Image Forgery Detection

A major project report submitted in partial fulfilment of the requirement for the award of degree of

Bachelor of Technology

in

Computer Science & Engineering / Information Technology

Submitted by

Rohit Ranjan (201531)

Anuj Taneja (201271)

Under the guidance & supervision of

Dr. Nancy Singla

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology,

Waknaghat, Solan - 173234 (India)

CERTIFICATE

This is to certify that the work which is being presented in the project report titled **"Image Forgery Detection"** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** submitted in the department of **Computer Science & Engineering and Information Technology**, Jaypee University of Information Technology, Waknaghat is an authentic record of work carried out by **Rohit Ranjan (201531)** and **Anuj Taneja (201271)** during the period from August 2023 to May 2024 under the supervision of **Dr. Nancy Singla**, Assistant Professor (SG), **Department of Computer Science and Engineering and Information Technology**.

Rohit Ranjan (201531) **Anuj Taneja (201271)**

The above statement made is correct to the best of my knowledge.

Dr. Nancy Singla
Assistant Professor (SG)
Department of CSE & IT
Jaypee University of Information Technology

CANDIDATE'S DECLARATION

We hereby declare that the work presented in this report entitled **'Image Forgery Detection'** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering / Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from August 2023 to May 2024 under the supervision of **Dr. Nancy Singla** (Assistant Professor, Department of Computer Science & Engineering and Information Technology).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Rohit Ranjan
Roll No.: 201531

Anuj Taneja
Roll No.: 201271

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Supervisor Name: Dr. Nancy Singla
Designation: Assistant Professor (SG)
Department: Computer Science and Engineering/Information Technology
Dated:

ACKNOWLEDGEMENT

First, I express my heartiest thanks and gratefulness to the Lord for His divine blessing to make it possible to complete the project work successfully.

We owe our profound gratitude and indebtedness to our project supervisor **Dr. Nancy Singla**, who took keen interest and guided us all along in our project work titled **―Image Forgery Detection**, till the completion of our project by providing all the necessary information for developing the project. The project development helped us in research, and we got to know a lot of new things in our domain. We are really thankful to her.

I would also generously welcome each one of those individuals who have helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, educating and non-instructing, who have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

Rohit Ranjan
(201531)

Anuj Taneja
(201271)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviation	Full Form
CNN	Convolutional Neural Network
NIST	National Institute of Standards and Technology
GAN	Generative Adversarial Network
VGG	Visual Geometry Group
ELA	Error Level Analysis
IRVM	Improved Relevance Vector Machine
CMFD	Copy-Move Forgery Detection
BWT	Biorthogonal wavelet transform
SVD	Singular value decomposition
GSO	Glowworm Swarm Optimization
SVM	Support Vector Machine
JPEG	Joint Photographic Experts Group
IDE	Integrated Development Environment
RAM	Random Access Memory
PiL	Pillow
SRM	Steganalysis Rich Model

ABSTRACT

This project introduces a method employing Convolutional Neural Networks (CNNs) to identify manipulated areas within digital images, aiming to address growing concerns regarding image authenticity. Leveraging CNNs' capability to extract complex features from images, the model learns distinctive patterns encompassing both local and global characteristics indicative of tampering.

A diverse dataset comprising various forgery types (e.g., copy-move and splicing) is utilized to train and assess the CNN model's performance. Enhancing the model's resilience, pre-processing techniques like noise reduction, resizing, and augmentation are implemented. To prevent overfitting and boost generalization, novel loss functions and regularization techniques are integrated.

The experimental analysis showcases the CNN model's proficiency in accurately identifying manipulated regions across different types of manipulations, exhibiting high precision and recall rates. These results highlight the model's potential for practical use in image forensics, contributing significantly to preserving content integrity in an era marked by rampant digital alterations.

In summary, this project introduces a CNN-based approach adept at detecting and pinpointing manipulated regions within digital images, offering a promising solution to uphold the credibility and trustworthiness of visual content.

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

Detecting manipulations in digital images has become a critical concern due to the increasing sophistication of editing tools that challenge the reliability and authenticity of visual content. This project focuses on the implementation of a Convolutional Neural Network (CNN) based method to identify and locate manipulated regions within digital images accurately.

Due to the easy accessibility of image and video editing software, various forms of manipulations, such as copy-move, splicing, and retouching have emerged, posing threats to maintaining the credibility of visual content. Convolutional Neural Network (CNN) offers promise in this regard due to its ability to learn complex patterns and features, resulting in it being a good solution for detecting image forgeries.

In order to improve the performance of the CNN model various preprocessing techniques are employed. These techniques include reducing noise, resizing images and using augmentation methods. Additionally innovative approaches such, as incorporating loss functions and regularization techniques are implemented to prevent overfitting and enhance the model's adaptability.

The main objective of this project is to contribute to the advancement of image forensics by introducing a methodology based on CNNs. This methodology aims to detect and locate forged regions within images. By doing it seeks to strengthen trust and credibility in media by safeguarding against falsified or misleading visual content. Through the use of techniques this project aims to address the challenges posed by image manipulation and provide a solution for ensuring the integrity and authenticity of visual information.

1.2 PROBLEM STATEMENT

In today’s era image forgery has become a problem due to the ease with which photographs can be manipulated using cutting edge editing tools and techniques. Given the use of media, online journalism and digital communication platforms it is crucial to develop accurate methods, for detecting image fraud. In an age where misinformation's prevalent, distinguishing between photographs and digitally altered ones can be challenging. This poses difficulties in guaranteeing the trustworthiness and authenticity of content.

The challenge is to provide efficient image forgery detection methods that can instantly recognize and categorize altered photos while separating them from authentic, unaltered ones. This calls for the creation of algorithms capable of analyzing many forms of picture modifications, including copy-move, splicing, retouching, and more, in a variety of settings and at varied levels of complexity. The answer should support a variety of picture formats, resolutions, and digital platforms, allowing for various image content sources. Designing and implementing cutting-edge picture forensics tools that make use of computer vision, deep learning, and machine learning techniques is the main goal in resolving this issue. These technologies ought to be able to not only identify forgeries but also reveal details about the particular manipulation strategies used, helping us to comprehend how forging strategies are changing.

To protect the integrity of visual information, the challenge of image fraud detection necessitates cutting-edge research and development. Successful solutions will enable people, organizations, and platforms to base judgments on real visual content, reducing the spread of fake news and maintaining the legitimacy of digital imagery. The development of precise forgery detection algorithms is an urgent need in today’s increasingly digital and networked society in order to guarantee the reliability of photographs.

1.3 OBJECTIVES

- Developing a Convolutional Neural Network (CNN) Model for Copy-Move Forgery Detection.
- Developing a Convolutional Neural Network (CNN) Model for Splicing Detection.
- Development of a User-Friendly Web-Interface Integrated with the Trained Model

1.4 SIGNIFICANCE AND MOTIVATION OF THE PROJECT WORK

Significance - The significance of developing an image forgery detection model extends across various domains, addressing critical challenges in the digital landscape while fostering trust, credibility, and authenticity in the realm of visual media. The project's contributions hold immense importance in several aspects –

- Combating Misinformation and Fake Visual Content:

  In an era where misinformation and fake visual content proliferate across digital platforms, the development of an accurate image forgery detection model serves as a potent tool against deceptive practices. It helps in identifying manipulated images, thereby curbing the spread of misleading information, fake news, and fabricated visuals that can sway public opinion, damage reputations, or propagate false narratives.

- Safeguarding Image Integrity in Various Sectors:

  Across industries such as journalism, advertising, healthcare, and academia, ensuring the integrity and authenticity of visual content is paramount. The image forgery detection model acts as a safeguard, protecting the credibility of visual data used in research, publications, medical imaging, marketing campaigns, and various professional domains. It ensures that decisions and actions based on visual information are founded on authentic and unaltered data.

- Fostering Trust and Reliability in Digital Media:

  As the digital landscape continues to evolve, establishing trust and reliability in digital media content becomes increasingly crucial. The project's contribution in accurately detecting image forgeries promotes transparency and trustworthiness in the content shared online, strengthening confidence among users, consumers, and stakeholders in the authenticity of visual information.

- Technological Advancements and Innovation in Image Analysis:

  The development of an advanced image forgery detection model signifies a leap forward in technological innovation. It drives research and development in the field of image analysis, encouraging the exploration of new methodologies, algorithms, and techniques aimed at enhancing detection accuracy, expanding the model's capabilities, and staying ahead of evolving forms of image manipulation and forgery.

Motivation –

- Mitigating Misinformation: In an era flooded with manipulated visuals, creating an image forgery detection model stems from the urgent need to combat misinformation. Such a model acts as a shield against the proliferation of fake news, deceptive propaganda, and digitally altered images that distort reality.

- Preserving Authenticity in Digital Media: The motivation arises from the necessity to preserve the integrity and authenticity of digital media. By detecting image forgeries, this model ensures that visual content used in various industries, such as journalism, healthcare, and advertising, maintains its credibility and trustworthiness.

- Empowering Forensic Analysis: Developing this model is driven by the goal of empowering digital forensics experts and law enforcement agencies with a robust tool for verifying the

authenticity of visual evidence. It aids in conducting thorough investigations and ensuring the integrity of evidence presented in legal proceedings.

- Enhancing Trust and Reliability: The motivation also lies in fostering trust and reliability in digital content. By offering a reliable means of distinguishing between users and stakeholders in the authenticity of visual information shared online.

- Advancing Technological Innovation: The endeavor to create an image forgery detection model is motivated by the aspiration to push the boundaries of technological innovation. It encourages research and development in image analysis techniques, driving advancements to stay ahead of evolving methods of image manipulation.

1.5 ORGANIZATION OF PROJECT REPORT

The report is organized as follows –

Chapter 1 presents us with the introduction of image forgery, along with the problem statement, objectives, significance and organization of the project.

Chapter 2 outlines the existing related works in the domain of image manipulation detection, further providing the outputs of their analysis which is compared and used in the project.

Chapter 3 introduces the model that has been trained and tested to detect forgeries in digital images. This is the chapter that includes the requirements, analysis, the project design and its architecture and the implementation of the project.

Chapter 4 delves in the training and testing phase, by examining datasets, experimental setups and both qualitative and quantitative results obtained from the model all the while assessing model performance.

Chapter 5 examines the image forgery detection outcomes and the overview of the experimental setup.

Chapter 6 presents the conclusion of the study along with future scopes of the project.

CHAPTER 2: LITERATURE SURVEY

2.1 OVERVIEW OF RELEVANT LITERATURE

Image forgery detection is a critical field encompassing a vast array of research endeavors aimed at uncovering manipulated digital images. The literature on this subject showcases multifaceted methodologies that traverse traditional image forensics techniques to contemporary deep learning approaches. Forensic analysis methods constitute a significant portion of this domain, where passive techniques rely on statistical properties, noise inconsistencies, and compression artifacts, while active methods employ watermarking, digital signatures, and cryptographic mechanisms. Understanding the distinctive features of various manipulation types, such as copy-move, splicing, and retouching, is pivotal in the development of effective detection algorithms.

The emergence of deep learning has revolutionized image forgery detection. Convolutional Neural Networks (CNNs) and their derivatives have demonstrated remarkable potential in discerning manipulated images. Researchers have proposed diverse architectures, including Siamese networks, GAN-based models, and attention mechanisms, to extract discriminative features for identifying forgeries. The utilization of transfer learning and ensemble methods using deep networks has also shown promising results in handling diverse forgery types and complexities.

