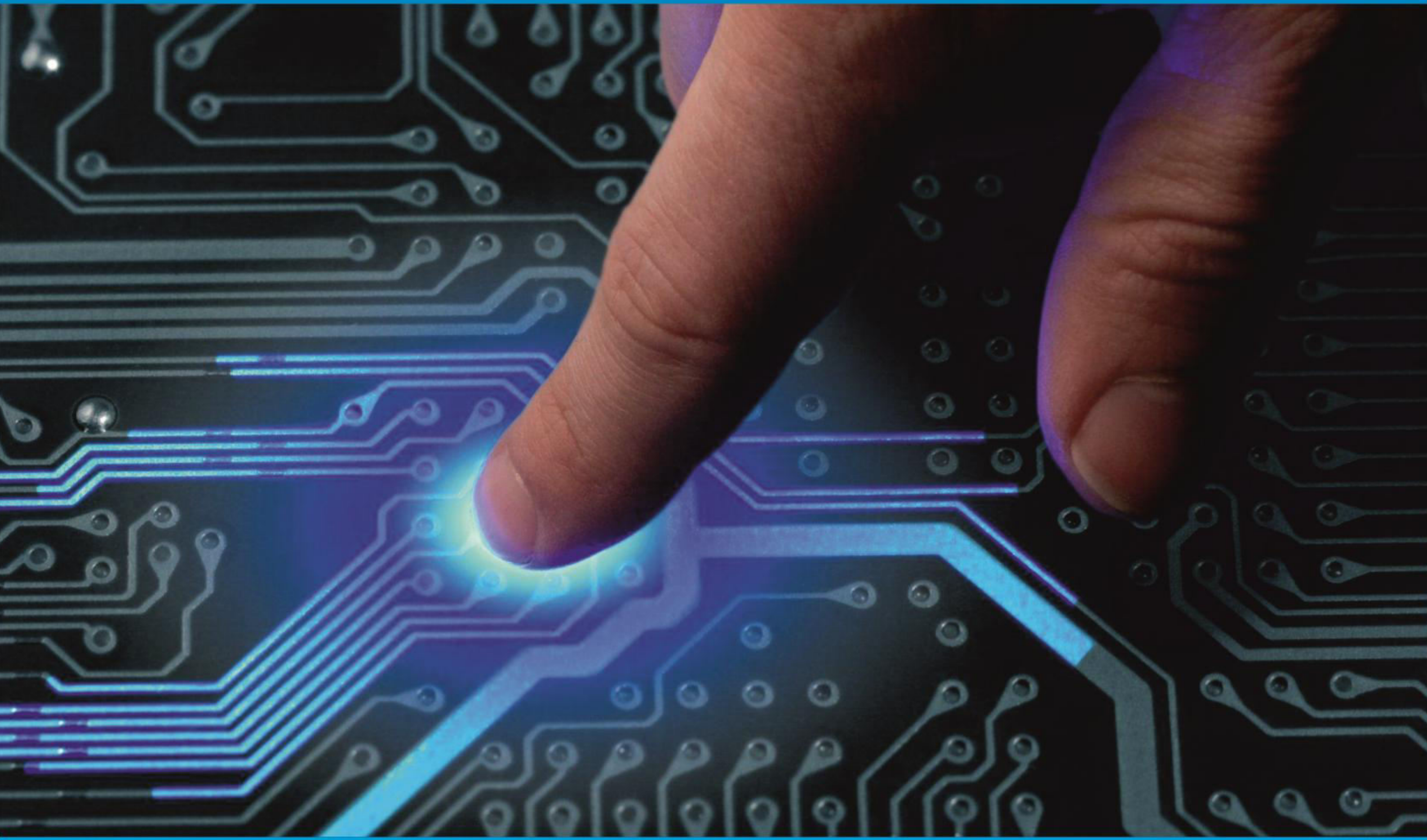
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 10, Issue 7, July 2022

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.165

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.165 |

|| Volume 10, Issue 7, July 2022 ||

| DOI: 10.15680/IJIRCCE.2022.1007056 |

Facial Image Manipulation Detection

Sneha H R[1], N R Ashika [2] , Nowfah K I [3] , Priyanka R. [4] , Ms. Meghana H.M[5]

