**Tribhuvan University**

**Institute of Science and Technology**

**Seminar Report**

**On**

**Nepali currency recognition using CNN**

**Submitted to**

**Central Department of Computer Science & Information Technology**

**Tribhuvan University, Kirtipur**

**Kathmandu, Nepal**

**Submitted by**

**Akkal Bahadur Bist**

**Roll No.: 19 / 079**

**In partial fulfillment of the requirement for Master's Degree in Computer**

**Science and Information Technology (M.Sc. CSIT), Second Semester**

**2024**

**Tribhuvan University**

**Institute of Science and Technology**

**SUPERVISOR’S RECOMMENDATION**

I hereby recommend that this Seminar report is prepared under my supervision by **Mr. Akkal Bahadur Bist** entitled “**Nepali currency recognition using CNN**” be accepted as fulfillment in partial requirement for the degree of Master's of Science in Computer Science and Information Technology.

…………………………………

Supervisor

Asst. Prof. Ram Krishna Dahal

Central Department of Computer Science and Information Technology

**Tribhuvan University**

**Institute of Science and Technology**

**LETTER OF APPROVAL**

This is to certify that the seminar report prepared by **Mr. Akkal Bahadur Bist** entitled ‘**Nepali currency recognition using CNN**' in partial fulfillment of the requirements for the degree of Master's of Science in Computer Science and Information Technology has been well studied. In our opinion, it is satisfactory in the scope and quality as a project for the required degree.

Evaluation Committee

|  |  |  |
| --- | --- | --- |
| ………………………….. |  | ………………………… |
| Asst. Prof. Sarbin Sayami |  | Asst. Prof. Ram Krishna Dahal |
| (HOD |  | (Supervisor) |

Central Department of Computer Central Department of Computer

Science & Information Technology Science & Information Technology

..………………………

(Internal)

# **ACKNOWLEDGEMENT**

With immense please, I submit my deepest gratitude to the Central Department of Computer Science and Information Technology, Tribhuvan University.

First of all, I would like to thank **Asst. Prof.** **Sarbin Sayami**, Head of Central Department of Computer Science and Information Technology, Tribhuvan University, **Asst. Prof. Bikash Balami** and **Asst. Prof. Arjun Singh Saud** for their guidance, feedback and encouragement.

I am very glad to express my deepest sense of gratitude and sincere thanks to my highly respected and esteemed supervisor **Asst. Prof.** **Ram Krishna Dahal** for her valuable guidance in carrying out this work under her effective supervision and enlightenment. I am also grateful for her excellent guidance, patience, trust and correction of my seminar work.

I would also like to thank my friends and Amrit Campus, Thamel student who helped me during the study and research of the topics related to the seminar.

Akkal Bahadur Bist

Roll No: 19/079

# **ABSTRACT**

Machine Learning (ML) has been an active field of research for a long time, in recent years it has gained more traction because of the advent of new techniques and increase in compute power. Deep learning methods, a subset of machine learning methods, have been shown to be highly accurate in image recognition and classification tasks, which is evidently in line with our desires.

In this seminar, I experimented Convolution Neural Network (CNN) architecture Model with multiple deep learning architectures and techniques, and landed at a simple Convolutional Neural Network (CNN) architecture. This architecture extracts essential features from images for classification, thus increasing efficiency and accuracy. Finally, Nepali currency recognition was done from basic Convolution Neural Network (CNN) model and model with different changes are done at a training time and compare with another model like ResNet50 and VGG19 model. All in all, my main contributions is: (1) through using CNN, the average accuracy of Nepali currency recognition obtained was 97% on test set in basic CNN model, 95% and 96% on test set in same model with different parameters, 91% in ResNet50 and 90% in VGG19 Model. (2) easy to choose architecture from CNN model for Nepali currency recognition and classification.