Datasets serve as crucial assets in the advancement of forgery detection algorithms. Benchmark datasets such as CASIA [1], NIST, and COVERAGE provide researchers with comprehensive sets of manipulated images for training and evaluating detection methods. Nevertheless, challenges persist regarding dataset biases, variations in image quality, and the necessity for standardized evaluation metrics to accurately assess the performance of detection algorithms.
The rapid evolution of image manipulation techniques presents new challenges in forgery detection. Generative models, particularly Generative Adversarial Networks (GANs), produce highly realistic forgeries that pose significant obstacles in differentiating them from authentic

images. Tackling issues related to robustness, scalability, and real-time detection remains a focal point in current research endeavors.

Beyond technical aspects, the literature also delves into ethical and legal implications. Ethical considerations encompass privacy concerns and the potential misuse of detection techniques. Scholars emphasize the need for ethical guidelines governing the development and application of forgery detection technology to mitigate unintended consequences and ensure responsible usage.

In summary, the literature on image forgery detection is diverse, spanning traditional forensics techniques to cutting-edge deep learning methods. While significant strides have been made, persistent challenges persist, emphasizing the necessity for continued research and development to counter increasingly sophisticated image manipulation techniques. The ethical considerations surrounding the use of these technologies further underscore the multidimensional nature of this field.

A Deep Learning based approach to detect image forgery is presented by N. P. Nethravathi et al. [2]. In particular, it compares performances of ELA and CNNs along with pre-trained VGG-16 model in detail. Deep learning techniques in detecting image forgery as ELA-CNN model gives 99.87% accuracy and correct detection of invisible images is 99%. However, the VGG-16 gives only an accuracy of 97.93%, and a validation rate of 75.87%.
It underscores the need for accurate image forgery detection devices in upholding digital materials' integrity as well as combating any adverse effects that come with picture manipulation within different fields like journalism, crime investigation, and cyber space. It investigates use of deep learning-based algorithms in identifying fakes and how they can become effective detection systems. The paper notes some challenges including the effect of ELA quality on model success and potential areas for future work include exploring different data processing approaches as well as validating on wider cohorts. In general, these results should emphasize ongoing measures undertaken to tackle digital image falsification and its negative outcomes by means of advancing image tampering detection algorithms.

M. Zanardelli et al. (2022) [3]. scrutinized modern methods for detecting image manipulation, specifically in identifying copy-move, splicing, and DeepFake alterations. A detection method showcased exceptional performance by nearly perfect accuracy across standard datasets for identifying copied regions and their original sources. Splicing detection achieved an accuracy of 85% on the CASIA2 dataset, accurately pinpointing forged regions.

However, in DeepFake detection, no single method emerged as dominant across benchmark datasets, with accuracies averaging around 65%. While some approaches showed promise, no definitive winning strategy was evident in this field. The study underscored the effectiveness of techniques integrating pre-processing, post-processing, and detection algorithms for achieving higher accuracy in detecting image tampering. Despite advancements in detecting certain manipulations, improving accuracy in detecting DeepFakes remains a significant challenge.

The literature lacks robust methods to effectively detect and counter the surge in fake images and videos. With the widespread use of smart devices and sophisticated editing tools, there's a critical need for innovative forgery detection techniques, especially those leveraging deep learning.

An innovative method for detecting image forgeries using Convolutional Neural Network (CNN) was introduced by SS. Ali et al. [4]. This approach focuses on identifying tampering by exploiting the differences in image compression between original and altered segments. By analyzing the variations in recompressed images, the model distinguishes manipulated areas from genuine ones, effectively detecting both copy-move and splicing types of image tampering. Extensive evaluation on the CASIA2 [26] image forgery database showcases the effectiveness of the proposed CNN-based technique. It achieves a commendable accuracy of 92.23% in detecting image forgeries, surpassing existing methods that concentrate on pixel-level forgery detection. Notably, the method's swiftness in processing and its ability to identify various tampering methods indicate its practical potential for real-world applications due to its efficiency and reliability.

Future improvements include refining the technique for enhanced forgery localization, integrating it with other image localization methods, and expanding its capabilities to address concerns related to spoofing. Additionally, efforts are underway to adapt the method for

detecting forgeries in smaller-resolution images and to create a comprehensive forgery database for training and advancing deep learning-based forgery detection systems.
Despite the notable achievements in image forgery detection using CNNs and compression artifacts, this research lacks exploration into the technique's robustness across diverse image sizes and resolutions.

Portrayal of prevalent forms of image forgeries like retouching, copy-move forgery, and image splicing facilitated by easily accessible editing software is shown by J. Ega et al. [5]. in this research. Through vivid examples, it illuminates these forgery types, underlining their extensive presence and implications in societal platforms and media. The study further highlights the pervasive use of fake images for malicious intent, including political manipulation and false representation, underscoring their relevance in contemporary society. Concerning forgery detection, the paper contrasts active and passive approaches, with a primary focus on passive methods and a deep dive into copy-move forgery detection. It categorizes detection techniques into block-based, keypoint-based, and hybrid approaches, elucidating their roles in identifying copy-move forgery. The research expounds on challenges inherent in block-based detection, such as computational complexities, insensitivity to alterations, and the need for empirical block size selection, proposing strategies like lexicographic sorting of feature vectors to mitigate computational overhead.
It falls short in offering a detailed examination of recent developments or emerging techniques in forgery detection. This limitation could impact its applicability in the swiftly advancing landscape of digital forensics.

N. K. Rathore et al. [6] makes use of the Improved Relevance Vector Machine (IRVM), for detecting image forgery, specifically focusing on Copy-Move Forgery Detection (CMFD). The study addresses the challenges prevalent in existing methods related to feature extraction, computational time, and accuracy. The proposed IRVM system leverages Biorthogonal wavelet transform (BWT) combined with Singular value decomposition (SVD) for efficient feature extraction and classification. Through a step-by-step process, the IRVM system preprocesses images, reduces noise, extracts features, identifies duplicate vectors, and employs Glowworm Swarm Optimization (GSO) for optimal weight determination. Experimental evaluations

conducted across five sets of images demonstrate the IRVM's superior performance in accuracy, sensitivity, specificity, precision, recall, F-measure, and G-mean. The IRVM consistently displayed high accuracy rates, reaching 92.22%, across various sets, showcasing its effectiveness in detecting image forgeries.

The study's results illustrate the IRVM's robustness and efficiency in detecting manipulated images. Comparative analyses against existing methods, such as HMM, SVM and SVM-based forgery detection, highlight the IRVM's superior performance across multiple evaluation metrics. Its ability to achieve higher accuracy, sensitivity, and specificity rates, along with improved precision and recall, underscores the IRVM's efficacy in image forgery detection.

This study highlights a significant need for more robust and efficient methods to detect image forgery, particularly in scenarios involving copy-move operations. While the Improved Relevance Vector Machine (IRVM) shows promise, future research should focus on advanced machine learning or neural network techniques to enhance detection accuracy further. This investigation must cover various types of image manipulations and ensure efficiency, especially when dealing with large datasets.

Prevalent forms of manipulation like copy-move, resampling, splicing, and JPEG compression, all capable of altering or concealing information in satellite images are introduced by A. Kuznetsov (2019) [7]. To tackle the challenge of splicing, the study employs convolutional neural networks (CNNs), specifically a VGG-16-inspired architecture. Operating on fixed-size patches, this approach involves analyzing image fragments via a sliding window strategy, enabling discriminant function construction without predefined feature extraction. With convolutional and fully connected layers, augmented by dropout layers to mitigate overfitting, the proposed CNN architecture demonstrates robustness in detecting distortions, achieving an impressive accuracy of 97.8% (for fine-tuned) and 96.4% (for zero-stage trained) on the CASIA dataset, surpassing existing methods.

The experiments conducted validate the CNN-based solution's efficacy, even post JPEG compression, albeit with reduced accuracy. Despite a notable accuracy decline (from 67.1% to 66.3% for Q=90 to Q=80 compression), precision and recall rates remain reasonable. Comparative analysis against state-of-the-art techniques underscores the superiority of the proposed approach, marking a notable advancement in detecting image manipulations. Future

directions include further evaluation against diverse CNN models like Mobilenet and Resnet-50, a focus on detecting distorted areas, and comprehensive performance assessments against other splicing detection methodologies.

Table 1. Literature Review Table

S. No.	Paper Title [Cite]	Journal/ Conference (Year)	Tools/ Techniques/ Dataset	Results	Limitations
1.	Image Forgery Detection using Deep Neural Network [1]	IRJET (2023)	Convolutional Neural Network, Error Level Analysis, CASIA v1.0 Dataset.	Accuracy of 99.87%using ELA-CNN, Accuracy of 97.93% using VGG-16 model.	Limited amount of data for training deep networks.
2.	Image Forgery Detection: A Survey of recent Deep-Learning approaches [2]	Multimedia Tools and Applications (2022)	Convolutional Neural Networks. Transfer Learning. Generative Adversarial Networks.	Survey of recent methods of copy move and splicing detection.	-
3.	Image Forgery Detection using Deep Learning by Recompressing Images [3]	Electronics (2022)	Accuracy, Precision, Recall. CASIA v2.0 Dataset.	Achieved accuracy of 92.23% on CASIA v2.0 Dataset.	Model does not perform well for tiny images.
4.	A Review on Digital Image Forgery Detection [4]	IRPH (2021)	Pixel Level Analysis, Copy-Move Forgery Detection Dataset.	Hybrid approach-Block & keypoint based detection achieved the highest accuracy as compared to both the approaches separately.	Performance may vary based on dataset size.
5.	Image Forgery Detection using Singular Value	The National Academy	SVD Feature Extraction. CoMoFoD Dataset.	Achieved accuracy of 92.22% using	Small Image Size used

	Decomposition with some Attacks [5]	of Sciences, India (2020)		CoMoFoD dataset.	(512 x 512)
6.	Digital Image Forgery Detection using Deep Learning approach [6]	Journal of Physics (2019)	Data Collection, Data Preprocessing, Loss Functions, CASIAv2.0 Dataset.	The results showed an accuracy of 97.8% for fine-tuned model and 96.4% for the zero-stage trained.	Limited amount of training data. Fixed input patch size
7.	CNN based Image Forgery Detection using pre-Trained AlexNet ,l model [7]	ICCIIoT (2018)	Feature ExtractionEvaluationMetrics MICC-F220Dataset.	Achieved 93.94% accuracy, 100% recall, 89.19% precision.	Limited to MICC-F220 dataset. Limited discussion on the experimental results.
8.	Image Manipulation Detection using Convolutional Neural Network [8]	Research India Publications (2017)	Image Processing, High Pass filter, Hidden feature extraction.	Proposed algorithm effectively detects image manipulations achieving 95% accuracy.	No clear potential computational requirements stated.
9.	A Deep Learning Approach To Universal Image [9] Manipulation Detection Using A New Convolutional Layer [9]	IH&MM Sec '16	Convolutional Neural Network, Gaussian Blurring, Additive Gaussian white noise.	Model was able to detect manipulations with 99.31% accuracy.	Limited to specific image manipulations.

2.2 KEY GAPS IN THE LITERATURE

- Limited Training Data: Many studies highlight the challenge of limited data for training deep neural networks, which can hinder the performance and generalization of models.
- Model Performance on Small Images: Some models do not perform well on tiny images, indicating a need for improvement in handling small-scale images.
- Variability in Performance Based on Dataset Size: The performance of forgery detection models may vary depending on the size and quality of the dataset used for training and evaluation.
- Limited Discussion on Experimental Results: Some studies provide limited discussion on the experimental results, making it challenging to assess the reproducibility and reliability of the proposed methods.
- Limited to Specific Datasets: Some approaches are limited to specific datasets, such as the MICC-F220 dataset, which may restrict their applicability to other datasets or real-world scenarios.
- Unclear Computational Requirements: Some studies do not clearly state the computational requirements of the proposed algorithms, making it difficult to assess their feasibility for practical implementation.