Students, Department of Computer Science and Engineering, Malnad College of Engineering, Hassan, Karnataka, India[1,2,3,4]

Assistant Professor, Department of Computer Science and Engineering, Malnad College of Engineering, Hassan, Karnataka, India[5]

ABSTRACT: Deepfake algorithms can create fake images that humans cannot distinguish from authentic ones. Identifying the authenticity and processing history of an image is an important task in multimedia forensics. By analyzing traces left by different image manipulations, this is capable of detecting targeted editing operations. CNNs are capable of learning classification features directly from data, in their existing form. Using this approach calculating a confidence for fakeness and provided it as a validation point. This method is based on the observations that current Deepfake algorithm can only generate images of limited resolutions, which need to be further warped to match the original faces in the source Image. Such transforms leave distinctive artifacts in the Deepfake images. This targets the artifacts in affine face warping as the distinctive to distinguish real and fake content. This uses Face Forencics++ dataset and considered 600 videos which are further divided into 70 percent training and 30 percent testing

KEYWORDS: Deep fake algorithm, CNNs, FaceForencics++

I. INTRODUCTION

A deepfake is a Image/Video that is edited and manipulated, performed morphing on facial data or on the whole body. Deepfakes use a form of artificial intelligence called deep learning to make images of fake events, hence the name deepfake generated. We observe that the availability of huge social media platforms that can reach over the globe, and the data can be accessible to all over world, this data (i.e photos, videos , voice) was being misused to create deepfakes. Deepfakes were made, by using what's called a generative adversial network, or Gan. A Gan pits two artificial intelligence algorithms against each other. The first algorithm, known as the generator, is fed random noise and turns it into an image. This synthetic image is then added to a stream of real images – of celebrities, say – that are fed into the second algorithm, known as the discriminator. At first, the synthetic images will look nothing like faces. But repeat the process countless times, with feedback on performance, and the discriminator and generator both improve. Given enough cycles and feedback, the generator will start producing utterly realistic faces of completely non existent celebrities. It is hard to make a good deepfake on a standard computer. Most are created on high end desktops with powerful graphics cards or better still with computing power in the cloud. This reduces the processing time from days and weeks to hours. Plenty of tools are now available to 1 help people make deepfakes. Several companies will make them for you and do all the processing in the cloud. There's even a mobile phone app, Zao, that lets users add their faces to a list of TV and movie characters on which the system has trained. Spotting a deepfake is very hard these days as technology advances. In 2018, US researchers discovered that deepfake faces don't blink normally. No surprise there: the majority of images show people with their eyes open, so the algorithms never really learn about blinking. At first, it seemed like a silver bullet for the detection problem. But no sooner had the research been published, than deepfakes appeared with blinking. Such is the nature of the game: as soon as a weakness is revealed, it is fixed. Poor-quality deepfakes are easier to spot. The lip synching might be bad, or the skin tone patchy. There can be flickering around the edges of transposed faces. And fine details, such as hair, are particularly hard for deepfakes to render well, especially where strands are visible on the fringe. Badly rendered jewelry and teeth can also be a giveaway, as can strange lighting effects, such as inconsistent illumination and reflection on the iris

II. LITERATURE SURVEY

S. Bayram, et al. proposed a technique for the detection of doctoring in digital image. Doctoring typically involves multiple steps, which typically involve a sequence of elementary image-processing operations, such as scaling, rotation, contrast shift, smoothing, etc. The methodology used is based on the three categories of statistical features including binary similarity, image quality and wavelet statistics. The three categories of forensic features are as 1. Image Quality Measures 2. Higher Order Wavelet Statistics 3. Binary Similarity Measure. Swaminathan et al. proposed a method to estimate both in-camera and post-camera operation fingerprints for verifying the integrity of photographs. This paper introduces a new methodology for the forensic analysis of digital camera images. The proposed method is based on the observation that many processing operations, both inside and outside acquisition devices, leave distinct intrinsic traces on digital images, and these intrinsic fingerprints can be identified and employed to verify the integrity of digital data. The intrinsic fingerprints of the various in-camera processing operations can be estimated through a detailed imaging model and its component analysis. Further processing applied to the camera captured image is modelled as a manipulation filter, for which a blind deconvolution technique is applied to obtain a linear time-invariant approximation and to estimate the intrinsic fingerprints associated with these post camera operations. H. Cao et al. designed a new ensemble manipulation detector to simultaneously detect a wide range of manipulation types on local image patches. Fan et al. proposed to correlate statistical noise features with exchangeable image file format header features for manipulation detection. M. C. Stamm and K. J. R. Liu, proposed different methods not only for the detection of global and local contrast enhancement but also for identifying the use of histogram equalization and for the detection of the global addition of noise to a previously JPEG-compressed image. M. Stamm and K. Liu focuses on recovering the possible information about the unmodified version of image and the operations used to modify it, once image alterations have been detected. An iterative method based on probabilistic model is proposed to jointly estimate the contrast enhancement mapping used to alter the image as well as the histogram of the unaltered version of the image. The probabilistic model identifies the histogram entries that are the most likely to occur with the corresponding enhancement artifactss factor of the previous JPEG compression and the amount of linear contrast enhancement.