**Keywords:**

*Image Classification, Machine Learning (ML), Convolutional Neural Network (CNN), Nepali Currencies, ResNet50, VGG19, Convolution Layer (CONV), Pooling Layer (POOL), Fully Connected layer (FC), Stride, Padding, Filter, Kernel, Confusion Matrix, Learning Rate, Epoch, TensorFlow, Data Collection.*

**TABLE OF CONTENT**

[ACKNOWLEDGEMENT III](#_Toc165370299)

[ABSTRACT IV](#_Toc165370300)

[LIST OF FIGURES VI](#_Toc165370301)

[LIST OF TABLES VII](#_Toc165370302)

[LIST OF ABBREVIATIONS VIII](#_Toc165370303)

[CHAPTER 1: INTRODUCTION 1](#_Toc165370304)

[1.1 Background 1](#_Toc165370305)

[1.2 Problem Statement 2](#_Toc165370306)

[1.3 Objective 2](#_Toc165370307)

[CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW 3](#_Toc165370308)

[2.1 Background Study 3](#_Toc165370309)

[2.1.2 Convolutional Neural Network 3](#_Toc165370310)

[2.2 Previous Works, Discussions and Findings 7](#_Toc165370311)

[CHAPTER 3: METHODOLOGY 9](#_Toc165370312)

[3.2 Tools and IDE 9](#_Toc165370313)

[3.3 Data Analysis 9](#_Toc165370314)

[3.4 Data Preparation 11](#_Toc165370315)

[3.5 CNN Architecture Design: 12](#_Toc165370316)

[CONCLUSION 17](#_Toc165370317)

[REFERENCES 18](#_Toc165370318)

# **LIST OF FIGURES**

[Figure 1: Typical Structure of CNN 3](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370321)

[Figure 2: Convolution operation (2-D), kernel size = 2, strides = 1, padding = 0 4](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370322)

[Figure 3: Max Pooling 6](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370323)

[Figure 4: Nepali Currency Dataset with labels 10](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370324)

[Figure 5: Basic Model of CNN Architecture 12](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370325)

[Figure 6: CNN Model Defining Code 13](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370326)

[Figure 7: Basic CNN Model Training and Validation Accuracy and Loss 14](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370327)

[Figure 8: Recognize currency in time and accuracy percentage 15](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370328)

[Figure 9: Basic CNN model Confusion matrix with different padding, stride and filters used 15](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370329)

# **LIST OF TABLES**

[Table 1: Kernel Matrix, Convolution Input and Output matrix 4](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370330)

[Table 2: ReLU function used matrix. 5](file:///C:\Users\AKKAL\Desktop\Seminar\SeminarReport.docx#_Toc165370331)

[Table 3: Convolution on 2D Image / Single Channel 7](#_Toc165370332)

[Table 4: CNN Architecture Layer Model 11](#_Toc165370333)

[Table 5: Model Evaluation Result 16](#_Toc165370334)

# **LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| AI | Artificial Intelligence |
| CNN | Convolutional Neural Network |
| DL | Deep Learning |
| GPU | Graphics Processing Units |
| ML | Machine Learning |
| POOL | Pooling Layer |
| RAM | Random Access Memory |
| RELU | Rectified Linear Unit |
| ResNet | Residual Network |
| SGD | Stochastic Gradient Descent |
| VGG | VGG Visual Geometry Group |
| NFDN | National Federation of the Disabled |

# **CHAPTER 1: INTRODUCTION**

## **Background**

The National Federation of the Disabled - Nepal (NFDN) reported [1] that, according to the 2021 census of Nepal, 2.2% of the Nepali population have some form of disability. Amongst them, 16.88% have low vision, 5.37% are blind and 1.56% are deaf and blind – in raw numbers, this total to about 172622 people. Needless to say, this is a significant number of people.

Those with vision impairments face numerous challenges that many of us overlook. These include difficulties with mobility, accessing information, communication, entertainment, safety, and social acceptance. It's crucial for us to use our skills to address these issues however we can.

The use of modern technology seems like a rational approach to solving problems for the impaired. As Computer Science students, this is even more natural to us; thus, I decided to apply my skill set in order to solve a particular niche that, primarily, visually impaired people face.

The above-mentioned niche is the problem of currency recognition. Object recognition is an important and highly demanded area of features extraction. An object can be anything in real life. It can be text in a document, a license plate of a vehicle, an iris in a person’s eyes, a sign in a sign language, a face of a person, and so on. Similarly, paper currency recognition is as important as any other object recognition.

Convolutional Neural Networks (CNN) emerged as a powerful tool for image classification. CNN models like VGG, LeNet-5, AlexNet, ImageNet, ResNet, GoogleNet etc. CNNs have also found successful applications in remote sensing image analysis. Various CNN-based scene classification methods have been developed, including using pre-trained CNNs as feature extractors, fine-tuning them on datasets, or globally initializing weights for training.