CHAPTER 3: SYSTEM DEVELOPMENT

3.1 REQUIREMENTS AND ANALYSIS

Following are the requirements to run the project without any system errors or issues:

3.1.1 SOFTWARE REQUIREMENTS

The project has the following software requirements –

- Programming Language: Python 3.3 or higher
- Integrated Development Environment (IDE): Visual Studio Code (VS Code), Jupyter Notebook

3.1.2 HARDWARE REQUIREMENTS

The project has the following hardware requirements –

- Random Access Memory (RAM): 8 GB or higher
- CPU Requirements: 1.6 GHz
- Operating System: Windows

3.1.3 LIBRARY REQUIREMENTS

- Numpy – Numpy is a fundamental library for numerical computations and handling array operations, especially in image processing tasks.
- OpenCV – It is used for image processing tasks like filtering and resizing images.
- Streamlit – It is used to create the web application interface for uploading images and displaying the results of the forgery detection model.
- Tensorflow – Tensorflow is utilized for deep learning tasks.
- Matplotlib – It is used to visualize images, ELA results, and performance graphs during model training.
- PiL (Pillow) – It is Employed for image processing tasks like ELA, image manipulation.

3.2 PROJECT DESIGN AND ARCHITECTURE

- The Convolutional Neural Network (CNN) model (Fig 1) used in this study is based on the CASIA 2 dataset, a well-known benchmark dataset in the field of image manipulation detection. The dataset contains a vast collection of both original and altered photos, making it ideal for training and testing the model's performance.
- Before feeding the images into the CNN model, numerous preprocessing processes are used to improve the data's quality and compatibility.
- Image Resizing: The images in the CASIA 2 dataset vary in size, which can impair the model's performance. To remedy this, all photos are downsized to a common size (e.g., 256x256 pixels) using interpolation techniques that maintain the aspect ratio.
- Feature Selection: In addition to scaling, various feature selection techniques can be used to extract useful information from photos. This can include approaches like histogram equalization, which increases image contrast and allows the model to recognize minor variations between legitimate and altered photos.
- These preprocessing processes serve to prepare the dataset for training the CNN model, guaranteeing that it can learn and identify patterns associated with picture counterfeiting.

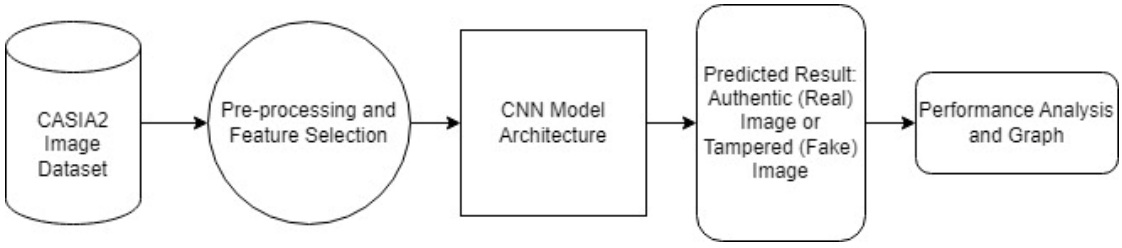


Fig 1: Project Design

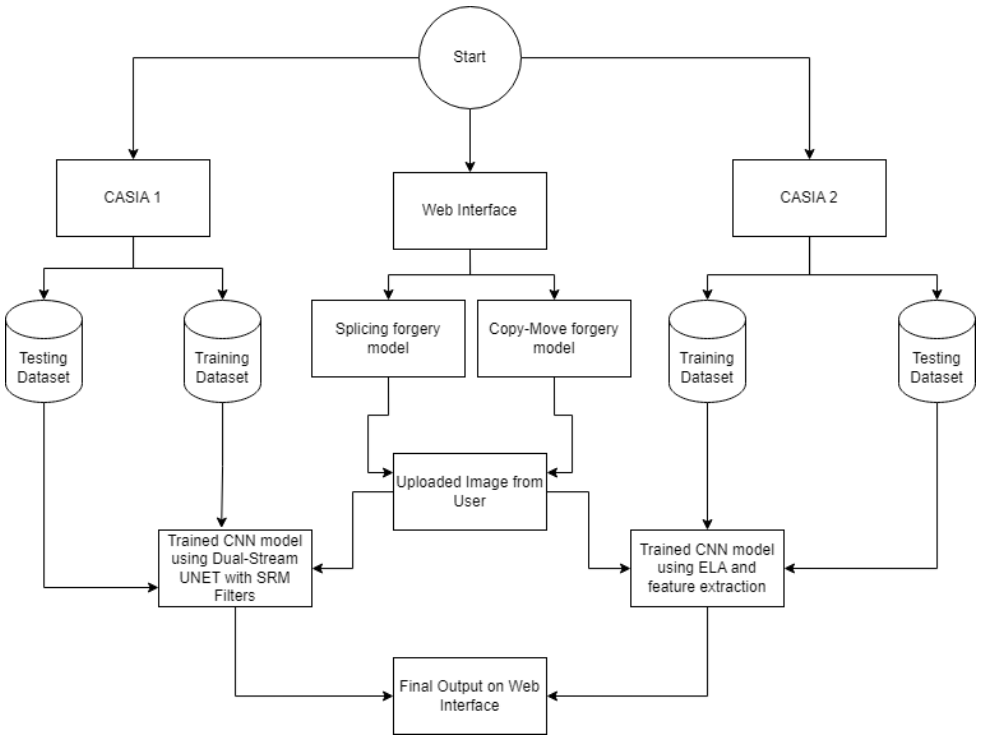


Fig 2: Project Architecture

- CASIA 1 dataset (Fig 2) is used to train and test the Splicing CNN model. A Dual-Stream UNET architecture is adopted for the second phase of Splicing forgery detection. The architecture consists of two streams: The first one is for processing the RGB images and the second one is for processing a noise feature map generated by passing the image through SRM (Steganalysis Rich Model) filters. The output from both the streams is then integrated and passed through a CNN layer with a sigmoid activation function in order to produce a binary image which will indicate all the manipulated regions in the input image.
- A combination of Error Level Analysis (ELA) and Convolutional Neural Network (CNN) is used along with CASIA 2 dataset to train and test the Copy-Move model. Based on the differences in the compression levels, ELA is used to identify the regions of an image that may have been altered.

3.3DATE PREPARATION

3.3.1 DESCRIPTION OF THE DATASET

This project requires a large number of images, consisting of both forged and authentic images which will be used to train te model. For this purpose, CASIA2 [26] dataset is being used which is readily available on Kaggle.

It consists of a total of 12,000 images with around 7000 of them being authentic images and around 5000 tampered images.

There are four folders in the dataset, namely –

- Tp – This folder contains the fake copy-move images
- Sp – This folder contains the fake spliced images
- Au – This folder contains all the real images
- CASIA 2 Groundtruth – This folder contains a fake image mask

3.3.2 DATA CLEANING AND PREPROCESSING

The dataset underwent several preprocessing steps to standardize the images for Copy-Move model training:

- Image Resizing: All images were resized to a uniform size of 128x128 pixels using bicubic interpolation to maintain image quality.
- ELA Image Generation: Error Level Analysis (ELA) images were generated for each original image in the dataset, aiding in identifying potential manipulations and compressions.
- ELA Image Generation: Error Level Analysis (ELA) images were generated for each original image in the dataset, aiding in identifying potential manipulations and compressions.

As for the Splicing model training, the dataset underwent the following preprocessing steps:

- Image Resizing: The RGB images were resized to 512x512 pixels to match the input size expected by the dual-stream UNET model.
- Data Splitting: The dataset was split into training and validation sets of 80% and 20% to evaluate the performance of the model.

3.4 IMPLEMENTATION

3.4.1 COPY-MOVE FORGERY DETECTION MODEL

System Overview – Currently, CASIA1 and CASIA2 datasets are being utilized, both of these datasets are available on Kaggle for the project. The task is divided into two phases. Initially, the goal is to classify images as either real or fake, constituting a classification problem. Subsequently, in the second phase, the aim is to predict the tampered regions within the images, resembling a Binary-Image Segmentation problem. While there is no strict latency requirement for immediate results, it is essential that the model does not take an excessive amount of time to determine both the authenticity of the image and identify the tampered regions.

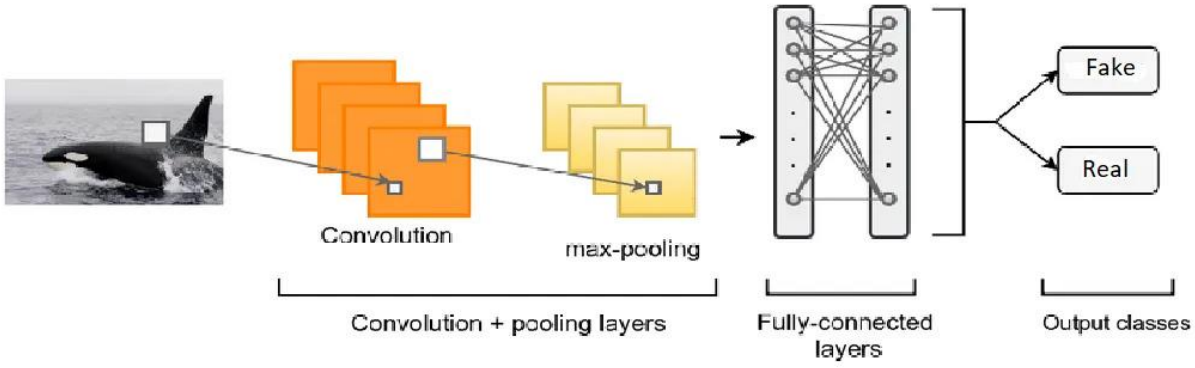


Fig 3: Copy-Move model architecture [18]

3.4.1.1 PHASE 1

In phase 1, the model aims to predict whether an image is real or fake, which is a classification problem. The model architecture is shown in Fig 3.

3.4.1.2 DATA EXPLORATION

- Begin by importing fundamental Python libraries for tasks such as data exploration, manipulation, and model development.
- Within the dataset, there are three distinct folders:

  Tp: This directory comprises of tampered images.

  CASIA2 Groundtruth: This directory encompasses masks for fake images.