III **REQUIREMENTS**

A. Hardware Requirements

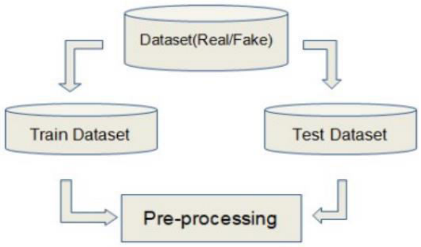
- Operating System – windows 10 SDK 41
- Processor – x86-64 architecture.(Tested with Intel i3 - above i3 4thGen gives good performance)
- Performance with CPU – Good
- RAM – minimum 4GB

B. Software Requirements

- Windows – windows 10 SDK 41 and Anaconda Navigator
- Python (Newest Versions) All below Modules would be installed in python
- Tensorflow – pip install tensorflow
- Streamlit – pip install streamlit
- Numpy – pip install numpy
- Matplotlib – pip install matplotlib
- Platform to run app.py – Heroku(PAAS) 34 3.2

IV. PROPOSED SYSTEM

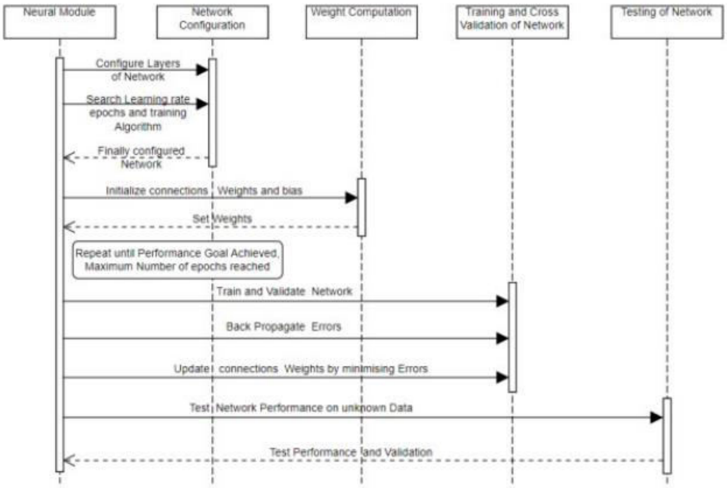
With the raise of face manipulation techniques such as Face2Face and Deepfake, more fake face content are spreading over internet, which brings serious challenges to public confidence. And authenticating this is important task as humans can't distinguish. Analyzing traces left by different Image Manipulations and using CNN's approach to detect Deepfakes based on validating Warping Artifacts. CNN's are capable of learning classification features directly from data, in their existing form. There are many tools available for creating the Deep Fake, but for DF detection there is hardly any tool available. Our approach for detecting the DF will be a great contribution. will be providing a web-based platform along with a standalone application for the user to upload the video and classify it as fake or real. This can be scaled up from developing a web-based platform to a browser plugin for automatic DF detections. Even big applications like WhatsApp, Facebook can integrate this project with their application for easy pre-detection of DF before sending to another user. One important objective is to evaluate its performance and acceptability in terms of security, user-friendliness, accuracy 40 and reliability. Our method is focusing on detecting all types of Deep Fake like replacement DF, retrenchment DF and interpersonal DF. And it goes through following Phases: 5.3.1 Dataset Here used a mixed dataset like Youtube, FaceForensics++.The newly prepared dataset contains 50% of the original images and 50% of the manipulated deepfake images. And then the dataset is split into 70% train and and 30% test set.



