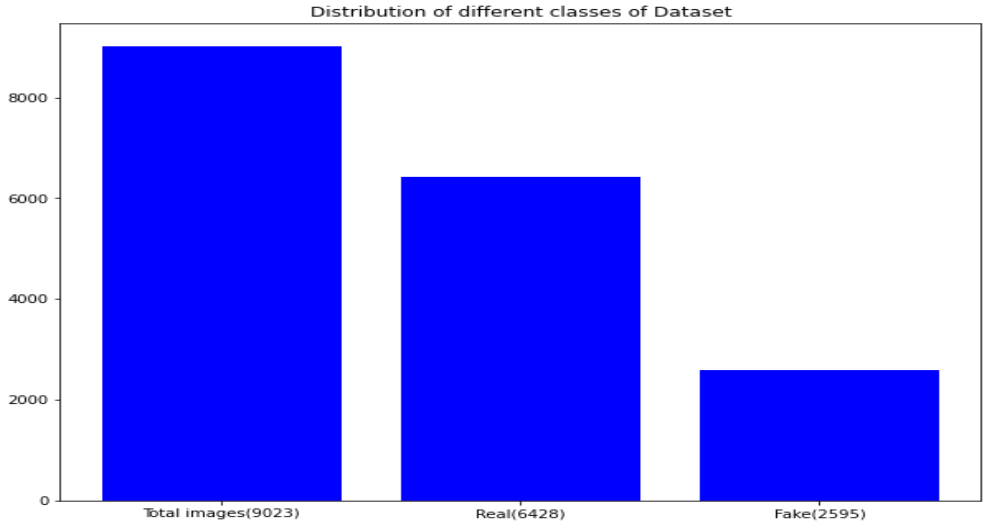
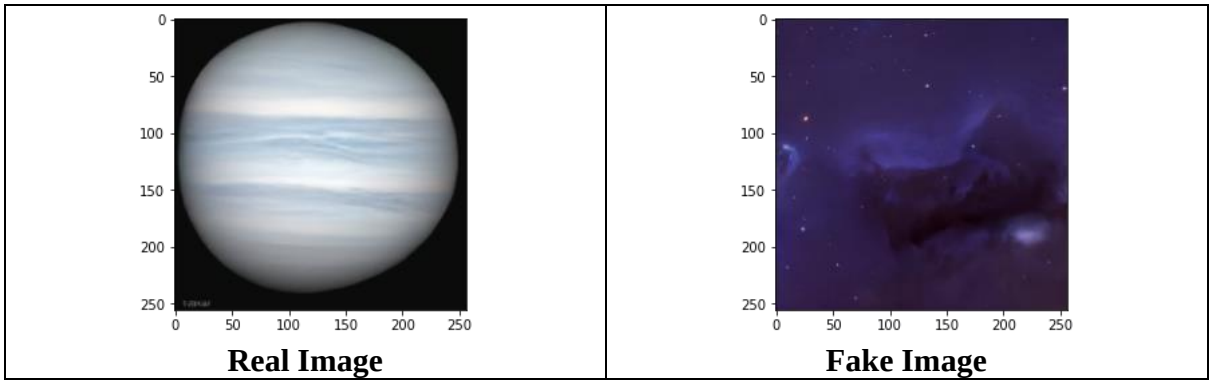
APOD Image Classification using CNN model.

Due to frequent increase of tampering in digital images, automatic detection and classification by advanced techniques of machine learning has become a serious challenge. The purpose of this work is to develop and implement framework for deep learning algorithm using Convolutional Neural Network (CNN) for automatic classification of images from the astronomy picture of the day dataset to achieve promising results. CNN is type of feed forward artificial neural network algorithm inspired by the human neurons that will be used to categorize the images into classes of real and fake image on the basis of the labels. The dataset used was downloaded from the repository using
https://www.apod.nasa.gov/apod/ap21109.html.
This repository contains the zip folder for Training and Validation image dataset. Each of these folders further contains the folders of Real and images. There were about 5000 images in real image and 2000 images in fake image folder which was found to contain the sufficient amount of data images for efficient classification. The total number of images with real and fake images can be seen in the figure below.