  Au: This folder holds authentic, real images.
- To initiate the analysis, the main focus will be on fake images housed in the 'Tp' folder. As this directory also contains miscellaneous files, only those files with the extensions '.jpg' or '.png' will be specifically read and considered.

```
fake_image_data=pd.DataFrame(fake_image_data)
fake_image_data.head()
```

✓ 0.0s

	image_path	label	image_id
0	./CASIA2/Tp/Tp_D_CND_S_N_txt00028_txt00006_108...	fake	Tp_D_CND_S_N_txt00028_txt00006_10848
1	./CASIA2/Tp/Tp_D_CNN_M_B_nat00056_nat00099_111...	fake	Tp_D_CNN_M_B_nat00056_nat00099_11105
2	./CASIA2/Tp/Tp_D_CNN_M_B_nat10139_nat00059_119...	fake	Tp_D_CNN_M_B_nat10139_nat00059_11949
3	./CASIA2/Tp/Tp_D_CNN_M_B_nat10139_nat00097_119...	fake	Tp_D_CNN_M_B_nat10139_nat00097_11948
4	./CASIA2/Tp/Tp_D_CNN_M_N_ani00052_ani00054_111...	fake	Tp_D_CNN_M_N_ani00052_ani00054_11130

```
print("Number of fake images are {}".format(fake_image_data.shape[0]))
```

✓ 0.0s

```
Number of fake images are 2064
```

Fig 4: Number of fake images

There are a total of 2064 fake images in the dataset as shown in Fig 4, each associated with a corresponding mask image stored in the 'CASIA 2 Groundtruth' folder, indicating the tampered regions. However, it’s important to note that this mask information is not needed in the phase 1 of the analysis.

```
real_image_data=pd.DataFrame(real_image_data)
real_image_data.head()
```
✓ 0.0s

	image_path	label	image_id
0	./CASIA2/Au/Au_ani_00001.jpg	real	Au_ani_00001
1	./CASIA2/Au/Au_ani_00002.jpg	real	Au_ani_00002
2	./CASIA2/Au/Au_ani_00003.jpg	real	Au_ani_00003
3	./CASIA2/Au/Au_ani_00004.jpg	real	Au_ani_00004
4	./CASIA2/Au/Au_ani_00005.jpg	real	Au_ani_00005

```
print("Number of real images are {}".format(real_image_data.shape[0]))
```
✓ 0.0s

```
Number of real images are 7437
```

Fig 5: Number of real images

The count of authentic (real) images is 7437 as shown in Fig 5, a substantially larger number compared to fake images, posing the challenge of an imbalanced dataset. To address this issue, a technique known as under sampling will be employed, reducing the number of pristine images. As a result, an almost balanced dataset of images is achieved belonging to both the fake and real classes, totaling half of the original dataset.

3.4.1.3 DATA PREPROCESSING

The approach involves conducting an error level analysis on the input image and feeding the resulting Error Level Analysis (ELA) to a simple convolutional neural network (CNN) trained to discern between fake and authentic images. Error level analysis is a method for detecting manipulated images by capturing them at a specific quality level and subsequently calculating the variation from the compression level. When an image is initially saved as a JPEG, it undergoes compression. Most editing software, such as Adobe Photoshop, GIMP, and Adobe Lightroom, supports JPEG compression operations. If the image is then modified using editing software, it is compressed once again. Neal Krawetz introduced the concept of Error Level Analysis for images after observing how errors propagate during the saving of a JPEG image,

especially when sections are cut out and the image is subsequently compressed. of an image and pasting it into another image, the ELA for the pasted section often detects a more significant error, which means it is brighter than the rest of the image higher ELA level.

- Function for generating ELA image of an input image.
- A function is defined that will generate an ELA image for an image plus it will normalize the pixel value of the ELA image (from 0–255 to 0–1). It will also resize the image to 128*128.
- It is then iterated over the dataset to generate X (input) and Y (output). X contains a 128*128 long vector and Y is the corresponding label for that X (0 for fake and 1 for real)
- Next, the data is separated (X and Y) into training and validation sets. A testing set will not be built as there are not enough data points available.
- A custom data generator is built so that there is no need to preprocess all the data points at a time and store them in the RAM. A batch of data points can be preprocessed and passed through Convolution Neural Network.

3.4.1.4 MODEL BUILDING AND TRAINING

- A simple neural network is made with 2 convolution layers with a max pooling layer and a dropout layer followed by two dense layers and one output layer. The model has a total of 4194320 parameters as shown in Fig 6.

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 128, 128, 3)]     0

 conv2d (Conv2D)             (None, 128, 128, 128)     3584

 conv2d_1 (Conv2D)           (None, 128, 128, 64)      73792

 max_pooling2d (MaxPooling2  (None, 64, 64, 64)        0
 D)

 dropout (Dropout)           (None, 64, 64, 64)        0

 flatten (Flatten)           (None, 262144)            0

 dense (Dense)               (None, 16)                4194320

 dense_1 (Dense)             (None, 8)                 136

 dense_2 (Dense)             (None, 1)                 9

...
Trainable params: 4271841 (16.30 MB)
Non-trainable params: 0 (0.00 Byte)
```

Fig 6: Number of Parameters

- The model is compiled with Adam (learning rate=0.0001) as optimizer and binary cross entropy as loss function because there are only two classes.
- Two callbacks are defined (A callback is a set of functions to be applied at given stages of the training procedure. It is defined and used when there is a need to automate some tasks after every training/epoch that helps you have control over the training process).

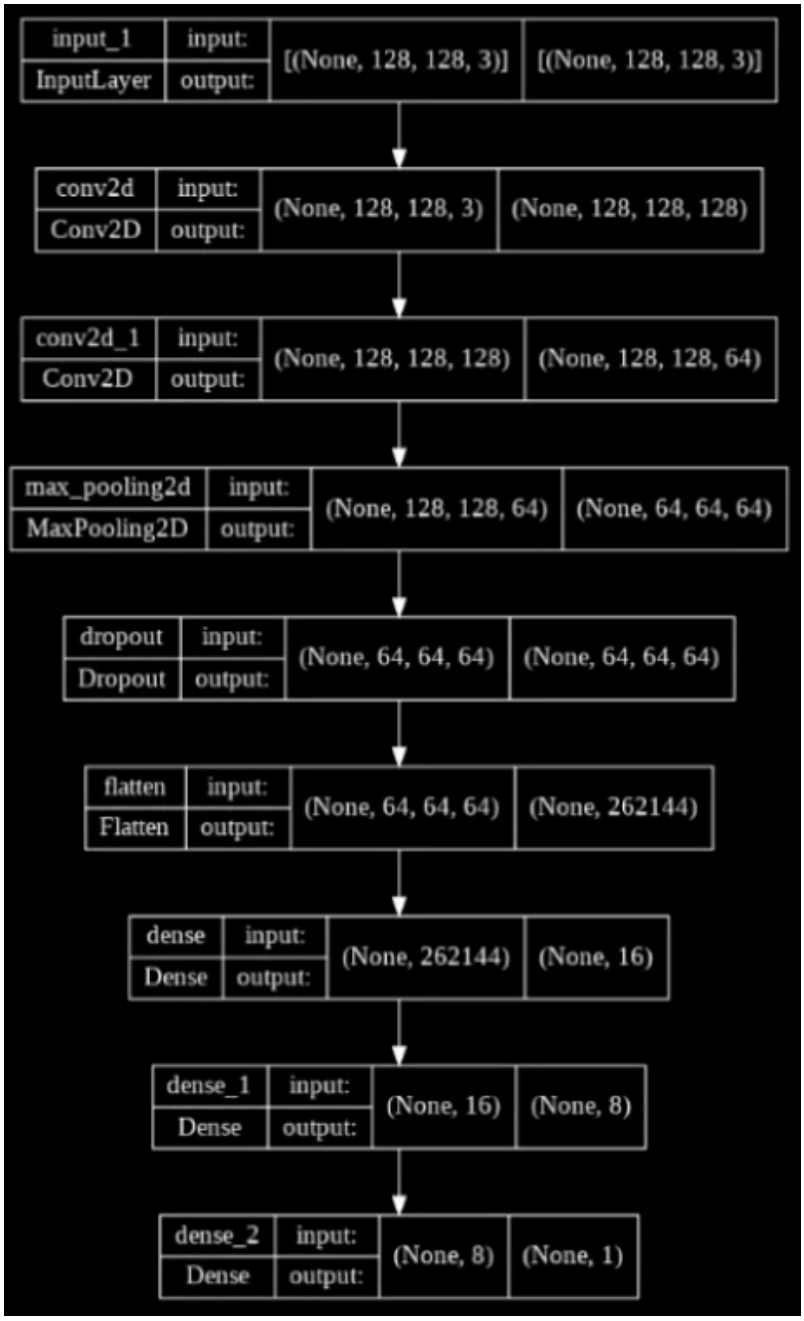


Fig 7: Structure of CNN model

- The Structure of the copy-move forgery detection CNN model is shown in Fig 7.
- LearningRateScheduler It will decrease the learning rate by 10 percent after every 5 epochs.
- Early stopping – It will monitor validation loss, if it doesn't decrease after 3 epochs it will stop the training model.

3.4.1.5 PHASE 2

In the second phase, the main objective is to predict the specific region of a manipulated image where tampering has occurred. This task essentially boils down to a binary image segmentation problem. The input to the model is a forged image, and the desired output is a binary image that highlights the areas where tampering has taken place.

3.4.1.6 DATA EXPLORATION

- Importing essential libraries for data manipulation and constructing a deep learning model.
- In this context, manipulated images will be exclusively utilized, and for these corresponding masks are provided. These masks delineate the specific regions within the images that have undergone tampering. So essentially, the process involves reading each file individually and extracting the image path, assigning a label (in case, "fake"), and creating an image ID.
- For every manipulated image, there is a corresponding mask image. Iterations will be run through the CASIA 2 Groundtruth folder in the dataset (Fig 8), which contains all the mask images. During this iteration, the image ID and the path of the respective mask image will be recorded

```
fake_image_mask={'image_id':[],'mask_image_path':[]}
fake_image_mask_path='./CASIA2/CASIA 2 Groundtruth'
for file in os.listdir(fake_image_mask_path):
    if file.endswith('jpg') or file.endswith('png') :
        temp_path=fake_image_mask_path+"/"+str(file)
        fake_image_mask['mask_image_path'].append(temp_path)
        fake_image_mask['image_id'].append(file[:-7])
fake_image_mask=pd.DataFrame(fake_image_mask)
fake_image_mask.head()
```

✓ 0.0s

	image_id	mask_image_path
0	Tp_D_CND_M_N_ani00018_sec00096_00138	./CASIA2/CASIA 2 Groundtruth/Tp_D_CND_M_N_ani0...
1	Tp_D_CND_M_N_art00076_art00077_10289	./CASIA2/CASIA 2 Groundtruth/Tp_D_CND_M_N_art0...
2	Tp_D_CND_M_N_art00077_art00076_10290	./CASIA2/CASIA 2 Groundtruth/Tp_D_CND_M_N_art0...
3	Tp_D_CND_S_N_ani00073_ani00068_00193	./CASIA2/CASIA 2 Groundtruth/Tp_D_CND_S_N_ani0...
4	Tp_D_CND_S_N_ind00078_ind00077_00476	./CASIA2/CASIA 2 Groundtruth/Tp_D_CND_S_N_ind0...

```
fake_image_data=fake_image_data.merge(fake_image_mask,on='image_id')
```

✓ 0.0s

Fig 8: Groundtruth folder containing mask Images

- Now there are two datasets, one contains a fake image path and image ID second contains a mask image path and image ID. So, the idea is to merge both of them (on image id) into one which contains the image path and mask image path.