V. ANALYSIS AND DESIGN

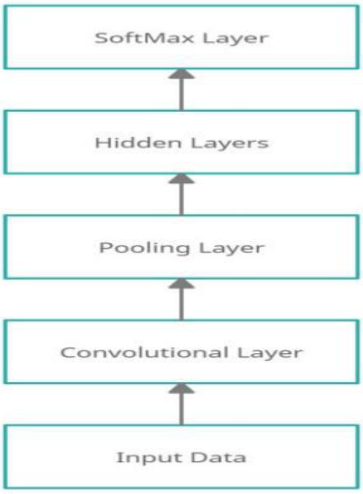
A. Sequence diagram:

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.


Neural Module
Network Configuration
Weight Computation
Training and Cross Validation of Network
Testing of Network
Configure Layers of Network
Search Learning rate epochs and training Algorithm
Finally configured Network
Initialize connections Weights and bias
Set Weights
Repeat until Performance Goal Achieved, Maximum Number of epochs reached
Train and Validate Network
Back Propagate Errors
Update connections Weights by minimising Errors
Test Network Performance on unknown Data
Test Performance and Validation

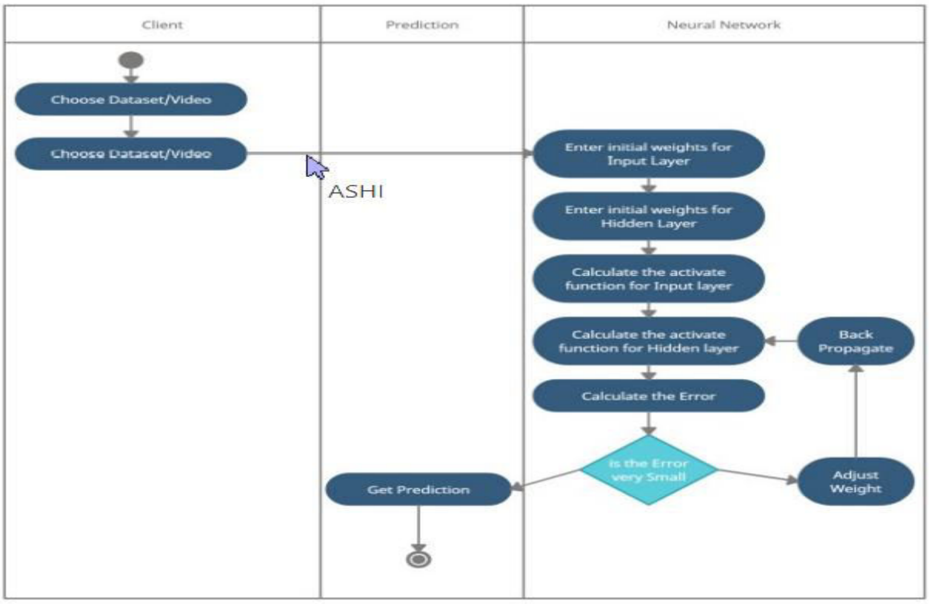
B. State Chart Diagram:

A state diagram is a type of diagram used in computer science and related fields to describe the behavior of systems. State diagrams require that the system described is composed of a finite number of states; sometimes, this is indeed the case, while at other times this is a reasonable abstraction



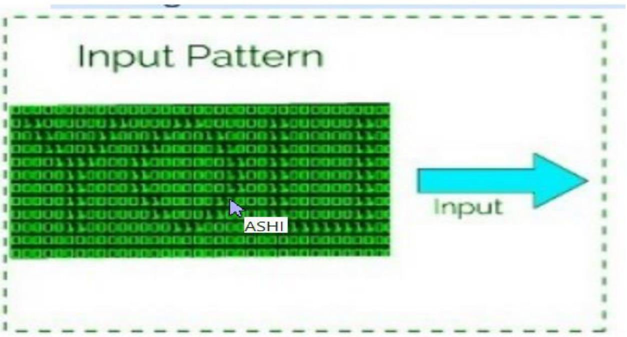
C. Activity diagram:

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.