In this seminar involved training various CNN models, including basic CNN, ResNet50, and VGG19, using a dataset of Nepali currency. Through experimentation with different parameters, the aim was to achieve the highest accuracy in recognizing Nepali currency notes. Given the significance of this technology for visually impaired individuals, the project focused on leveraging deep learning techniques to accurately classify currency notes. The ultimate goal was to develop a smartphone app that allows users to scan currency notes and receive audio feedback indicating their value, thereby addressing the challenges faced by the visually impaired in Nepal and beyond.

This report used a special type of Convolutional Neural Network known as basic CNN model for Nepali currency recognition using ‘*Nepali Currency*’ datasets. There use different Currency Value are: 5, 10, 20, 50, 100, 500 & 1000 paper based notes around 15400 notes dataset with help of Amrit Campus students.

## **Problem Statement**

The accurate and efficient recognition of Nepali currency is a significant challenge, particularly in the context of automated systems such as vending machines, ATMs, and currency exchange kiosks. Existing methods for currency recognition often rely on traditional image processing techniques, which may not be robust enough to handle variations in lighting, orientation, and background clutter. Therefore, there is a growing need to explore advanced techniques, such as Convolutional Neural Networks (CNNs), to develop a reliable currency recognition system for Nepali currency.

## **Objective**

* To train a basic CNN model for recognizing Nepali currency, experiment with various parameters such as kernel size, stride, padding, filters, activation function, and batch size to achieving higher accuracy. Additionally, train this data into pretrained CNN models like ResNet50 and VGG19.
* To compare the recognition accuracy, efficiency, and robustness of basic CNN model architectures with pretrained models like ResNet50 and VGG19 for Nepali currency recognition.

# **CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW**

## **2.1 Background Study**

### **2.1.2 Convolutional Neural Network**

A CNN consists of an input layer, output layer, as well as multiple hidden layers. The hidden layer of a CNN typically consists of a series of convolutional layers that convolves with a multiplication or other dot product. The activation function is commonly a ReLU layer and is subsequently followed by additional convolutions such as pooling layers, fully connected layers and normalization layers, referred to as hidden layers because their inputs and outputs are masked by the activation function and final convolution.

The final convolution, in turn, often involved back-propagation in order to more accurately weight the end product. Though the layers are colloquially referred to as convolutions, this is only by convention. Mathematically, it is a sliding dot product or cross-correlation. This has significance for the indices in the matrix, in that it affects how weight is determined at a specific index point. Example of a typical CNN is depicted in Figure 1.

In the first layer, CNN matched parts rather than the whole image, therefore breaking the image classification process down into smaller parts (features). A 3x3 grid was defined to represent the features extraction by the CNN for evaluation.

Figure 1: Typical Structure of CNN (<https://towardsdatascience.com/>)

The following process, known as filtering, involved lining the feature with the image patch. One-by-one, each pixel was multiplied by the corresponding feature pixel, and once completed, all the values were summed and divided by the total number of pixels in the feature space. The final value for the feature was then placed into the feature patch. This process was repeated for the remaining feature patches followed by trying every possible match repeated application of this filter, which is known as a convolution.

Table 1: Kernel Matrix, Convolution Input and Output matrix

|  |  |
| --- | --- |
| 1 | 1 |
| 1 | 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| 5 | 4 | 7 | 6 |
| 3 | 7 | 9 | 2 |
| 6 | 2 | 1 | 3 |