```
fake_image_data.tail()
✓ 0.0s
```

	image_path	label	image_id	mask_image_path
1999	./CASIA2/Tp/Tp_S_NRN_S_O_ani10103_ani10103_106...	fake	Tp_S_NRN_S_O_ani10103_ani10103_10634	./CASIA2/CASIA 2 Groundtruth/Tp_S_NRN_S_O_ani1...
2000	./CASIA2/Tp/Tp_S_NRN_S_O_arc10129_arc10129_118...	fake	Tp_S_NRN_S_O_arc10129_arc10129_11895	./CASIA2/CASIA 2 Groundtruth/Tp_S_NRN_S_O_arc1...
2001	./CASIA2/Tp/Tp_S_NRN_S_O_cha00077_cha00077_110...	fake	Tp_S_NRN_S_O_cha00077_cha00077_11017	./CASIA2/CASIA 2 Groundtruth/Tp_S_NRN_S_O_cha0...
2002	./CASIA2/Tp/Tp_S_NRN_S_O_cha10126_cha10126_121...	fake	Tp_S_NRN_S_O_cha10126_cha10126_12153	./CASIA2/CASIA 2 Groundtruth/Tp_S_NRN_S_O_cha1...
2003	./CASIA2/Tp/Tp_S_NRN_S_O_cha10187_cha10187_123...	fake	Tp_S_NRN_S_O_cha10187_cha10187_12308	./CASIA2/CASIA 2 Groundtruth/Tp_S_NRN_S_O_cha1...

Fig 9: Two datasets of Fake and Real Images

The dataset contains fake image paths and mask image paths corresponding to each fake image, as illustrated in Fig 9. The total number of fake images in the dataset is 2004.

3.4.1.7 DATA PREPROCESSING

- In the image forgery detection process, a crucial step involves modifying the mask image to distinguish between tampered and non-tampered regions. This modification is achieved by representing the tampered region as black and the non-tampered region as white, as illustrated in Fig 10. Initially, the mask image contained pixel values ranging from 0 to 255. To ensure compatibility with the sigmoid activation function used in the output layer of the model, these values are scaled down to either 0 or 1. Specifically, all pixel values that were originally 255 (representing white) are converted to 0.0 (black), while pixel values that are not equal to 255 are converted to 1.0 (white). This transformation allows the model to interpret the mask image accurately and make informed decisions regarding the authenticity of the input image.

- The scaling of pixel values in the mask image is critical for the proper functioning of the model's output layer. The sigmoid activation function, commonly used in binary classification tasks, expects input values to be within a certain range (typically between 0

and 1). By converting the pixel values in the mask image to 0 or 1, the sigmoid function can effectively process this data and generate meaningful predictions. This step ensures that the model can accurately differentiate between tampered and non-tampered regions based on the information provided by the mask image, ultimately improving the overall performance and reliability of the forgery detection system.

```
fig=plt.figure(figsize=(15,15))
ax1=fig.add_subplot(131)
ax2=fig.add_subplot(132)
temp_mask_arr_mod=change_pixel_value(temp_mask_arr)
ax1.imshow(temp_arr)
ax2.imshow(temp_mask_arr_mod,cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f0ad81b7690>
```

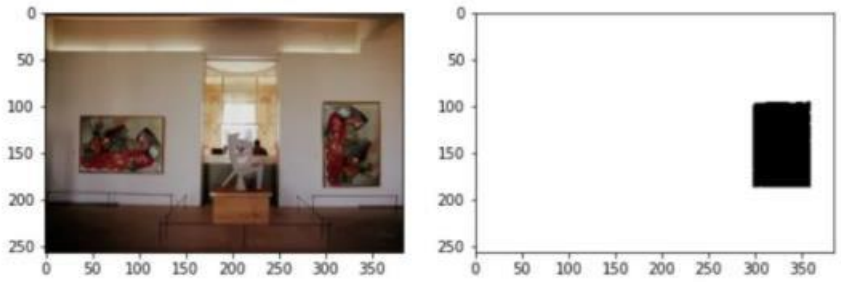


Fig 10: Tampered Region

- The Dataset (fake image data) is then split into training and validation datasets.
- For this task, a dual-stream UNET will be used. The first stream input will be the RGB image second stream input will be a noise feature map which will be obtained by passing the image through SRM (Steganalysis Rich Model) filters.
- The Precision, Recall and F1-Score can be calculated from the code shown in Fig 11

```python
from sklearn.metrics import precision_score, recall_score, f1_score
from sklearn.metrics import confusion_matrix

# Function to convert probability to binary prediction
def binarize_prediction(prediction, threshold=0.5):
    return 1 if prediction >= threshold else 0

# Load your dataset and ground truth labels
# Assuming 'ground_truth_labels' is a list of true labels (1 for forged, 0 for pristine)
# Assuming 'predictions' is a list of model predictions (probabilities between 0 and 1)

# Example (replace with your actual data)
ground_truth_labels = [1, 0, 0, 1, 1]
predictions = [0.8, 0.5, 0.1, 0.2, 0.6]

# Binarize predictions based on a threshold (default threshold is 0.5)
binary_predictions = [binarize_prediction(pred) for pred in predictions]

# Calculate metrics
precision = precision_score(ground_truth_labels, binary_predictions)
recall = recall_score(ground_truth_labels, binary_predictions)
f1 = f1_score(ground_truth_labels, binary_predictions)

# Print the results
print("Precision:", precision)
print("Recall", recall)
print("f1",f1)

# Confusion Matrix
conf_matrix = confusion_matrix(ground_truth_labels, binary_predictions)
print("Confusion Matrix:")
pr
print(conf_matrix)
```

✓ 0.0s

Fig 11: Code to calculate Precision, Recall & F1 Score

3.4.2 SPLICING DETECTION MODEL

- The ELA Image Generation Function is a code segment that defines a function named 'convert_to_ela_image'. This function takes two inputs: the path to an image file and the quality level for resaving the image. The function then executes a series of operations to create an ELA (Error Level Analysis) image based on the input image. ELA images are useful in detecting image manipulations because they highlight areas of an image that may have been altered or compressed differently from the rest of the image. By resaving the image at a specified quality level and comparing it to the original, ELA images can reveal discrepancies that indicate tampering or manipulation.
- In the data preparation step, the input images are converted into ELA images using the 'convert_to_ela_image' function. These ELA images are then resized to a standard size

of 128 x 128 pixels. This resizing is crucial for standardizing the input images, ensuring that they are all the same size and format before being processed by the splicing detection model. Standardizing the input images in this way helps to improve the efficiency and accuracy of the model, as it reduces the variability in the input data and ensures that the model is trained on a consistent set of images. The original image and the image after conversion to ELA is shown in Fig 12.

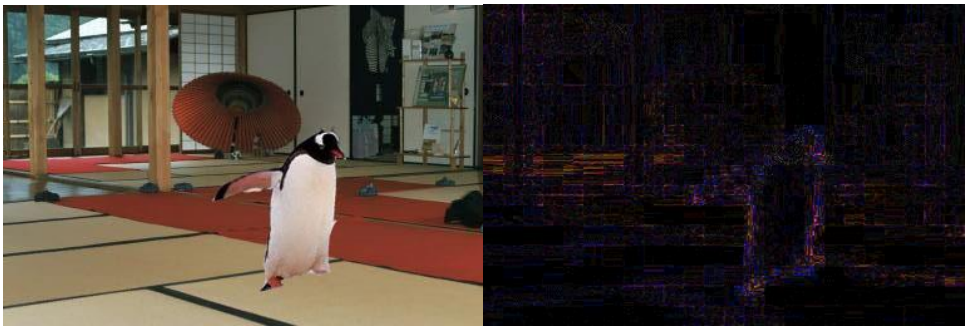


Fig 12: a) Original image b) Image after conversion to ELA

3.5 KEY CHALLENGES

- Since, it is a CNN it requires a heavy computational power to train the model and with the use of more and more big dataset consisting of more images there is a need for a heavy GPU enabled machine to train it efficiently and faster.
- Ensuring that the models generalize well to unseen data and can detect a wide range of image manipulations effectively can be challenging.
- Detecting complex manipulations such as splicing and retouching can difficult to identify.
- Handling noise and distortion in images can affect the model's ability to detect forgeries accurately.

CHAPTER 4: TESTING

Following the successful training of the models, the testing and validation phase will begin by evaluating the models' performance on randomly selected photos from the dataset.

4.1 TESTING STRATEGY

Testing strategy of both the models are described below:

4.1.1 COPY-MOVE FORGERY DETECTION MODEL

```
def convert_to_ela_image(path, quality):
    temp_filename = 'temp_file.jpg'
    ela_filename = 'temp_ela_file.png'

    image = Image.open(path).convert('RGB')
    image.save(temp_filename, 'JPEG', quality = quality)
    temp_image = Image.open(temp_filename)

    ela_image = ImageChops.difference(image, temp_image)

    extrema = ela_image.getextrema()
    max_diff = max([ex[1] for ex in extrema])
    if max_diff == 0:
        max_diff = 1
    scale = 255.0 / max_diff

    ela_image = ImageEnhance.Brightness(ela_image).enhance(scale)

    return ela_image
```

Fig 13: Function to convert image to ELA image

In this code snippet (Fig 13), a function "convert_to_ela_image" is defined which applies Error Level Analysis (ELA) to an input image. ELA identifies portions of a picture that may have

been digitally manipulated by comparing the original image to a re-saved version with a different compression level. The function accepts an image file path and a quality argument as input. It creates a temporary JPEG version of the image and then calculates the difference between the original and temporary images to identify areas of substantial variation. Finally, it increases the brightness of the difference image to highlight the altered regions before returning the final ELA image.

```
image_size = (128, 128)

def prepare_image(image_path):
    return np.array(convert_to_ela_image(image_path, 85).resize(image_size)).flatten() / 255.0
```

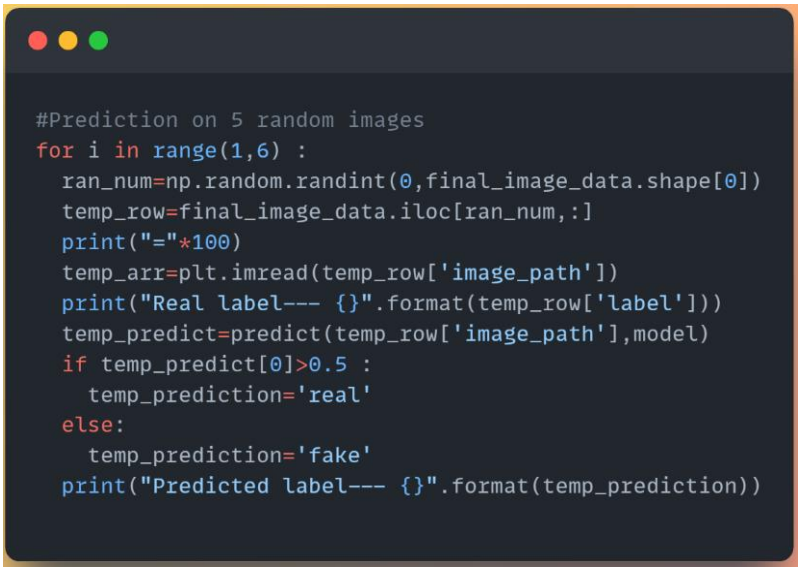
Fig 14: Function to prepare test image

This code sample (Fig 14) provides the function “prepare_image”, which turns an input image to an Error Level Analysis (ELA) image, resizes it to 128x128 pixels, flattens it into a NumPy array, and normalizes the pixel values to a range of 0 to 1.