Client
Prediction
Neural Network
Choose Dataset/Video
Choose Dataset/Video
ASHI
Enter initial weights for Input Layer
Enter initial weights for Hidden Layer
Calculate the activate function for Input layer
Calculate the activate function for Hidden layer
Back Propagate
Calculate the Error
Is the Error very Small
Adjust Weight
Get Prediction

D. Pre-processing:

pre-processing the input image data to convert it into meaningful floatingpoint tensors for feeding into Convolutional Neural Networks. Just for the knowledge tensors are used to store data, they can be assumed as multidimensional arrays. A tensor representing a 64 X 64 image having 3 channels will have its dimensions (64, 64, 3). Currently, the data is stored on a drive as JPEG files, So let's see the steps taken to achieve it. Algorithm: • Read the picture files (stored in data folder). 41 • Decode the JPEG content to RGB grids of pixels with channels. • Convert these into floating-point tensors for input to neural nets.



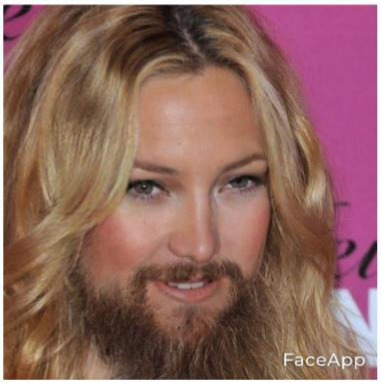
It may seem a bit fussy, but Keras has utilities to take over this whole algorithm and do the heavy lifting for you. Keras has a module with image-processing helping tools, located at keras.preprocessing.image. It contains the class ImageDataGenerator, which lets you quickly set up Python generators that can automatically turn image files on disk into batches of preprocessed tensors

E. Fitting the model:

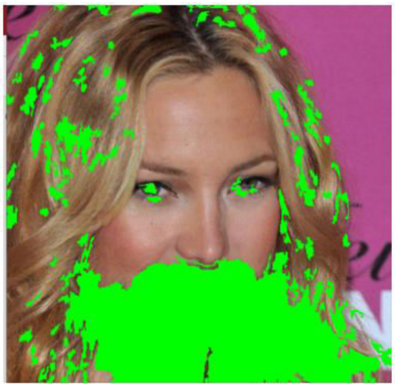
Let's fit the model to the data using the generator, it is done using the fit generator method, the equivalent of fit for data generators like given below. Its first argument is a Python generator that will yield batches of inputs and targets indefinitely because the data is being generated endlessly, the Keras model needs to know how many samples to draw from the generator before declaring an epoch over. This is the role of the steps per epoch argument. Now deciding the steps per epoch parameter, as we have total of 2000 training images and each batch is of size 20, hence, the steps per epoch will be 2000 / 20 = 100.

VI. RESULTS

Snapshot Showing the Real Image



Snapshot Showing the Fake Image



Snapshot showing manipulated region in the image

VII. CONCLUSIONS

Two classes of data REAL and FAKE are chosen for testing and validation of image classification using deep learning. The convolutional neural network used is ResNext architecture for classification purposes. From the experiments, it is observed that the images are classified correctly and shows the effectiveness of deep learning algorithms. These data sets were used both for training and testing purposes using CNN. This project presents a brief survey on image manipulation detection methods for contrast enhanced and cut-and paste type of forged images. Many approaches have been proposed for such type of retouching forgery detection, each one has certain merits and demerits. The techiques described overcome the limitations of previous approaches. The techniques that are robust against the post processing operations and anti-forensic techniques need to be developed

REFERENCES

1. "A . C. Popescu and H. Farid. "exposing digital forgeries by detecting traces of resampling.
2. J. W. Huang " W. Q. Luo and G. P. Qiu. "jpeg error analysis and its applications to digital image forensics.
3. I. Sutskever " A. Krizhevsky and G. Hinton. ""imagenet classification with deep convolutional neural networks.
4. J. Fridrich and J. Kodovsk'y. Rich models for steganalysis of digital images. " IEEE Trans. Inf. Forensics Security, vol. 7
5. M. Kirchner and T. Gloe. On resampling detection in re-compressed images. In IEEE Int. Workshop Inf. Forensics Security
6. Belhassen Bayar and Matthew C. Stamm. "constrained convolutional neural networks:a new approach towards general purpose Image manipulation detection

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING