experiment with it for picture segmentation and classification. TensorFlow is an open source software framework that performs numerical calculations using data flow diagrams. Google TensorFlow was formally opened on November 9, 2015, and Google TensorFlow version 1.0 was published in 2017, signifying its official use in the production environment [7]. The TensorFlow calculation framework can handle many deep learning algorithms such as CNN, RNN, and LSTM, but its use is not confined to deep learning and can also be used to build standard machine learning. Image preprocessing and other such acivities can be easily performed using Tensorflow and Keras.
In the context of Indian money, the Reserve Bank of India (RBI) offers new designs and variants on a regular basis to improve security features and prevent counterfeiting. As a result, building a reliable and efficient method for identifying Indian currency notes has become increasingly important. Traditional techniques of money recognition, which rely on handmade features and rule-based approaches, frequently struggle to manage Indian rupee's complex and diverse properties. This is when neural networks come into play. The use of effective Deep Learning algorithms can lead to a more straightforward approach to classification and identification tasks. CNN, which was first introduced by Yann LeCun, employs three layers: convolution, pooling, and non-linearity, which are the main aspects of deep learning for

pictures [6]. The tuning of the hyper-parameters is also critical to the overall effectiveness and efficiency of the network [6]. As such, we need to experiment with different values to get a near about perfect accuracy. Using neural networks available and provided by Tensorflow, a simple image classification model was constructed that can accept photographs of notes as input and provide the necessary note categorization output. Further sections deal with the background fundamentals required for the project, the proposed methodology for the classification that contains information about the model, the experimental and comparative analysis and finally the conclusion for the project.

Related Works

[1] The need of timely and precise plant disease identification for agricultural output is emphasised in the article. It emphasises the use of image processing and machine learning techniques for agricultural pest and disease detection, addressing the constraints of depending entirely on human expertise. The emphasis is on RGB photos owing of their low cost and widespread availability.
The report serves as a significant resource for academics by providing a complete assessment of current findings in this topic. It addresses the transition from handmade feature-based classifiers to deep learning techniques, which have demonstrated promising outcomes in terms of high recognition accuracies. However, there are still issues with generalisation to diverse datasets and field circumstances.
[2] The application of transfer learning (TL) in medical picture classification problems is investigated in this review study. The research emphasises the usefulness of TL in overcoming data shortage and conserving computing resources through a detailed examination of 121 qualifying papers. The bulk of research examined numerous models, with deep models like Inception being the most frequently used. The preferred TL methodologies were feature extraction and fine-tuning from scratch. The paper advises data scientists and practitioners to employ deep models as feature extractors to find a balance between computational costs and predictive power.
[3] This paper provides a thorough analysis and investigation of the use of deep learning in picture identification. It emphasises image recognition technology's theoretical and practical importance in improving computer vision and artificial intelligence. The study begins with a summary of image recognition technology development, followed by an introduction to three fundamental deep learning learning models: convolutional neural networks, recurrent neural networks, and generative adversarial networks. A comparison of these models is offered. Deep learning research findings in numerous image identification disciplines, such as face recognition, medical image recognition, and remote sensing image categorization, are also reviewed.
[4] This research investigates the link between Fully Connected (FC) layers and dataset properties in order to automate the process of learning a Convolutional Neural Network (CNN) architecture. The authors study the impact of deeper/shallower designs on CNN performance with regard to FC layers, as well as the impact of deeper/wider datasets on CNN performance. They intend to formalise concepts such as deep, shallow, and broad in the context of CNN architectures and datasets. The paper aims to provide insights and findings regarding the suitability of architecture depth for different dataset characteristics through experiments using four different CNN architectures of varying depths and four widely used image classification datasets, including CIFAR-10, CIFAR-100, Tiny ImageNet, and CRCHistoPhenotypes.
[5] Image categorization is a popular study issue in today's culture, as well as an essential path in image processing research. In machine learning, SVM is an extremely strong classification model. CNN is a sort of feedforward neural network with a deep structure and convolution

calculation. It is one of the most well-known deep learning algorithms. This study compares and analyses classic machine learning and deep learning image categorization techniques using SVM and CNN as examples.

[6] This work uses deep learning approaches to solve the challenge of 3D reconstruction from 2D photos. While traditional methodologies depended on picture registration and optimisation techniques, current advances in deep learning have allowed CNNs to be used for single-view reconstruction. However, the use of CNNs for image classification in the context of 3D reconstruction has received little attention. Based on the data, the study argues that by focusing on the best-performing categories, an image classification-based 3D reconstruction technique might be constructed.