```
def predict(img_path,model) :
    pi=prepare_image(img_path)
    pi=pi.reshape(1,128,128,3)
    predict=model.predict(pi)
    return predict
```

Fig 15: Function to get prediction for test image

In this snippet (Fig 15), the test image is first prepared by converting into ELA image and getting resized, after that getting the prediction for the test image using the trained model.



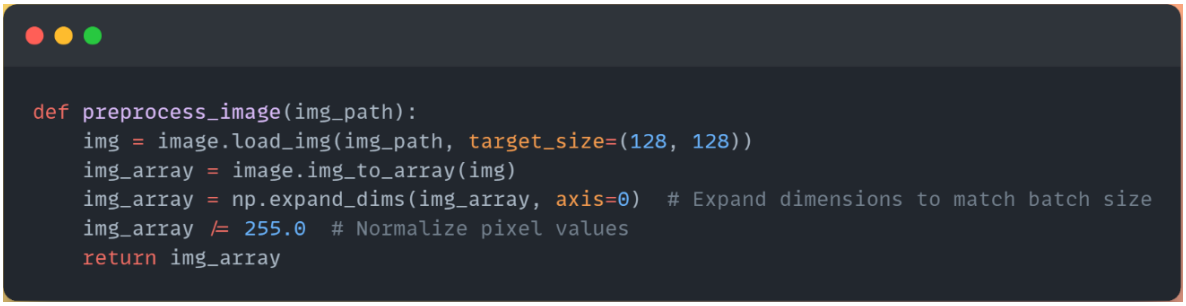
```
#Prediction on 5 random images
for i in range(1,6) :
  ran_num=np.random.randint(0,final_image_data.shape[0])
  temp_row=final_image_data.iloc[ran_num,:]
  print("="*100)
  temp_arr=plt.imread(temp_row['image_path'])
  print("Real label--- {}".format(temp_row['label']))
  temp_predict=predict(temp_row['image_path'],model)
  if temp_predict[0]>0.5 :
    temp_prediction='real'
  else:
    temp_prediction='fake'
  print("Predicted label--- {}".format(temp_prediction))
```

Fig 16: Prediction on 5 random images

In this snippet (Fig 16), 5 random images are selected from the dataset and are sent to predict function to get the prediction of whether they are fake or authentic.

4.1.2 SPLICING FORGERY DETECTION MODEL

This function “preprocess_image” (Fig 18) loads an image from the specified location and resizes it to 128x128 pixels using Keras' image.load_img method. It then converts the image to a NumPy array, increases its dimensions to match the desired batch size (adding a dimension at axis 0), and divides the pixel values by 255.0 to normalize them to a range of 0 to 1. The processed picture array is returned for future usage in a model.



```
def preprocess_image(img_path):
    img = image.load_img(img_path, target_size=(128, 128))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)  # Expand dimensions to match batch size
    img_array /= 255.0  # Normalize pixel values
    return img_array
```

Fig 17: Function to preprocess the test image

```python
from keras.models import load_model
from keras.preprocessing import image
import numpy as np

# Load the saved model
model = load_model('my_model.h5')

# Path to your test image
test_image_path = 'CASIA1/Sp/Sp_D_CNN_A_ani0053_ani0054_0267.jpg'

# Preprocess the test image
test_image = preprocess_image(test_image_path)

# Get predictions for the test image
def interpret_predictions(predictions):
    if predictions[0][0] > 0.5:
        return "Real"
    else:
        return "Fake"

# Get predictions for the test image
predictions = model.predict(test_image)

# Interpret the predictions
prediction_label = interpret_predictions(predictions)

# Print the interpretation
print("Prediction:", prediction_label)
```

Fig 18: To test a custom image

A custom input image is used in Fig 18, which is then fed to the trained model to predict whether it is fake or authentic.

4.2 TEST CASES AND OUTCOMES

While classifying for the 5 random images to test the model trained for copy-move forgery detection, it classified all the 5 images correctly. As shown in Fig 19.

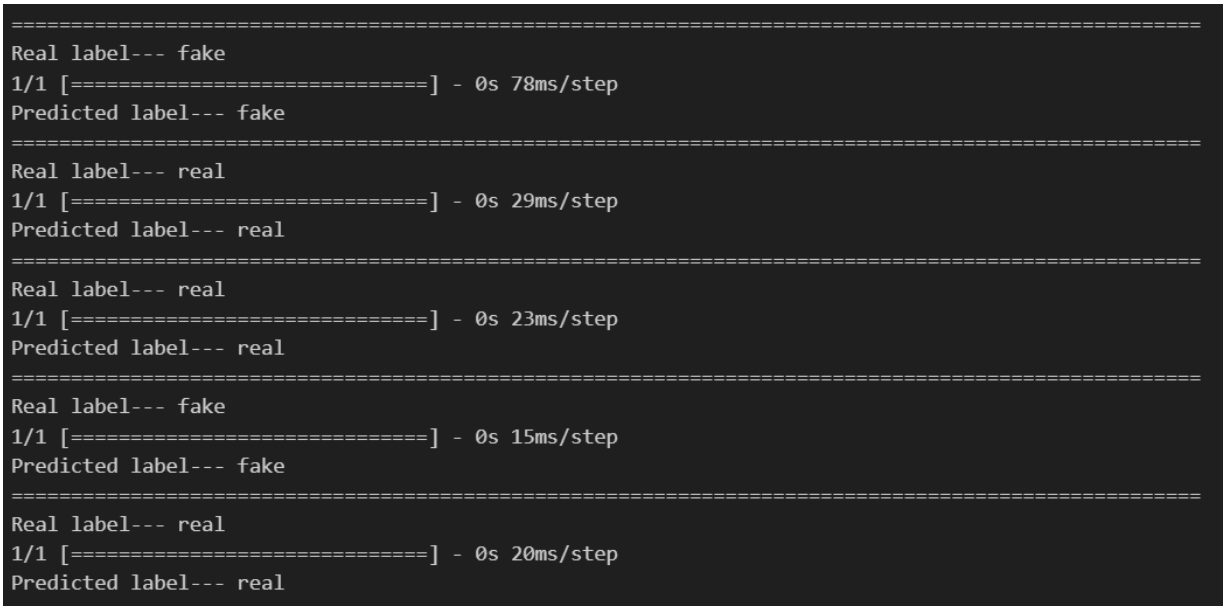


Fig 19: Prediction for copy-move forgery detection

While classifying for the 1 custom image to test the model trained for splicing forgery detection, it classified the custom image correctly. As shown in Fig 20.

Fig 20: Prediction for splicing detection

CHAPTER 5: RESULTS AND EVALUATION

5.1 RESULTS

The combination of both models produced an image forgery detection system capable of identifying two different types of image modifications with high accuracy. The splicing forgery detection model, which used a Dual-Stream UNet architecture with SRM filters, along with copy-move forgery detection model which used ELA-CNN will be validated after successful training of both the models in order to see how they are performing on the testing dataset.

5.1.1 COPY-MOVE FORGERY DETECTION MODEL

```
plt.figure(figsize=(10,10))
plt.plot(history.history['val_loss'],label='Validation loss')
plt.plot(history.history['loss'],label='Loss')
plt.title("Epoch v/s Loss Graph")
plt.xlabel("Epoch")
plt.ylabel("Loss")
plt.legend()
plt.show()
```

Fig 21: Plotting the model loss

In this snippet (Fig 21), Epoch v/s Loss graph is being plotted (Fig 22). Here, in this graph, it illustrates how the loss of the model changes over the epochs during the training process. The loss represents the difference between the predicted output of the model and the actual ground truth of truth labels. The y-axis having lesser values indicates better performance over the epochs.

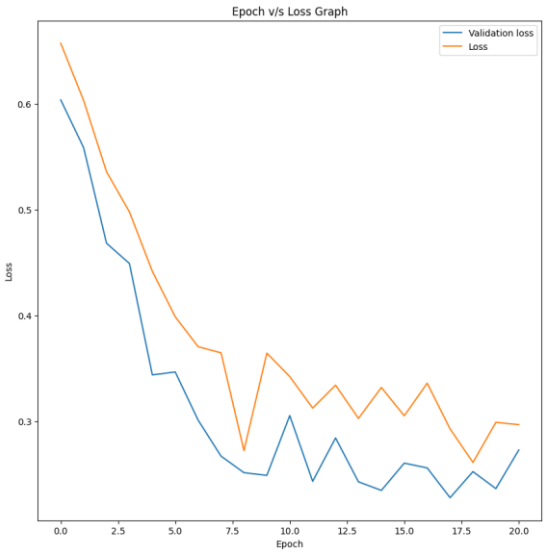


Fig 22: Epoch v/s Loss graph

5.1.2 SPLICING FORGERY DETECTION MODEL

```python
import matplotlib.pyplot as plt

# Plot the loss and accuracy curves for training and validation
fig, ax = plt.subplots(2, 1, figsize=(10, 8))
ax[0].plot(history.history['loss'], color='b', label="Training loss")
ax[0].plot(history.history['val_loss'], color='r', label="Validation loss")
ax[0].set_ylabel('Loss')
legend = ax[0].legend(loc='best', shadow=True)

ax[1].plot(history.history['accuracy'], color='b', label="Training accuracy")
ax[1].plot(history.history['val_accuracy'], color='r', label="Validation accuracy")
ax[1].set_ylabel('Accuracy')
ax[1].set_xlabel('Epoch')
legend = ax[1].legend(loc='best', shadow=True)

plt.show()
```

Fig 23: Function to plot Accuracy and Loss v/s Epoch

This code (Fig 23) employs Matplotlib to generate two subplots that display the training and validation loss (top plot) and training and validation accuracy (bottom plot) over epochs. It retrieves the loss and accuracy history from a a trained model and plots it with blue lines for

training and red lines for validation. The graphs as shown in Fig 24 helps visualize the model's performance during training by demonstrating how loss drops and accuracy improves over epochs.

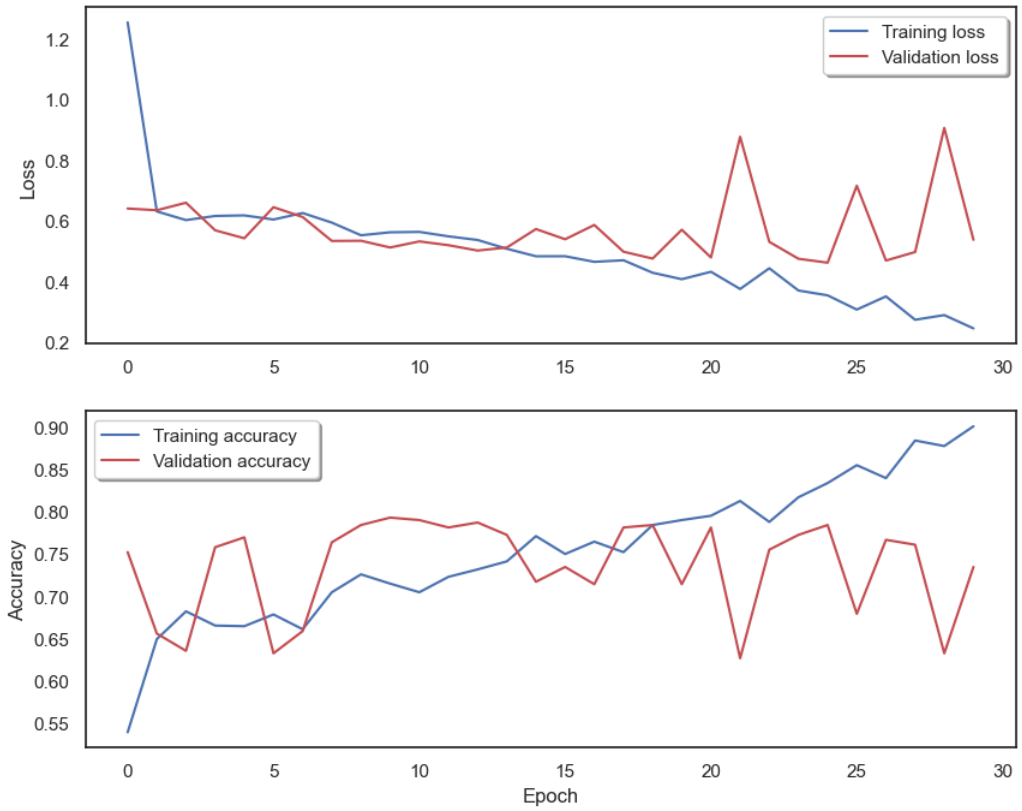


Fig 24: Accuracy and Loss v/s Epoch graphs

- Training and validation loss and accuracy graphs: Continual decrease in training loss and initial decrease of validation loss but then gradual increase is a sign of overfitting.
- The validation accuracy starts to decrease while the training accuracy continues to increase which is also a sign of overfitting.

The “plot_confusion_matrix” function (Fig 25) generates a visual representation of a confusion matrix. It employs matplotlib to create a color-coded matrix, with each cell representing the number of true positives, false positives, true negatives, and false negatives. This function accepts the confusion matrix cm and a list of classes as input, together with optional parameters for normalization, title, and colormap. It then presents the confusion matrix, which includes

labels for true and predicted labels, making it easier to interpret the performance of a classification model.