|  |  |  |
| --- | --- | --- |
| 19 | 27 | 24 |
| 18 | 19 | 15 |

1. A 2 x 2 kernel (b) The convolution input

(c) The convolution output

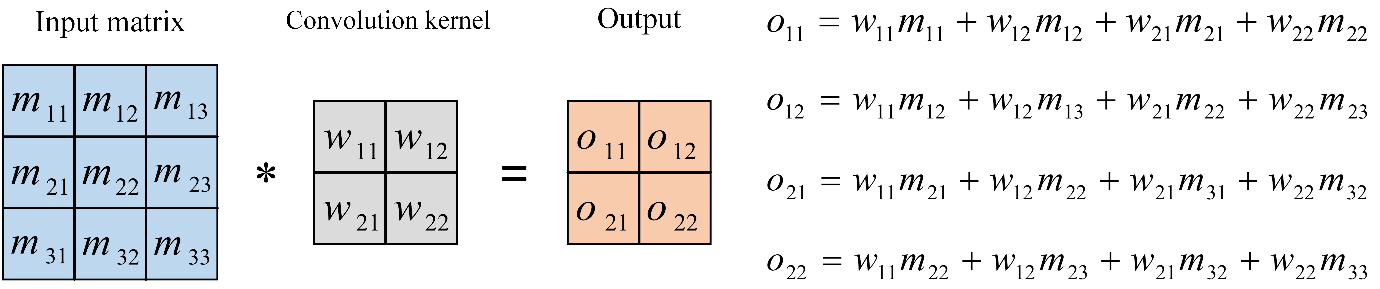


Figure 2: Convolution operation (2-D), kernel size = 2, strides = 1, padding = 0

When the convolution kernel was overlapped on top of the input image, the computation of the product between the numbers at the same location in the kernel and the input could be done and a single number could be obtained by summing these products together. For example, when the kernel was overlapped with the top left region in the input, the convolution result at that spatial location was 1×5+1×3+1×4+1×7 = 19. The kernel was then moved down by one pixel and the next convolution result was obtained as: 1×3+1×7+1×6+1×2 = 18. The kernel down was then moved down till it reached the bottom border of the input matrix (image). Then, the kernel was returned to the top, and moved to its right by one element (pixel). The convolution for every possible pixel location was repeated until the kernel was moved to the bottom right corner of the input image.

Transfer Function

Table 2: ReLU function used matrix.

|  |  |  |  |
| --- | --- | --- | --- |
| 15 | 20 | -10 | 35 |
| 18 | -110 | 25 | 100 |
| 20 | -15 | 25 | -10 |
| 101 | 75 | 18 | 23 |

|  |  |  |  |
| --- | --- | --- | --- |
| 15 | 20 | 0 | 35 |
| 18 | 0 | 25 | 100 |
| 20 | 0 | 25 | 0 |
| 101 | 75 | 18 | 23 |

ReLU Layer

(0, 0)

The normalization layer of a CNN, also referred to as the process of Rectified Linear Unit (ReLU), involved changing all negative values within the filtered image to 0. The purpose of ReLU was to increase the nonlinearity of the CNN. The output was f(x) = max(0,x).The next layer of a CNN was referred to as “max pooling”, which involved shrinking the image stack. In order to pool an image, the window size had to be defined (e.g. usually 2x2/3x3 pixels) and the stride had to be defined. The window was then filtered across the image in strides, with the max value being recorded for each window. Max pooling was considered as a sample-based discretization process. The objective was to down-sample an input representation (image, hidden-layer output matrix, etc.), reducing its dimensionality and allowing for assumptions to be made about features contained in the sub-regions binned. Max pooling reduced the dimensionality of each feature map whilst retaining the most important information.

**Pooling Layer:**

Pooling layers, also referred to as down sampling, serve to reduce the dimensionality of the input, thereby decreasing the number of parameters. Similar to convolutional layers, pooling operations involve traversing a filter across the input. However, unlike convolutional layers, the pooling filter does not possess weights. Instead, the filter applies an aggregation function to the values within its receptive field, generating the output array. Two primary types of pooling are commonly employed:

1. **Max Pooling:** It selects the pixel with the maximum value to send to the output array.
2. **Average pooling:** It calculates the average value within the receptive field to send to the output array.
3. **Min pooling:** a pooling operation in convolutional neural networks (CNNs), selects the minimum value within a receptive field to include in the output array, aiding in spatial dimension reduction and feature extraction.

|  |  |
| --- | --- |
| 7 | 7 |
| 6 | 8 |

Max Pooling

2X2 Filter & Stride of 2

|  |  |  |  |
| --- | --- | --- | --- |
| **7** | 2 | 5 | 2 |
| 4 | 5 | 4 | **7** |
| 3 | 3 | 4 | 2 |
| **6** | 4 | **8** | 6 |

Figure 3: Max Pooling

The behavior of the convolutional layer is primarily governed by the following main hyperparameters:

**Kernel size:** It determines the size of the sliding window. It is generally recommended to use smaller window sizes, preferably odd values such as 1, 3, 5, and occasionally, rarely 7.

**Stride:** The stride parameter determines the number of pixels the kernel window will move during each step of convolution. Typically, it is set to 1 to ensure that no locations are missed in an image. However, it can be increased if the intention is to simultaneously reduce the input size.

**Padding:** Padding refers to the technique of adding zeros to the border of an image. By applying padding, the kernel can fully filter every position of an input image, ensuring that even the edges are properly processed.