[8] In this study, a spatial feature extraction technique using deep convolutional neural networks (CNN) is proposed for hyperspectral image (HSI) classification. HSI classification is a challenging task due to the large number of bands with strong correlations in both spectral and spatial domains, as well as limited training samples.

The study investigates the effect of seven different optimizers on the deep CNN model for HSI classification. The optimizers considered are SGD, Adagrad, Adadelta, RMSprop, Adam, AdaMax, and Nadam. The study introduces a spatial feature extraction technique based on deep CNNs for HSI classification and evaluates the performance of different optimizers. The results highlight the effectiveness of the deep CNN model with the Adam optimizer in this application.

[9] This research focuses on bank note identification using image processing methods. Various techniques and methods are studied for classifying bank notes from different countries. The experiments are conducted on separate image datasets for each country's bank notes. Deep learning, a machine learning technique, is employed to analyze and learn the features of authentic bank notes. The goal is to identify essential features using neural networks. Deep learning is particularly advantageous in processing large amounts of data, which is essential in the era of big data and real-world applications. The research examines bank notes from multiple countries by extracting their features in detail and analyzing them using deep learning techniques. The proposed system recommends a deep learning-based algorithm for detecting forged bank notes through general scanners. This algorithm can be used by individuals to prevent personal monetary damages caused by fake bank notes.

[10] This research paper explores the development history of convolutional neural networks (CNNs) and their applications in image classification. It analyzes various deep CNN architectures and highlights the importance of network architecture, optimization methods, and training techniques in improving image classification results. The paper also discusses the future prospects of CNNs in terms of architecture and training advancements.

[11] This paper presents a deep convolutional neural network (CNN) architecture for COVID-19 diagnosis based on chest X-ray image classification. The study addresses challenges related to dataset availability and quality by employing various preprocessing techniques, including dataset balancing, medical experts' image analysis, and data augmentation. Experimental results demonstrate an overall accuracy of up to 99.5%, showcasing the effectiveness of the proposed CNN model. The model achieves perfect accuracy (100%) when tested on a set of 100 X-ray images from the processed dataset and maintains a high performance of 99.5% when tested on an independent dataset of COVID-19 X-ray images. Comparative analysis with other machine learning algorithms further validates the superiority of the proposed model, particularly when evaluated on an independent testing set.

[12] This research paper introduces a dilated convolution algorithm for image classification. A dilated CNN model is proposed, which replaces traditional convolution kernels with dilated convolution kernels to reduce computational resources. The model is tested on

handwritten digit recognition and wide-band remote sensing image datasets. Results show improved accuracy and reduced training time compared to traditional CNN models. The proposed approach significantly enhances image classification performance.
[13] This article presents a novel framework for hyperspectral image classification using a simplified 2D-3D CNN. The framework combines a 2D CNN for spatial feature extraction and a 3D convolution layer for exploiting band correlation data. The proposed approach achieves superior fused features and improves classification accuracy. Experimental results demonstrate the effectiveness of the lightweight 2D-3D CNN network in extracting refined features for hyperspectral image classification.
[14] This article proposes an active deep learning approach for hyperspectral image (HSI) classification to improve performance while reducing labeling costs. The approach integrates active learning and deep learning in a unified framework. Initially, a convolutional neural network (CNN) is trained with a limited number of labeled pixels. Then, the most informative pixels from a candidate pool are actively selected for labeling. The CNN is fine-tuned iteratively using the newly labeled pixels. Additionally, a Markov random field (MRF) is used to enforce class label smoothness and enhance classification performance. Experimental results on benchmark HSI datasets demonstrate that the proposed approach outperforms other traditional and deep learning-based methods with significantly fewer labeled samples.
[15] This research proposes convolutional neural network-based classifiers to visually identify fruits via cameras for efficient billing in supermarkets. The models achieve superior accuracy compared to previous studies, enabling a quicker and streamlined billing process.
[16] This article reviews the concepts of Random Forest (RF) and Support Vector Machines (SVM) in the context of remote sensing image classification. A meta-analysis of 251 peer-reviewed journal papers is conducted, creating a comprehensive database. The analysis focuses on various aspects including the characteristics of the studies, comparative performance of RF and SVM based on different parameters, and challenges and recommendations for future research. The results provide valuable insights for researchers seeking accurate results in their thematic applications
[17] This paper focuses on the application of convolutional neural network (CNN) algorithms for classifying pneumonia in chest X-ray datasets. Three techniques are evaluated: linear support vector machine classifier with local rotation and orientation-free features, transfer learning using VGG16 and InceptionV3 CNN models, and training a capsule network from scratch. Data augmentation is applied to all three methods. The results show that data augmentation improves performance across all algorithms. Transfer learning is found to be more effective on small datasets compared to a support vector machine with oriented fast and rotated binary (ORB) features and capsule network. Retraining specific features on the target dataset and appropriate network complexity are identified as crucial factors for performance improvement.