Total Number of Images= 9023
Real Images =6428
Fake Images=2595



The implementation of the model was done using the open source deep leaning framework of google TensorFlow. All the implementation was done using Keras written in Python 3 language using the google Colab environment. The backend training was done using TensorFlow.

Preprocessing

In this dataset we found the dataset was separate, so we combined the dataset and then used the python splitfolder library to randomly split the data into training, testing and validation data with the ratio of 70%, 20% and 10% respectively. There was class imbalance issue so our group decided to use data augmentation to build random images using operation of flipping images horizontally, zooming, rescaling, rotation etc. The training dataset is used purely for adjusting the internal parameters, e.g weights and biases during the training. Validation set is used to measure the generalization error and also used to measure the performance metrics.

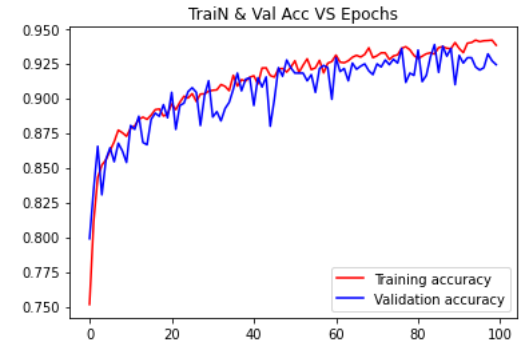
Structure of CNN

Convolution neural network used in our model uses structured layers of Convolution, activation, pooling, flatten and dense layer. Convolution layer is used to automatically extracts image features using the filter function. Activation is a non-linear layer that is used to set criteria for the firing of neuron and allow to learn non-linear mapping. We used the activation function of ReLU and sigmoid to remove the negative value. Pooling layer is used to reduce the number of parameters and computation. Dense layer is also called fully connected layer which is used at the end of the network to classify the images by using all the extracted features. We used sigmoid function at this layer to calculate the target class probability.

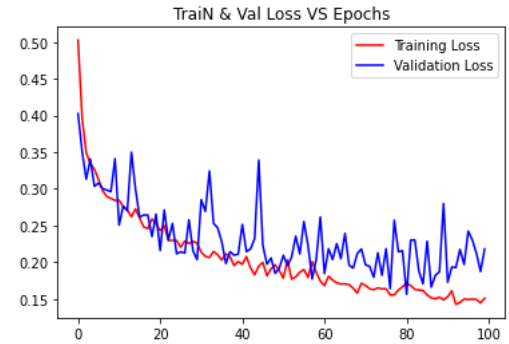
CNN was found robust against the overfitting, and this has increased it generalizability due to the use of dropout layer, batch normalization layer and regulation layer. Dropout is a regularization layer used to overcome overfitting during the training phase. In our model we set the dropout with a value of 0.25. The model uses 100 epochs with a batch size of 32 images to train the model. The figure of model structure and epoch are shown in figures below

Performance Evaluation

The performance of an implementated model is expressed in term of accuracy and error rate. The accuracy and loss function of the model are calculated for training and validation and these metrics were visualized by using Matplotlib library of python. The model achieves an overall classification accuracy of 94.16% on training data and 92.49% on validation data. The model was also compared to calculate the training loss and validation loss. There was an overall 0.15 training loss and 0.18 validation loss. The graph of model performance are shown in figure below.