**Number of filters /Depth:** The number of filters in a convolutional layer determines the number of patterns or features that the layer will seek to identify. In other words, it governs the number of distinct characteristics or elements that the convolutional layer will focus on detecting.

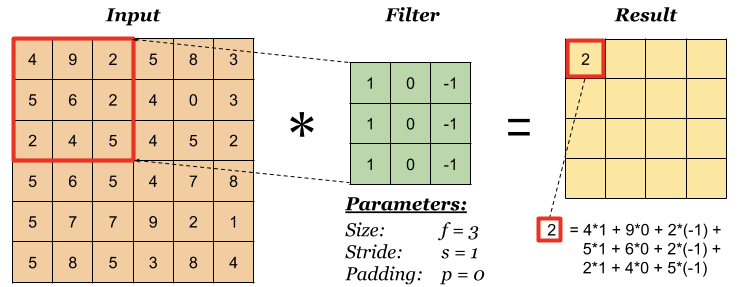
The output size of the convoluted layer is determined by several factors, including the input size, kernel size, stride, and padding. The formula to calculate the output size is as follows:

Let’s take an example, imagine we have an input image with dimensions of 6x6 pixels. For the convolutional operation, we use a kernel with dimensions of 3x3 pixels, a stride of 1, and no padding (padding of 0).

To calculate the output size of the convoluted image, we can apply the following formula: output size = 1 + (input size — kernel size + (2 \* padding)) / stride.

Plugging in the values, we get: output size = 1 + (6–3 + (2 \* 0)) / 1 = 1 + (3 / 1) = 1 + 3 = 4.

Table 3: Convolution on 2D Image / Single Channel (https://medium.com/)



Hence, the resulting convoluted image will have dimensions of 4x4 pixels.

## **2.2 Previous Works, Discussions and Findings**

First successful implementation of CNN was done by Yan LeCunn in 1990’s which is used to read zip codes and digits [2]. CNN has been proved to be a powerful tool for image classification and object detections. Noura A. Semary et. al., in their paper [3], show a relatively simple approach to currency recognition for Egyptian currencies. They use classical image processing techniques to perform image segmentation, enhancement, region of interest extraction and template matching. They got an impressive accuracy of 89%, especially considering that they didn’t leverage modern machine learning techniques.

Buvana. M et. al. [4] combined both classical image processing techniques and deep learning to perform currency recognition for Indian currencies. They used classical image processing techniques to perform preprocessing to training data before feeding them to a Convolutional Neural Network (CNN) to extract out essential features of the data. Extracting essential features to perform classification is both effective and efficient. So, we would say that this approach is a step forward from a full image processing-based system.

Qian Zhang et. al. [5] also used CNN to get average accuracy up to 96.6% on New Zealand currency. In this comprehensive master’s thesis, the author concludes: “we found from the experimental results that, in general, CNN is much suitable for our currency identification requirements.” The author tried out multiple CNN architectures like ResNet50 and VGG19 Model. Ayush Antre et.al. [6] also uses CNN to get above 90% accuracy on Indian currency.

All in all, the evidence points to the conclusion that training a CNN based model is probably the best choice for highly accurate currency recognition.

# **CHAPTER 3: METHODOLOGY**

The two basic neural networks - CNN and RNN i.e., CNN played vital role for the implementation of the Image Activity Classification and Recognition using Image. It is compare the multi-class image recognition and classification performance of the popular basic Neural Network architecture, The Convolution Neural Network (CNN) with changes different parameters like (Kernel Size, Stride, Padding Filters) and train the data set in other two popular transfer learning architectures – ResNet50 and VGG19. Both ResNet and VGG19 models that they used are pre-trained on Image my dataset. For the training model to take ‘Nepali Currency’ image dataset that collected from various mobile cameras photos with the help of lots of peoples for currency recognition. The performance of all three models will be compared using confusing matrices and their average accuracy implementation consisted of following steps:

**3.1. Data Collection**

This seminar study, using ‘Nepali Currency’ dataset this are collected through Amrit Campus eight semester students, my friend and me using mobile cameras images. There is total 1550 images have 14000 images belonging to each 7 classes in training data sets, 700 images belonging to 7 classes in testing data sets and 840 images belonging to 7 classes in validation sets.

## **3.2 Tools and IDE**

* Python with, NumPy, TensorFlow, Keres, Matplotlib and cv2 in Python Jupiter and Kaggle platform.