```
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]

    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, cm[i, j],
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

Y_pred = model.predict(X_val)
Y_pred_classes = np.argmax(Y_pred,axis = 1)
Y_true = np.argmax(Y_val,axis = 1)
confusion_mtx = confusion_matrix(Y_true, Y_pred_classes)
plot_confusion_matrix(confusion_mtx, classes = range(2))
```

Fig 25: Function to plot confusion matrix

The confusion matrix (Fig 26) generated from the "plot_confusion_matrix" function is explained below:

- True Positive (TP): These are the cases where the model predicted the class as positive (P), and the actual class is also positive. In this case, TP = 92, which means there were 92 instances where the model correctly predicted the positive class.
- True Negative (TN): These are the cases where the model predicted the class as negative (N), and the actual class is also negative. TN = 166, indicating that there were 166 instances where the model correctly predicted the negative class.

- False Positive (FP): These are the cases where the model predicted the class as positive (P), but the actual class is negative (N). FP = 56, meaning there were 56 instances where the model incorrectly predicted the positive class.
- False Negative (FN): These are the cases where the model predicted the class as negative (N), but the actual class is positive (P). FN = 30, indicating that there were 30 instances where the model incorrectly predicted the negative class.

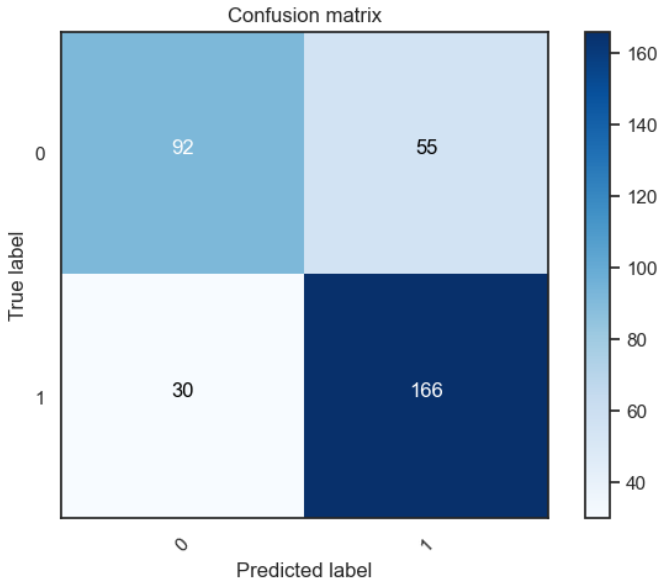


Fig 26: Confusion matrix

```python
from sklearn.metrics import precision_score, recall_score, f1_score

confusion_mtx = confusion_matrix(Y_true, Y_pred_classes)

precision = precision_score(Y_true, Y_pred_classes)
recall = recall_score(Y_true, Y_pred_classes)
f1 = f1_score(Y_true, Y_pred_classes)

print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)
```

Fig 27: Function to print Precision, Recall and F1 Score

In this snippet (Fig 27) the Precision, Recall and F1 Score (Fig 28) is being calculated and printed for the splicing model.

```
Precision: 0.751131221719457
Recall: 0.8469387755102041
F1 Score: 0.7961630695443644
```

Fig 28: Precision, Recall and F1 Score

5.1.3 WEB INTERFACE

After successfully training and testing both the models, the team implemented the web interface and easily integrated the models within its framework (Fig 29).

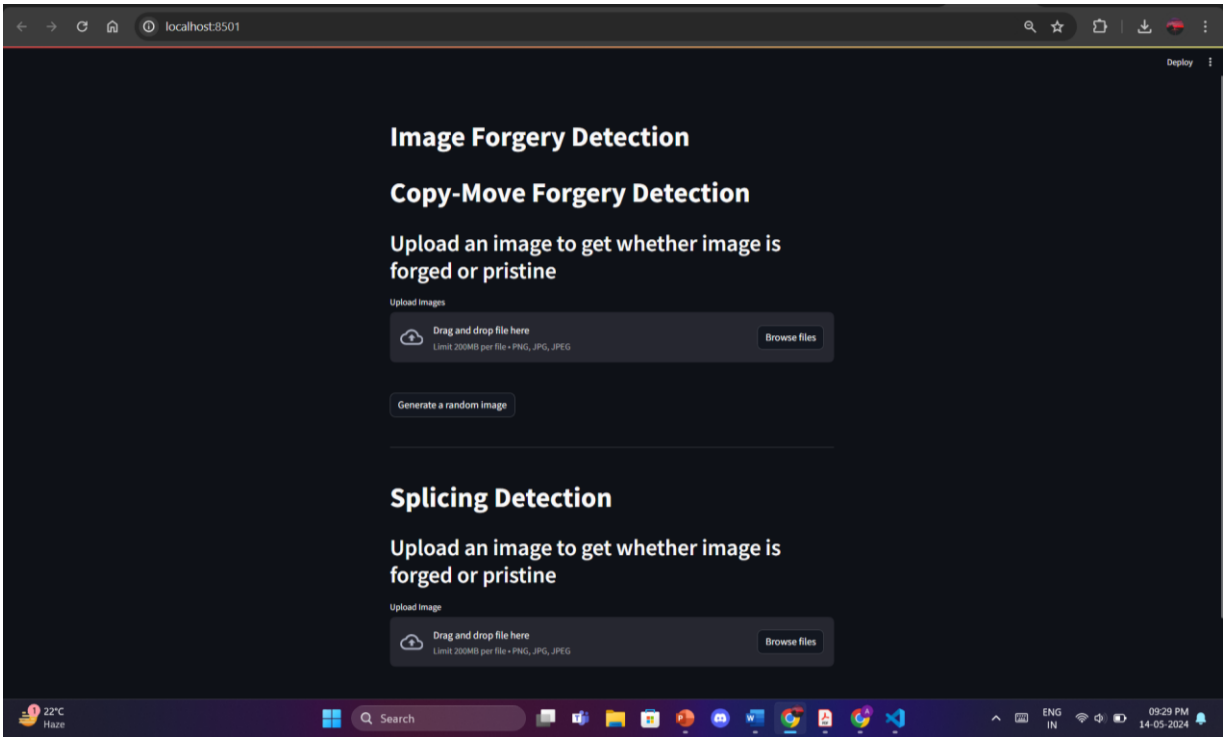


Fig 29: Home page of the web interface

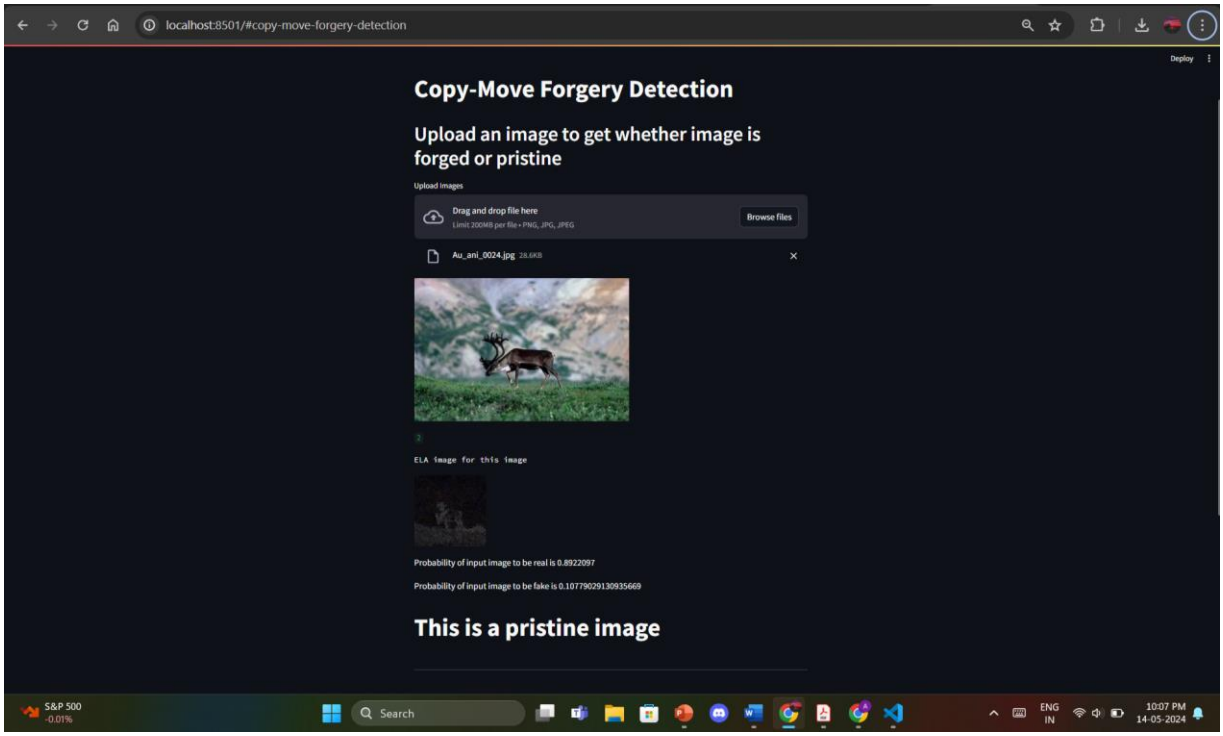


Fig 30: User-Interface for copy-move forgery detection

Here it is clear that the web interface is working properly for copy-move forgery detection as well as splicing forgery detection (Fig 30 & 31) and is able to efficiently generate the ELA image as well as display the probability of the image being real or fake.

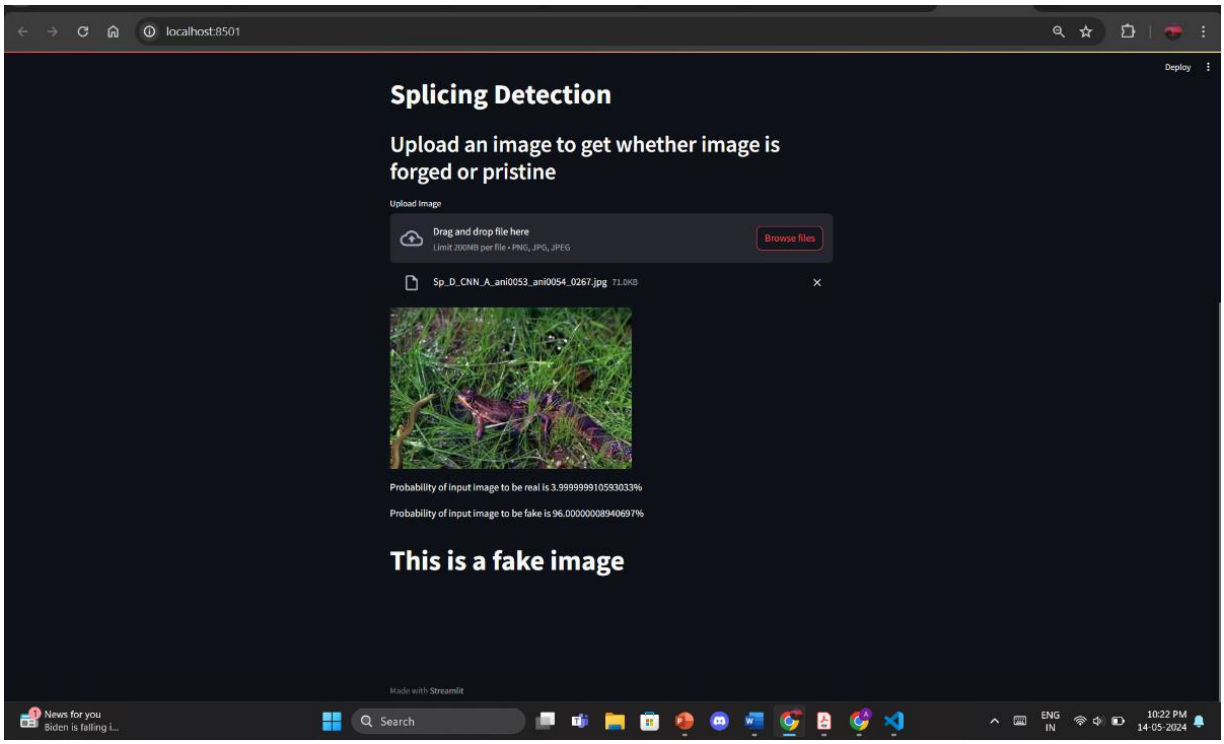


Fig 31: User-Interface for splicing forgery detection

CHAPTER 6: CONCLUSIONS AND FUTURE SCOPE

6.1 CONCLUSION

In this project, the two models utilized to detect weather the images have copy-move forgery or splicing forgery with the help of 2 separate CNN models. Commencing with the input image, the system generates an ELA image, highlighting regions of interest through a comparison with re-shaped version. Subsequently, features, encompassing texture and color information, are extracted from the ELA image. With a precision rate of 89% for the copy-move forgery and 75% for the splicing forgery detection the web interface is able to efficiently predicts both the type of forgery.

6.2 FUTURE SCOPE

- Dataset Expansion and Complexity: Enlarge the dataset with diverse forgery scenarios, resolutions, and compression levels to strengthen the model's capability to detect a broader range of manipulations.
- The model can also be upgraded to detect more types of image forgeries such as Image resampling and Image retouching.
- Interdisciplinary Collaborations: Collaborate with fields like cryptography and blockchain to explore image authentication and tamper-proofing solutions, enhancing image integrity and traceability.
- Advanced Machine Learning Integration: Incorporate advanced machine learning algorithms, like deep learning models, to enhance accuracy in detecting complex manipulations and improve adaptability.

REFERENCES

[1] https://www.kaggle.com/datasets/sophatvathana/casia-dataset

[2] N. P. Nethravathi, B. D. Austin, D. S. P. Reddy, G. V. N. S. P. Kumar, and G. K. Raju, "Image Forgery Detection Using Deep Neural Network," in Proc. 2023 6th Int. Conf. Intell. Comput. Control Syst. (ICICCS), 2023, pp. 216-221, doi: 10.1109/ICICCS54921.2023.9951952.

[3] M. Zanardelli, F. Guerrini, R. Leonardi, et al., "Image forgery detection: a survey of recent deep-learning approaches," Multimed Tools Appl, vol. 82, pp. 17521–17566, 2023.

[4] S. S. Ali, I. I. Ganapathi, N.-S. Vu, S. D. Ali, N. Saxena, and N. Werghi, "Image Forgery Detection Using Deep Learning by Recompressing Images," in Proc. 2019 IEEE Int. Conf. Image Process. (ICIP), Taipei, Taiwan, 2019, pp. 4046-4050, doi: 10.1109/ICIP.2019.8803393.

[5] J. Ega, D. S. S. Krishna, and V. M. Manikandan, "A Review on Digital Image Forgery Detection," in Proc. 2017 IEEE Int. Conf. Signal Process., Informatics, Commun. Energy Syst. (SPICES), Kozhikode, India, 2017, pp. 1-6, doi: 10.1109/SPICES.2017.8076270.

[6] N. K. Rathore, N. K. Jain, P. K. Shukla, U. S. Rawat, and R. Dubey, "Image Forgery Detection Using Singular Value Decomposition with Some Attacks," in Proc. 2018 5th Int. Conf. Signal Process., Comput. Control (ISPCC), 2018, pp. 41-46, doi: 10.1109/ISPCC.2018.8663238.

[7] A. Kuznetsov, "Digital image forgery detection using deep learning approach," J. Phys.: Conf. Ser., vol. 1368, no. 3, p. 032028, 2019, doi: 10.1088/1742-6596/1368/3/032028.

[8] A. Doegar, M. Dutta, and G. Kumar, "CNN based Image Forgery Detection using pre-trained AlexNet Model," in Proc. 2019 IEEE 8th Int. Conf. Commun. Electron. Syst. (ICCES), 2019, pp. 1555-1559, doi: 10.1109/ICCES46393.2019.8924430.

[9] D. Kim and H. Lee, "Image Manipulation Detection using Convolutional Neural Network," in Proc. 2016 IEEE Int. Workshop Inf. Forensics Security (WIFS), Abu Dhabi, United Arab Emirates, 2016, pp. 1-6.

[10] B. Bayar and M. C. Stamm, "A Deep Learning Approach to Universal Image Manipulation Detection Using a New Convolutional Layer," IEEE Trans. Inf. Forensics Security, vol. 13, no. 11, pp. 2479-2492, Nov. 2018, doi: 10.1109/TIFS.2018.2853711.

[11] C. D. Kaur and N. Kanwal, “An analysis of image forgery detection techniques,” Stat. Optim. Inf. Comput., vol. 7, no. 2, 2019, doi: 10.19139/soic.v7i2.542.

[12] P. Sharma, M. Kumar, and H. Sharma, "Comprehensive analyses of image forgery detection methods from traditional to deep learning approaches: an evaluation," Multimed Tools Appl, vol. 82, pp. 18117–18150, 2023, doi: 10.1007/s11042-022-13808-w.

[13] E. U. H. Qazi, T. Zia, and A. Almorjan, “Deep Learning-Based Digital Image Forgery Detection System,” Appl. Sci., vol. 12, no. 6, p. 2851, Mar. 2022, doi: 10.3390/app12062851.

[14] Sabeena and Dr. Lizy Abraham, "Digital image forgery detection approaches: A review and analysis," in Proc. 2nd Int. Conf. IoT, Social, Mobile, Analytics & Cloud Comput. Vis. Bio-Eng. (ISMAC-CVB 2020), 2020, pp. 327-335.

[15] S. alZahir and R. Hammad, "Image forgery detection using image similarity," Multimed Tools Appl., vol. 79, no. 1, pp. 28643-28659, 2020, doi: 10.1007/s11042-020-09502-4.

[16] S. Selvaraj and I. M. Ramya, "Image Forgery Detection Using Machine Learning," SSRN Working Paper No. 3950994, Oct. 2021.

[17] S. Bayram, I. Avcibaş, and B. Sankur, "Image manipulation detection," J. Electron. Imaging, vol. 15, no. 4, p. 041102, 2006, doi: 10.1117/1.2401138.

[18] B. Diallo, T. Urruty, P. Bourdon, and C. Fernandez-Maloigne, "Robust forgery detection for compressed images using CNN supervision," Forensic Sci. Int.: Reports, vol. 2, p. 100112, 2020, doi: 10.1016/j.fsir.2020.100112.

[19] Y. Abdalla, T. Iqbal, and M. Shehata, "Copy-move forgery detection and localization using a generative adversarial network and convolutional neural-network," Information, vol. 10, no. 9, p. 286, 2019, doi: 10.3390/info10090286.

[20] R. Agarwal and O. Verma, "An efficient copy move forgery detection using deep learning feature extraction and matching algorithm," Multimed Tools Appl., vol. 79, 2020, doi: 10.1007/s11042019-08495-z.

[21] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, and G. Serra, "A SIFT-based forensic method for copy-move attack detection and transformation recovery," IEEE Trans. Inf. Forensics Security, vol. 6, no. 3, pp. 1099-1110, Sep. 2011, doi: 10.1109/TIFS.2011.2129512.

[22] G. K. Birajdar and V. H. Mankar, "Digital image forgery detection using passive techniques: a survey," Digit. Investig., vol. 10, no. 3, pp. 226–245, 2013, doi: 10.1016/j.diin.2013.04.007.

[23] M. Elaskily, H. Elnemr, A. Sedik, M. Dessouky, G. El Banby, O. Elaskily, A. A. M. Khalaf, H. Aslan, O. Faragallah, and F. A. El-Samie, "A novel deep learning framework for copy-move forgery detection in images," Multimed Tools Appl., vol. 79, 2020, doi: 10.1007/s11042-020-08751-7.

[24] M. D. Ansari, S. P. Ghrera, and V. Tyagi, "Pixel-based image forgery detection: A review," IETE J. Educ., vol. 55, no. 1, pp. 40-46, 2014.

[25] T. Qazi, et al., "Survey on blind image forgery detection," IET Image Process., vol. 7, no. 7, pp. 660-670, 2013.

[26] N. B. Abd Warif, et al., "Copy-move forgery detection: survey, challenges and future directions," J. Netw. Comput. Appl., vol. 75, pp. 259-278, 2016.

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: …….

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report | Paper

Name: _____ __**Department:** _____ **Enrolment No** _____

Contact No. _____**E-mail.** _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

_____

_____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at ……..(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

<table>
<tr><th>Copy Received on</th><th>Excluded</th><th>Similarity Index (%)</th><th colspan="2">Generated Plagiarism Report Details (Title, Abstract & Chapters)</th></tr>
<tr><td rowspan="2"></td><td rowspan="4">• All Preliminary Pages
• Bibliography/Images/Quotes
• 14 Words String</td><td rowspan="2"></td><td>Word Counts</td><td></td></tr>
<tr><td rowspan="2">Report Generated on</td><td>Character Counts</td><td></td></tr>
<tr><td rowspan="2"></td><td>Submission ID</td><td>Total Pages Scanned</td><td></td></tr>
<tr><td></td><td>File Size</td><td></td></tr>
</table>

Checked by
Name & Signature **Librarian**

……

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

image forgery detection

ORIGINALITY REPORT

8%	4%	6%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Satyendra Singh, Rajesh Kumar. "Image forgery detection: comprehensive review of digital forensics approaches", Journal of Computational Social Science, 2024 Publication	1%
2	"Proceedings of Third International Conference on Sustainable Expert Systems", Springer Science and Business Media LLC, 2023 Publication	1%
3	core.ac.uk Internet Source	1%
4	"Medical Image Computing and Computer Assisted Intervention – MICCAI 2019", Springer Science and Business Media LLC, 2019 Publication	1%
5	Munera A. Jabaar, Saad N. Alsaad. "Detection of Spliced Images in Social Media Application", 2021 7th International	<1%