(a) Graph of Training and Validation Accuracy



(b) Graph of Training and Validation Loss

The training accuracy and training loss with validation accuracy and validation loss with the epoch is shown below

```
Epoch 81/100
198/198 [==============================] - 85s 429ms/step - loss: 0.1719 - accuracy: 0.9284 - val_loss: 0.1562 - val_accuracy: 0.9346
Epoch 82/100
198/198 [==============================] - 87s 441ms/step - loss: 0.1680 - accuracy: 0.9302 - val_loss: 0.2300 - val_accuracy: 0.9119
Epoch 83/100
198/198 [==============================] - 85s 428ms/step - loss: 0.1630 - accuracy: 0.9319 - val_loss: 0.2302 - val_accuracy: 0.9163
Epoch 84/100
198/198 [==============================] - 85s 429ms/step - loss: 0.1619 - accuracy: 0.9326 - val_loss: 0.1874 - val_accuracy: 0.9296
Epoch 85/100
198/198 [==============================] - 87s 440ms/step - loss: 0.1613 - accuracy: 0.9318 - val_loss: 0.1704 - val_accuracy: 0.9385
Epoch 86/100
198/198 [==============================] - 85s 429ms/step - loss: 0.1553 - accuracy: 0.9364 - val_loss: 0.2285 - val_accuracy: 0.9186
Epoch 87/100
198/198 [==============================] - 87s 437ms/step - loss: 0.1512 - accuracy: 0.9370 - val_loss: 0.1660 - val_accuracy: 0.9374
Epoch 88/100
198/198 [==============================] - 87s 440ms/step - loss: 0.1501 - accuracy: 0.9359 - val_loss: 0.1821 - val_accuracy: 0.9302
Epoch 89/100
198/198 [==============================] - 85s 428ms/step - loss: 0.1522 - accuracy: 0.9356 - val_loss: 0.1868 - val_accuracy: 0.9357
Epoch 90/100
198/198 [==============================] - 87s 438ms/step - loss: 0.1486 - accuracy: 0.9400 - val_loss: 0.2799 - val_accuracy: 0.9097
Epoch 91/100
198/198 [==============================] - 85s 431ms/step - loss: 0.1527 - accuracy: 0.9357 - val_loss: 0.1726 - val_accuracy: 0.9307
Epoch 92/100
198/198 [==============================] - 85s 429ms/step - loss: 0.1609 - accuracy: 0.9327 - val_loss: 0.1933 - val_accuracy: 0.9252
Epoch 93/100
198/198 [==============================] - 87s 439ms/step - loss: 0.1424 - accuracy: 0.9395 - val_loss: 0.1926 - val_accuracy: 0.9291
Epoch 94/100
198/198 [==============================] - 85s 428ms/step - loss: 0.1450 - accuracy: 0.9400 - val_loss: 0.2173 - val_accuracy: 0.9291
Epoch 95/100
198/198 [==============================] - 85s 427ms/step - loss: 0.1499 - accuracy: 0.9417 - val_loss: 0.1969 - val_accuracy: 0.9224
Epoch 96/100
198/198 [==============================] - 86s 434ms/step - loss: 0.1490 - accuracy: 0.9406 - val_loss: 0.2423 - val_accuracy: 0.9202
Epoch 97/100
198/198 [==============================] - 84s 424ms/step - loss: 0.1496 - accuracy: 0.9413 - val_loss: 0.2303 - val_accuracy: 0.9219
Epoch 98/100
198/198 [==============================] - 84s 425ms/step - loss: 0.1491 - accuracy: 0.9414 - val_loss: 0.2121 - val_accuracy: 0.9319
Epoch 99/100
198/198 [==============================] - 87s 439ms/step - loss: 0.1444 - accuracy: 0.9417 - val_loss: 0.1872 - val_accuracy: 0.9269
Epoch 100/100
198/198 [==============================] - 85s 431ms/step - loss: 0.1508 - accuracy: 0.9381 - val_loss: 0.2179 - val_accuracy: 0.9241
```

References:

1. Frank J, Eisenhofer T, Schönherr L, Fischer A, Kolossa D, Holz T. Leveraging frequency analysis for deep fake image recognition. InInternational conference on machine learning 2020 Nov 21 (pp. 3247-3258). PMLR.
2. Frank J, Eisenhofer T, Schönherr L, Fischer A, Kolossa D, Holz T. Leveraging frequency analysis for deep fake image recognition. InInternational conference on machine learning 2020 Nov 21 (pp. 3247-3258). PMLR.
3. Hamid Y, Elyassami S, Gulzar Y, Balasaraswathi VR, Habuza T, Wani S. An improvised CNN model for fake image detection. International Journal of Information Technology. 2022 Nov 16:1-1.
4. Ahmed SR, Sonuç E, Ahmed MR, Duru AD. Analysis Survey on Deepfake detection and Recognition with Convolutional Neural Networks. In2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA) 2022 Jun 9 (pp. 1-7). IEEE.
5. Salman FM, Abu-Naser SS. Classification of Real and Fake Human Faces Using Deep Learning. International Journal of Academic Engineering Research (IJAER). 2022;6(3).