## **3.3 System Specifications**

* This seminar leveraged the robust capabilities of the Kaggle platform for model training. Kaggle stands out as a leading hub for data science and machine learning competitions, offering a wealth of resources including diverse datasets, insightful tutorials, and a seamless cloud-based collaboration environment. Kaggle provide GPU T4 x2, 32GB RAM, 20-hour continuous run time in a week, and 73GB cloud storage.
* Core i5, 8GB RAM in laptop

## **3.4 Data Analysis**

**Introduction to Dataset:**

The dataset is used in the field of computer vision and machine learning but it is our collected data sets and used privately. It consists of collection of classes images, each class belonging to different Nepali Currency notes like (Rs. 5 in one class, Rs, 10 in two class, Rs. 20 in three class, Rs. 50 in four class, Rs. 100 in five class, Rs. 500 in six class and Rs. 1000 rupees in class seven) this dataset is split into three main subsets: Training Set and Test Set and Validation Set.

**Dataset Details:**

In this Nepali Currency dataset contains total 15400 images, divided into 14000 images for training, 7000 images for testing and 840 images for validation. The 7 classes are:

Class-1 – 5 rupees

Class-2 – 10 rupees

Class-3 – 20 rupees

Class-4 – 50 rupees

Class-5 – 100 rupees

Class-6 – 500 rupees

Class-7 – 1000 rupees

Figure 4: Nepali Currency Dataset with labels

## **3.5 Data Preparation**

In this step, the data had to prepare in the manner so that it was convenient to be given as given input learning model. Data are augmented using ImageDataGenerator and To splitting data set into some training set, validation set and test set for common machine learning model and provide intended to print the dimensions of the Nepali Currency dataset subsets using.

Table 4: CNN Architecture Layer Model

|  |  |  |
| --- | --- | --- |
| Training Set | Validation Set | Testing Set |

**Training Set:** Training set is used to adjust various weight and parameters of the model.

**Validation Set:** Validation Set are used to adjust the parameter of the model instead they are used to reduce over fitting problem.

**Testing Set:** Testing set are used for evaluating the predictive power of the model.

# Creating training, testing and validation sets code is:

*training\_set = train\_datagen.flow\_from\_directory('/kaggle/input/currency-final-one/train',*

*target\_size = img\_size,*

*batch\_size = batch\_size,*

*shuffle = True,*

*class\_mode = 'categorical')*

*test\_set = train\_datagen.flow\_from\_directory('/kaggle/input/currency-final-one/test',*

*target\_size = img\_size,*

*batch\_size = 700,*

*shuffle = True,*

*class\_mode = 'categorical')*

*valid\_set = train\_datagen.flow\_from\_directory('/kaggle/input/currency-final-one/valid',*

*target\_size = img\_size,*

*batch\_size = batch\_size,*

*shuffle = True,*

*class\_mode = 'categorical')*

Also perform target label of the dataset for divide data into categorical data like class, labels, binary vectors etc. and apply image transformation method to augmented data using:

*train\_datagen = ImageDataGenerator(rescale = 1./255,*

*horizontal\_flip=True,*

*rotation\_range = 20,*

*width\_shift\_range = 0.2,*

*height\_shift\_range = 0.2,*

*featurewise\_center=True,*

*featurewise\_std\_normalization=True,*

*vertical\_flip=True,*

*zoom\_range = 0.2,*

*)*

## **3.6 CNN Architecture Design:**

**CNN in Image recognition classification**

Basic CNN Model is design for sole purpose of Image Classification and recognition. The architecture of network will be modified; its filter each changed to see which will give the best output.

Input Layer

Stack of CONV, POOL, and FC Layer

Output Class Score

Output Layer

Figure 5: Basic Model of CNN Architecture

**Example:** [INPUT—CONV—RELU—POOL—FC]

****As already mentioned in this CNN pretrained model Resnet and VGG19 has fixed kernel size, padding, stride and filters in both model with initial basic CNN model architecture and train it.

Figure 6: CNN Model Defining Code

The accuracy of this architecture using our dataset in 280 iterations in each epoch and after training 16 epoch is very good accuracy 97% accuracy in Basic CNN model and epoch 18 has terminate in my defining 25 training epochs, because I have defined EarlyStopping function in patience=3.

Epoch 10/25

280/280 [==============================] - 971s 3s/step - loss: 0.1839 - accuracy: 0.9418 - val\_loss: 0.2150 - val\_accuracy: 0.9298

Epoch 11/25

280/280 [==============================] - 983s 4s/step - loss: 0.1352 - accuracy: 0.9561 - val\_loss: 0.1817 - val\_accuracy: 0.9310

Epoch 12/25

280/280 [==============================] - 990s 4s/step - loss: 0.1375 - accuracy: 0.9550 - val\_loss: 0.1731 - val\_accuracy: 0.9429

Epoch 13/25

280/280 [==============================] - 974s 3s/step - loss: 0.1144 - accuracy: 0.9608 - val\_loss: 0.2151 - val\_accuracy: 0.9345

Epoch 14/25

280/280 [==============================] - 972s 3s/step - loss: 0.1150 - accuracy: 0.9616 - val\_loss: 0.1913 - val\_accuracy: 0.9369

Epoch 15/25

280/280 [==============================] - 978s 3s/step - loss: 0.0939 - accuracy: 0.9689 - val\_loss: 0.1113 - val\_accuracy: 0.9607

Epoch 16/25

280/280 [==============================] - 977s 3s/step - loss: 0.0882 - accuracy: 0.9720 - val\_loss: 0.2404 - val\_accuracy: 0.9345

Epoch 17/25

280/280 [==============================] - 967s 3s/step - loss: 0.1162 - accuracy: 0.9634 - val\_loss: 0.1251 - val\_accuracy: 0.9643

Epoch 18/25

280/280 [==============================] - 970s 3s/step - loss: 0.0993 - accuracy: 0.9689 - val\_loss: 0.1724 - val\_accuracy: 0.9417

In this model, there change parameter (kernel size, stride, padding, and filter) values and train this model again but they not achieve this previous accuracy.



It has terminated 14 epoch and result has 95% accuracy.

Epoch 13/25

280/280 [==============================] - 1060s 4s/step - loss: 0.1500 - accuracy: 0.9541 - val\_loss: 0.2839 - val\_accuracy: 0.9036

Epoch 14/25

280/280 [==============================] - 1053s 4s/step - loss: 0.1515 - accuracy: 0.9512 - val\_loss: 0.2528 - val\_accuracy: 0.9226

Also, in (kernel size, stride, padding, and filter) are little changes they achieve 96% accuracy in basic model.

Epoch 10/25

280/280 [==============================] - 1137s 4s/step - loss: 0.1466 - accuracy: 0.9513 - val\_loss: 0.1596 - val\_accuracy: 0.9571

Epoch 11/25

280/280 [==============================] - 1148s 4s/step - loss: 0.1207 - accuracy: 0.9592 - val\_loss: 0.2382 - val\_accuracy: 0.9369

Epoch 12/25

280/280 [==============================] - 1126s 4s/step - loss: 0.1185 - accuracy: 0.9605 - val\_loss: 0.2664 - val\_accuracy: 0.9238

This model has training and validation accuracy and loss are respectively as:

Figure 7: Basic CNN Model Training and Validation Accuracy and Loss

It will make the predictions through the trained basic CNN model using the test image dataset is:

1/1 [==========================] - 0s 20ms/step

1/1 [==========================] - 0s 19ms/step

1/1 [==========================] - 0s 19ms/step

1/1 [==========================] - 0s 19ms/step

1/1 [==========================] - 0s 21ms/step

1/1 [==========================] - 0s 20ms/step

1/1 [==========================] - 0s 21ms/step

1/1 [==========================] - 0s 21ms/step

1/1 [==========================] - 0s 20ms/step

1/1 [==========================] - 0s 20ms/step

1/1 [==========================] - 0s 21ms/step

1/1 [==========================] - 0s 20ms/step

1/1 [==========================] - 0s 21ms/step

1/1 [==========================] - 0s 20ms/step

1/1 [==========================] - 0s 20ms/step

1/1 [==========================] - 0s 20ms/step

1/1 [==========================] - 0s 19ms/step

1/1 [==========================] - 0s 19ms/step

1/1 [==========================] - 0s 20ms/step

Figure 8: Recognize currency in time cost and accuracy recognition in percentage

Now, there is plot the confusion matrix to visualize the exact number of classifications with recognized currency are:

Figure 9: Basic CNN model plot Confusion matrix and another basic CNN model with changes value of padding, stride and filters used