II. Background fundamentals

Image Classification

Image classification is a fundamental job in computer vision that entails giving labels or categories to incoming pictures. It includes the creation of algorithms and models capable of automatically recognising and differentiating distinct items or patterns in photographs.
To do image classification, a dataset of labelled pictures is created, with each image assigned to a certain class. These photos are then pre-processed to normalise their size, resolution, and colour values. Feature extraction is critical since it entails extracting significant information from pictures to aid categorization. While traditional approaches require handcrafting specific

characteristics, deep learning techniques, notably convolutional neural networks (CNNs), have demonstrated improved performance by automatically learning important features directly from pictures. After training on the labelled dataset, the model is assessed on a distinct set of pictures known as the test set. Performance indicators such as accuracy, precision, recall, and F1 score are used to evaluate the model's classification efficacy. The trained model may then be used to categorise fresh, previously unseen pictures by feeding them into the model and receiving predictions or probability distributions across the various classes.

Deep Learning

The core idea underlying deep learning is to create and train artificial neural networks with numerous layers to automatically learn and extract usable representations from raw input. Unlike standard machine learning algorithms that rely on created features, deep learning models learn these representations directly from the data.

The neural network, which is constructed of linked layers of artificial neurons, is the central idea in deep learning. Each neuron conducts a basic computation on its input and transmits the result to the next layer. The network's numerous layers enable it to learn increasingly complicated properties and representations as information goes across them. Deep learning learning includes training the network with a huge dataset of known labels or targets. The model modifies its internal parameters or weights during training depending on the differences between its predictions and the real labels. This procedure is usually carried out using gradient descent algorithms, which iteratively update the weights in order to minimise a predetermined loss function.

Deep learning models, once trained, may be utilised for a variety of tasks such as image and audio recognition, natural language processing, anomaly detection, and more. They have achieved state-of-the-art achievements in several sectors by demonstrating extraordinary performance in processing complicated and unstructured data.

Convolutional Neural Networks (CNNs)

CNNs are deep neural networks that are especially built to interpret grid-like input such as photographs. CNNs scan and filter pictures with convolutional layers, collecting spatial hierarchies and patterns. They are well-known for their capacity to automatically learn and extract significant characteristics from photos, making them ideal for image classification applications.

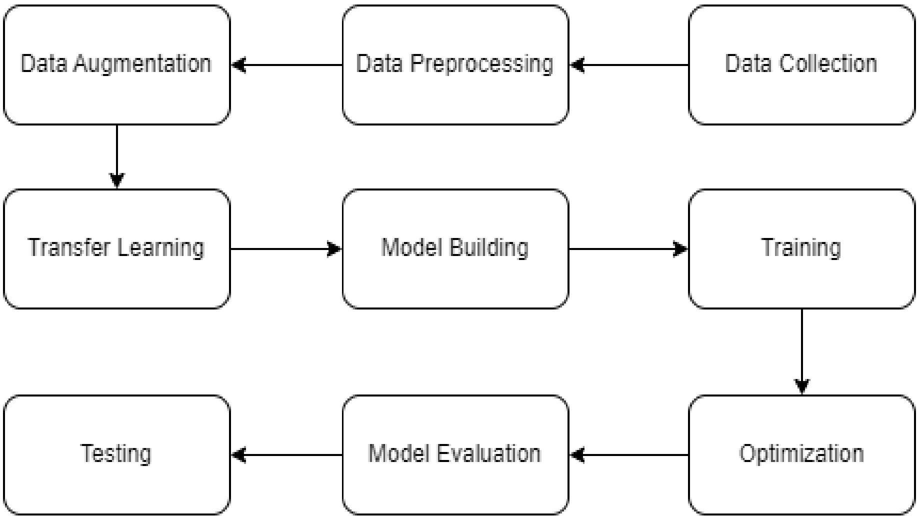
Indian Currency

As Indian money, the Reserve Bank of India (RBI) prints banknotes in various denominations. These banknotes each have their own design, security measures, and denominational variations. The precise categorization of Indian currency notes is crucial for the seamless and reliable operation of financial institutions, notably ATMs. The notes have recently undergone several modifications. Many things have been added or changed, ranging from the design to the colour of the notes. The Rs. 2000 notes were also prohibited; nevertheless, in the model used in the project, the option of the Rs. 2000 note is also provided.

Dataset and Data Augmentation

A comprehensive dataset of high-resolution images of Indian currency notes is necessary for training and evaluating the image classification model. Data augmentation techniques, such as rotation, scaling, and noise addition, are applied to the dataset to increase its diversity and robustness. Data augmentation helps the model generalize well to unseen variations and improve overall performance.

III. Proposed methodology



Dataset

A big collection of high-resolution pictures of Indian currency notes has been compiled using Kaggle and other accessible sources. Currency notes of various denominations, orientations, and conditions are included in the collection. Each data type has been arranged into a folder named with the quantity, such as '10' for ten rupee notes. The data was divided into training and testing sets. The training set has 183 photographs, while the testing set contains 77 photos. There are eight classes in all for the seven distinct sorts of notes, plus one for non-note photos labelled 'background'. '10','20','50','100','200','500', and '2000' are some of the other folders. Since the notes have a new format as a result of the RBI's revisions to the design, including the colour, consideration has been given to such alterations. Even though the 2000 rupee note has been deemed worthless, the folder has been preserved since individuals are still exchanging the currency at banks.

Fig. 1 Few images from the dataset

Data Preprocessing

Tensorflow's ImageDataGenerator() method is used to preprocess the dataset by normalising the image's size, resolution, and colour values. Resizing, cropping, and normalisation procedures are used to ensure uniformity and optimise the data for training. The original data has uneven dimensions. To improve the training process, the dimensions have been changed to (150,150).

Model Selection and Architecture

In order to generate the model, CNN layers were employed. VGG16 was chosen as the foundation model for transfer learning and fine-tuned for the specific goal of Indian currency categorization.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 4, 4, 512)         14714688

 flatten (Flatten)           (None, 8192)              0

 dense (Dense)               (None, 256)               2097408

 dropout (Dropout)           (None, 256)               0

 dense_1 (Dense)             (None, 8)                 2056

=================================================================
Total params: 16,814,152
Trainable params: 2,099,464
Non-trainable params: 14,714,688
_________________________________________________________________
```

Fig. 2 Model Summary

Initialise the pre-trained CNN model of choice with weights learnt from a large-scale dataset, such as VGG16. Add new layers related to the Indian currency categorization assignment to fine-tune the model. This enables the model to apply the learnt features from the pre-trained model to the currency categorization domain.

Training

Divide the preprocessed dataset into training, validation, and test sets. Feed the training data into the model and optimise the model's internal parameters with a suitable optimisation technique, such as. Keep an eye on the validation set's performance to fine-tune hyperparameters and avoid overfitting.

Data Augmentation

Use data augmentation strategies to boost the dataset's diversity and resilience. Random rotations, translations, flips, and other transformations are examples of this. During training, augment the training set on-the-fly to expose the model to a greater variety of variances and increase its generalisation skills.

Model Evaluation

Evaluate the trained model's performance on the test set in terms of accuracy, precision, recall, and other relevant metrics. To evaluate the success of the suggested methodology, compare the findings to baseline models or current methodologies.

Optimization

Based on the evaluation findings, optimise the hyper-parameters. Experiment with various architectural changes, regularisation approaches, and optimisation algorithms to increase the model's performance even more. You can benefit from the Adam optimizer's flexible learning rate and momentum, which can assist optimise the model's performance throughout training.

Deployment and Real-World Testing

Use the trained model to do real-world testing, such as identifying Indian rupee notes in various settings or environments. Examine the model's performance, as well as its practical application and correctness. For model testing, a separate folder named test is utilised. The categorization emerges based on the classes provided by index ex. Index class 4 represents a 2000 rupees note. The accuracy is high for the predictions however still some images get misclassified showing that it's not 100% accuracy or overfitted.

Model Architecture

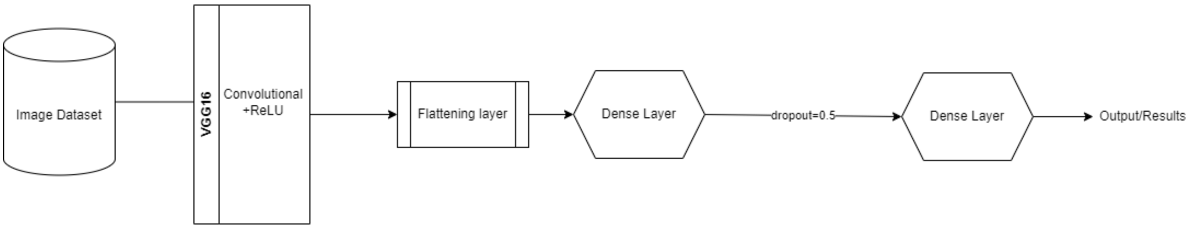


Figure 3 Model Description of Layers

IV. Experimental Analysis

VGG16 is a convolutional neural network architecture developed by the University of Oxford's Visual Geometry Group (VGG). It has demonstrated substantial performance in a variety of computer vision applications, such as picture categorization and object recognition.
The VGG16 model is accessible in TensorFlow via the tf.keras.applications module. It is a class. This class contains a pre-trained VGG16 model that was trained on the large-scale ImageNet dataset. It is made up of numerous convolutional blocks, each of which has several convolutional layers followed by a maximum pooling layer. The number of convolutional layers varies according on the block, with VGG16 having a total of 13 convolutional layers.
Except for the final output layer, VGG16 applies the Rectified Linear Unit (ReLU) activation function after each convolutional and fully connected layer.

Code

Notes_classifier.ipynb

```
import numpy as np
import pandas as pd
import os
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import seaborn as sns
import tensorflow as tf
from tensorflow import keras
from keras.preprocessing.image import ImageDataGenerator
from keras.preprocessing import image
plt.style.use("ggplot")
%matplotlib inline
main_train_dir = os.path.join("train/")
main_test_dir = os.path.join("test/")
```

```
print(main_train_dir)
print(main_test_dir)
two_thousand_dir = os.path.join("train/2000/")
five_hundered_dir = os.path.join("train/500/")
two_hundered_dir = os.path.join("train/200/")
one_hundered_dir = os.path.join("train/100/")
fifty_dir = os.path.join("train/50/")
twenty_dir = os.path.join("train/20/")
ten_dir = os.path.join("train/10/")
bg_dir = os.path.join("train/Background/")
two_thousand_names = os.listdir(two_thousand_dir)
five_hundered_names = os.listdir(five_hundered_dir)
two_hundered_names = os.listdir(two_hundered_dir)
one_hundered_names = os.listdir(one_hundered_dir)
fifty_names = os.listdir(fifty_dir)
twenty_names = os.listdir(twenty_dir)
ten_names = os.listdir(ten_dir)
bg_names = os.listdir(bg_dir)

print(two_thousand_names[:10])
print(five_hundered_names[:10])
print(two_hundered_names[:10])
print(one_hundered_names[:10])
print(fifty_names[:10])
print(twenty_names[:10])
print(ten_names[:10])
print(bg_names[:10])
print(f"total training of 2Thousand Notes : {len(two_thousand_names)}")
print(f"total training of 5Hundered Notes : {len(five_hundered_names)}")
print(f"total training of 2Hundered Notes : {len(two_hundered_names)}")
print(f"total training of 1Hundered Notes: {len(one_hundered_names)}")
print(f"total training of 50Notes : {len(fifty_names)}")
print(f"total training of 20Notes : {len(twenty_names)}")
print(f"total training of 10Notes : {len(ten_names)}")
print(f"total training of Background Pictures : {len(bg_names)}")
# parameters for graph we'll output images in a 4x4
nrows = 4
ncols = 4

# Index for iterating over images
pic_index = 0
# set up matplotlib fig, and size it to fit 4x4 pics
fig = plt.gcf()
fig.set_size_inches(ncols * 2, nrows * 2)

pic_index += 8

two_thousand_pix = [os.path.join(two_thousand_dir, fname)
        for fname in two_thousand_names[pic_index-8:pic_index]]
```

```
five_hundered_pix = [os.path.join(five_hundered_dir, fname)
                for fname in five_hundered_names[pic_index-8:pic_index]]


for i, img in enumerate(two_thousand_pix + five_hundered_pix):
    sub_plot = plt.subplot(nrows, ncols, i + 1)
    sub_plot.axis("Off")
    img_read = mpimg.imread(img)
    plt.imshow(img_read)

plt.show()
import tensorflow as tf
from tensorflow.keras.applications import VGG16

# Load the pre-trained VGG16 model without the top (classification) layers
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(150, 150, 3))

# Freeze the weights of the pre-trained layers so they are not updated during training
for layer in base_model.layers:
    layer.trainable = False

# Create a new model by adding your own classification layers on top of the pre-trained
model
model = tf.keras.models.Sequential([
    base_model,
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(8, activation='softmax')
])

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
#optimization

# Print the model summary
model.summary()

model.summary()
model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])

train_datagen = ImageDataGenerator(rescale=1/255)
validation_datagen = ImageDataGenerator(rescale=1/255)

#Data Augmentation
train_generator = train_datagen.flow_from_directory(main_train_dir,
                                                    batch_size=64,
                                                    target_size=(150,150),
                                                    class_mode="categorical")
```

```
validation_generator = validation_datagen.flow_from_directory(main_test_dir,
                                                        batch_size=16,
                                                         target_size=(150,150),
                                                        class_mode="categorical")
early_stopping = tf.keras.callbacks.EarlyStopping(monitor='val_accuracy', patience=10,
mode='max', verbose=1)

#fit the model with augmented data
history = model.fit(train_generator,
                    epochs=100,
                    steps_per_epoch=len(train_generator),
                    verbose=1,
                    validation_data=validation_generator,
                    validation_steps=len(validation_generator),
                    callbacks=[early_stopping])
acc = history.history["accuracy"]
val_acc = history.history["val_accuracy"]

loss = history.history["loss"]
val_loss = history.history["val_loss"]

epochs = range(len(acc))

plt.plot(epochs, acc, "r", label="Training Accuracy")
plt.plot(epochs, val_acc, "b", label="Validation Accuracy")

plt.legend()
plt.figure()

plt.plot(epochs, loss, "r", label="Training Loss")
plt.plot(epochs, val_loss, "b", label="Validation Loss")

plt.legend()
plt.show()
from tensorflow.keras.preprocessing import image

image_paths = [
    "test/Background/Background_val_12.jpg",
    "test/2000/31.jpg",
    "test/100/1.jpg",
    "test/50/3.jpg",
    "test/10/10__126.jpg",
    "test/200/2.jpg"
]

for path in image_paths:
    img = image.load_img(path, target_size=(150, 150))
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
```

```
    images = np.vstack([x])
    classes = model.predict(images, batch_size=10)
    predicted_class = np.argmax(classes[0])

    if predicted_class == 4:
        print(path + " is Two Thousand Rupees")
    elif predicted_class == 2:
        print(path + " is Five Hundred Rupees")
    elif predicted_class == 3:
        print(path + " is Two Hundred Rupees")
    elif predicted_class == 1:
        print(path + " is One Hundred Rupees")
    elif predicted_class == 5:
        print(path + " is Fifty Rupees")
    elif predicted_class == 6:
        print(path + " is Twenty Rupees")
    elif predicted_class == 7:
        print(path + " is Background")
    else:
        print(path + " is Ten Rupees")

    img = mpimg.imread(path)
    plt.imshow(img)
    plt.show()
// End of Code
//-------------------------------------------------------------------------
```

The model is often supplied with pre-trained weights from the ImageNet dataset. These weights encapsulate the learned representations of many visual qualities and help the model perform well on a number of image classification tasks. The dropout layer is used to reduce the input by a given percentage. For example, in the model, we may reduce 50% of the output to be used as input. This layer addresses the overfitting issue, allowing the model to be flexible. A dense layer is a fully linked layer that connects one layer to the next. The dense layer additionally mentions the output value. It is 8 in this case since we are categorising 8 classes.

As an optimization method we have used the Adam Optimizer and since the model is for multiclass classification we have used categorical_crossentropy as the loss metric.

```
...
3/3 [==============================] - 69s 25s/step - loss: 0.0431 - accuracy: 0.9945 - val_loss: 0.2929 - val_accuracy: 0.9091
Epoch 26/100
3/3 [==============================] - 74s 27s/step - loss: 0.0336 - accuracy: 1.0000 - val_loss: 0.3007 - val_accuracy: 0.9091
Epoch 26: early stopping
```

Fig. 4 The accuracy and value accuracy after successful model building or training

Based on the val_accuracy, an early stopping function has been developed to halt the epochs when the val_accuracy is no longer advancing. It is started with a patience of 10, which means it waits for ten consecutive values and stops the epochs if it observes a stagnant value.

V. Comparative Analysis

As a result of testing and training with the combo VGG16 as base layer and more classification layers we have obtained a model able to predict a good number of the photos correctly.

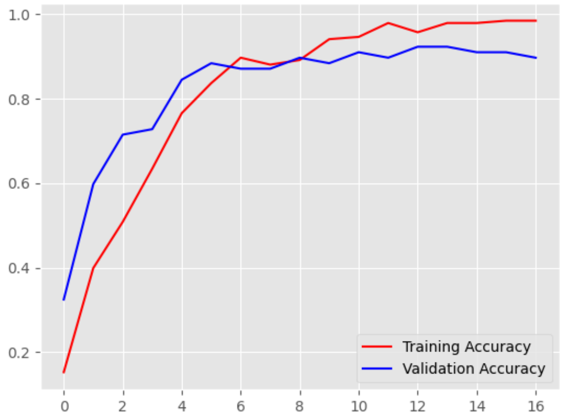


Fig. 5 Accuracy of model training

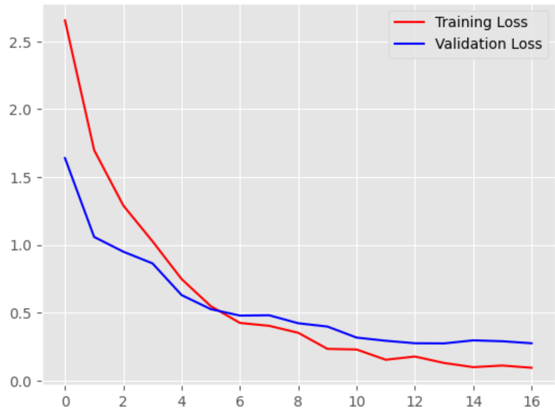
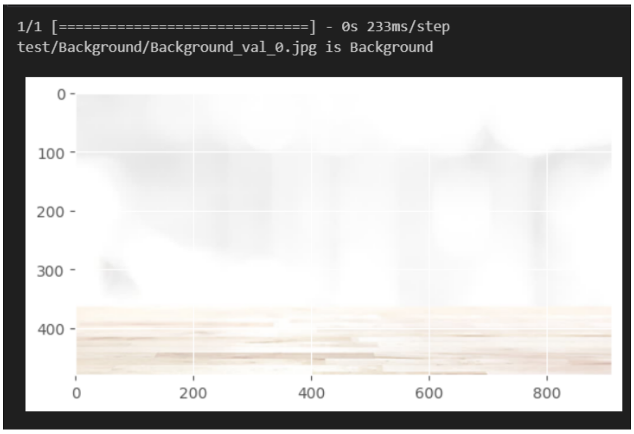


Fig. 6 Loss Overview

The epochs demonstrate how the values change with each iteration. Charts or graphs are used to depict the change. The value of accuracy increases (non-uniformly) with each iteration, but the process is terminated after the 16th epoch due to the virtually stationary value of value accuracy, i.e. the accuracy of the forecasts or values.

OUTPUT

The output of the model gives the classifications of the images.



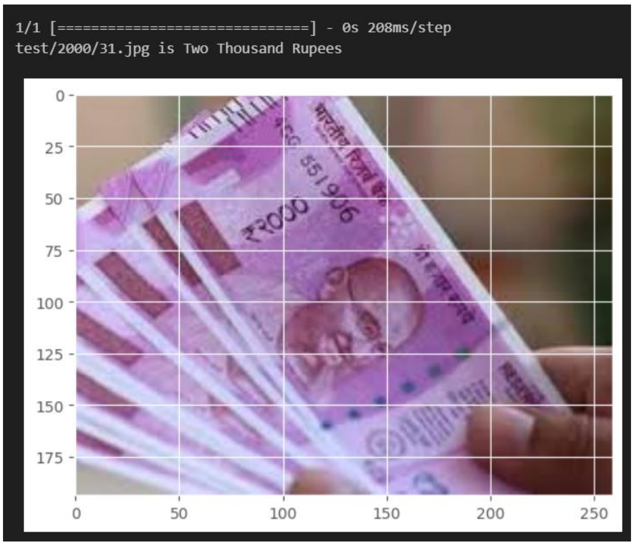


Fig. 7 Correctly classified notes

The predictions are made for an array of inputs in order to get a brief overview of how good the model is. It is observed that even though it is able to identify most of the pictures, some images are wrongly classified. The reason can be the simple CNN mode. It can also be blamed on the quality of data used for the training. Blurry or ambiguous data can lead to confusion for the model which then produces a wrong output.

VI. Conclusion

The project employed deep learning techniques to classify images of Indian cash. The objective was to create a model that could correctly detect different denominations of Indian banknotes. The research began with the collection and preparation of photos of Indian cash. Data augmentation techniques were used to improve the model's capacity to generalise by increasing the diversity and quantity of the dataset. The TensorFlow framework was used to build a deep learning model. Several convolutional layers were used for feature extraction, followed by pooling layers to minimise spatial dimensionality. To reduce overfitting, dropout layers were used, and fully linked layers were added for classification. The Adam optimizer was used to optimise the model's performance during the training phase. The model was trained on the updated dataset, and the progress of the training was tracked using relevant assessment measures such as accuracy and loss. The suggested approach performed well in categorising Indian money. The model's accuracy achieved a reasonable level, proving its ability to correctly detect different denominations. The benchmarking methodologies used to compare the model's performance to baseline models and current approaches enabled an accurate evaluation of its performance.

The study demonstrates the efficacy of deep learning and transfer learning approaches in picture categorization problems. Using pre-trained models like VGG16 allows the model to use learnt representations from large-scale datasets, enhancing accuracy and efficiency.

Overall, this project provides a solid method for automated Indian currency categorization. The model created may be used in a variety of settings, including automated teller machines (ATMs), vending machines, and money sorting systems, resulting in enhanced efficiency and accuracy in financial transactions. Expanding the dataset, experimenting with alternative architectures, and fine-tuning hyperparameters can all help to improve the model's performance. Because the dataset under consideration is limited, it may be anticipated that the accuracy of predictions will improve as the number of photos increases. Data that is unclear or noisy might also be an issue. In such a circumstance, incorrect class prediction is possible. A better model that can process the image better and deliver more accurate results can be constructed. Future work will be done in that context.

VII. References

[1] Ngugi, Lawrence C., Moataz Abelwahab, and Mohammed Abo-Zahhad. "Recent advances in image processing techniques for automated leaf pest and disease recognition–A review." Information processing in agriculture 8.1 (2021): 27-51.

[2] Kim, Hee E., et al. "Transfer learning for medical image classification: A literature review." BMC medical imaging 22.1 (2022): 69.

[3] Li, Yinglong. "Research and application of deep learning in image recognition." 2022 IEEE 2nd International Conference on Power, Electronics and Computer Applications (ICPECA). IEEE, 2022.

[4] Basha, SH Shabbeer, et al. "Impact of fully connected layers on performance of convolutional neural networks for image classification." Neurocomputing 378 (2020): 112-119.

[5] Wang, Pin, En Fan, and Peng Wang. "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning." Pattern Recognition Letters 141 (2021): 61-67.

[6] Yuniarti, Anny, Nanik Suciati, and Agus Zainal Arifin. "Image classification performance evaluation for 3d model reconstruction." 2020 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET). IEEE, 2020.

[7] Malhotra, Priyanka, et al. "Deep neural networks for medical image segmentation." Journal of Healthcare Engineering 2022 (2022).

[8] Bera, Somenath, and Vimal K. Shrivastava. "Analysis of various optimizers on deep convolutional neural network model in the application of hyperspectral remote sensing image classification." International Journal of Remote Sensing 41.7 (2020): 2664-2683.

[9] Jadhav, Megha, Yogesh kumar Sharma, and G. M. Bhandari. "Currency identification and forged banknote detection using deep learning." 2019 International Conference on Innovative Trends and Advances in Engineering and Technology (ICITAET). IEEE, 2019.

[10] Wang, Wei, et al. "Development of convolutional neural network and its application in image classification: a survey." Optical Engineering 58.4 (2019): 040901-040901.

[11] Reshi, Aijaz Ahmad, et al. "An efficient CNN model for COVID-19 disease detection based on X-ray image classification." Complexity 2021 (2021): 1-12.

[12] Lei, Xinyu, Hongguang Pan, and Xiangdong Huang. "A dilated CNN model for image classification." IEEE Access 7 (2019): 124087-124095.

[13] Yu, Chunyan, et al. "A simplified 2D-3D CNN architecture for hyperspectral image classification based on spatial–spectral fusion." IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 13 (2020): 2485-2501.

[14] Cao, Xiangyong, et al. "Hyperspectral image classification with convolutional neural network and active learning." IEEE Transactions on Geoscience and Remote Sensing 58.7 (2020): 4604-4616.

[15] Tripathi, Milan. "Analysis of convolutional neural network based image classification techniques." Journal of Innovative Image Processing (JIIP) 3.02 (2021): 100-117.

[16] Sheykhmousa, Mohammadreza, et al. "Support vector machine versus random forest for remote sensing image classification: A meta-analysis and systematic review." IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 13 (2020): 6308-6325.

[17] Yadav, Samir S., and Shivajirao M. Jadhav. "Deep convolutional neural network based medical image classification for disease diagnosis." Journal of Big data 6.1 (2019): 1-18.