Similarly training accuracy in pretrained model ResNet50 and VGG19 are:

Epoch 6/25

280/280 [========================] - 1055s 4s/step - loss: 0.2832 - accuracy: 0.9136 - val\_loss: 2.5237 - val\_accuracy: 0.6726

Epoch 7/25

280/280 [========================] - 1053s 4s/step - loss: 0.2777 - accuracy: 0.9187 - val\_loss: 0.9788 - val\_accuracy: 0.7988

Epoch 8/25

280/280 [========================] - 1050s 4s/step - loss: 0.2890 - accuracy: 0.9138 - val\_loss: 1.5266 - val\_accuracy: 0.7571

And VGG19 are:

Epoch 10/25

280/280 [==============================] - 1060s 4s/step - loss: 0.3403 - accuracy: 0.9011 - val\_loss: 0.4685 - val\_accuracy: 0.8393

Epoch 11/25

280/280 [==============================] - 1063s 4s/step - loss: 0.3164 - accuracy: 0.9064 - val\_loss: 0.4070 - val\_accuracy: 0.8631

Epoch 12/25

280/280 [==============================] - 1054s 4s/step - loss: 0.3071 - accuracy: 0.9067 - val\_loss: 0.4312 - val\_accuracy: 0.8726

Hence, the accuracy scores of all models listed in the table below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |

Table 5: Model Evaluation Result

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Epoch | Training Accuracy | Validation Accuracy | Test Accuracy |
| Basic CNN | 16 | 96.18% | 94.88% | 97% |
| Modified Basic CNN | 12 | 93.38% | 92% | 96% |
| ResNet50 | 8 | 91.38 | 75% | 92% |
| VGG19 | 12 | 90.66% | 87% | 90% |

This above accuracy results the CNN basic model is better than CNN modified basic model, ResNet50 and VGG16 model using training Nepali Currency dataset for image recognition.

# **RESULT**

The results of this seminar study demonstrate significant success in recognition of Nepali Currency with good accuracy. Through the meticulous training of convolutional neural network models using a dataset comprising 15,400 Nepali currency images, an impressive average accuracy of 97% was achieved, in same model with change (kernel size, stride, padding, filter size) give average 93% and 91% and another pretrained CNN model ResNet50, VGG19 model give less than 90% accuracy in average. These findings underscore the potential of the recognize for practical applications, while also highlighting avenues for further refinement through continued data collection and experimentation with advanced techniques and model architectures.

# **CONCLUSION**

In conclusion, it successfully built a Nepali Currency Recognition and Classification system using application. This system exhibits robust capabilities in accurately identifying Nepali currency notes through basic Convolution Neural Network (CNN) model. The custom CNN system developed for currency recognition boasts high accuracy in distinguishing authentic notes from counterfeit ones. However, it is important to note that this system currently focuses solely on recognizing the paper based Nepali currency note. Moving forward, our future endeavors will encompass integrating features from the flip side of the currency notes to further enhance counterfeit detection capabilities and fake note identification capabilities.

# **REFERENCES**

|  |  |
| --- | --- |
| [1] | N. F. o. t. D. -. N. (NFDN)., "Disability Data from Nepal Census 2021," NFDN Admin, 2023, April 4. |
| [2] | T. Adhikari, "Designing a Convolutional Neural Network for Image Recognition: A Comparative Study of Different Architectures and Training Techniques," *Available at SSRN 4366645,* 2023. |
| [3] | N. A. a. F. S. M. a. E. M. S. a. G. A. F. Semary, "Currency recognition system for visually impaired: Egyptian banknote as a study case," *2015 5th International Conference on Information \& Communication Technology and Accessibility (ICTA),* pp. 1--6, 2015. |
| [4] | M. J. P. M. P. A. M. Buvana, "Currency Recognition for Visually Impairment People using Deep Learning Algorithm," *International Journal of Research in Engineering and Science (IJRES),* vol. vol. 11, pp. no. 4, pp. 732-738,, April, 2023. |
| [5] | Q. a. Y. W. Q. Zhang, "Currency detection and recognition based on deep learning," *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS),* pp. 1--6, 2018. |
| [6] | O. K. P. J. G. D. N. R. S. A. Antre, "Fake Currency Detection Using Convolution Neural Network," *International Research Journal of Modernization in Engineering Technology and Science (IRJMETS),* vol. vol. 5, p. no. 4, April, 2023. |