## A

**MINI PROJECT REPORT**

**On**

# IMAGE FORGERY DETECTION BASED ON FUSION

# OF LIGHTWEIGHT DEEP LEARNING MODELS

*Submitted by,*

T.SAI PRANEESHA 21J41A6662

K. SHRAVYA 22J45A6605

A. GANGA VAISHNAVI 21J41A6606

A. NAGA LAKSHMI 21J41A6604

*In partial fulfilment of the requirements for the award of the degree of*

## BACHELOR OF TECHNOLOGY

in

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING IN ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING.

Under the Guidance of

### Dr. K. SATISH

Associate Professor, Computer Science and Engineering



# Department of computer science and engineering

## MALLA REDDY ENGINEERING COLLEGE

(An UGC Autonomous Institution, Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad) Maisammaguda , Secunderabad, Telangana, India 500100

#### NOVEMBER – 2024



**BONAFIDE CERTIFICATE**

This is to certify that this mini project work entitled “**IMAGE FORGERY DETECTION BASED ON FUSION OF LIGHT WEIGHT DEEP LEARNING MODELS**”, submitted by **TUNGENA SAI PRANEESHA (21J41A6662), KUCHANA SHARVYA (22J45A6605), ANNAPUREDDY NAGALAKSHMI (21J41A6604)**

**ATHINARAM GANGA VAISHNAVI (21J41A6606)** to Malla Reddy Engineering College affiliated to JNTUH, Hyderabad in partial fulfilment for the award of **Bachelor of Technology** in **COMPUTER SCIENCE AND ENGINEERING** is a bonafide record of project work carried out under our supervision during the academic year 2024 – 2025 and that this work has not been submitted elsewhere for a degree.

#### SIGNATURE

**B. SRINIVAS**

**Associate Professor**

**SUPERVISOR,**

CSE-AIML

Malla Reddy Engineering College Secunderabad, 500 100

#### SIGNATURE

#### DR. MOHANSRINIVAS

**HOD and Professor CSE-AIML**

Malla Reddy Engineering College Secunderabad , 500 100

**Submitted for Mini Project viva-voce examination held on**

**INTERNAL EXAMINER EXTERNAL EXAMINER**

# MALLA REDDY ENGINEERING COLLEGE

Maisammaguda, Secunderabad, Telangana, India 500100

## DECLARATION

I hereby declare that the project titled “**IMAGE FORGERY DETECTION BASED ON FUSION OF LIGHT WEIGHT DEEP LEARNING MODELS**”, submitted to Malla Reddy Engineering College (Autonomous) and affiliated with JNTUH, Hyderabad, in partial fulfillment of the requirements for the award of a **Bachelor of Technology** in **COMPUTER SCIENCE AND ENGINEERING(AIML)**, represents my ideas in my own words. Wherever others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity, and I have not misrepresented, fabricated, or falsified any idea, data, fact, or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute. It is further declared that the project report or any part thereof has not been previously submitted to any University or Institute for the award of degree.

Signature(s)

T.SAI PRANEESHA 21J41A6662

K. SHRAVYA 22J45A6605

1. NAGA LAKSHMI 21J41A6604
2. GANGA VAISHNAVI 21J41A6606

Secunderabad - 500 100

Date:

# MALLA REDDY ENGINEERING COLLEGE

Maisammaguda, Secunderabad, Telangana, India 500100

**ACKNOWLEDGEMENT**

We would like to express our sincere thanks to **Dr. A. Ramaswami Reddy, professor, Principal, MREC(A)** for providing the working facilities in the college.

Our sincere thanks and gratitude to **DR. U. MOHAN SRINIVAS, Professor & Head, Department of CSE-AIML, MREC(A)** for all the timely support and valuable suggestions during the period of our project.

We are thankful to our Project Coordinator **Dr B. SRINIVAS, Associate Professor**, Department of Computer Science and Engineering in AIML, for his cooperation during the project work.

We are greatly thankful and indebted to our internal guide,

**Dr. K. SATHISH, Professor, Department of CSE-AIML, MREC (A)** for her constant guidance, encouragement and moral support throughout the project.

Finally, we would also like to thank all the faculty and staff of the CSE-AIML Department who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

T.SAI PRANEESHA 21J41A6662

K. SHRAVYA 22J45A6605

A. NAGALAKSHMI 21J41A6604

A. GANGA VAISHNAVI 21J41A6606

**ABSTRACT**

## ABSTRACT:

Capturing images has been increasingly popular in recent years, owing to the widespread availability of cameras. Images are essential in our daily lives because they contain a wealth of information, and it is often required to enhance images to obtain additional information. A variety of tools are available to improve image quality; nevertheless, they are also frequently used to falsify images, resulting in the spread of misinformation. This increases the severity and frequency of image forgeries, which is now a major source of concern. Numerous traditional techniques have been developed over time to detect image forgeries. In recent years, convolutional neural networks (CNNs) have received much attention, and CNN has also influenced the field of image forgery detection. However, most image forgery techniques based on CNN that exist in the literature are limited to detecting a specific type of forgery (either image splicing or copy-move). As a result, a technique capable of efficiently and accurately detecting the presence of unseen forgeries in an image is required. In this paper, we introduce a robust deep learning based system for identifying image forgeries in the context of double image compression. The difference between an image’s original and recompressed versions is used to train our model. The proposed model is lightweight, and its performance demonstrates that it is faster than state-of-the-art approaches. The experiment results are encouraging, with an overall validation accuracy of 92.23%.

**KEY WORDS:** Accuracy Comparison Graph, Fine Tuned features with SVM

## TABLE OF CONTENTS

**ABSTRACT** v

[LIST OF FIGURES vii](#_TOC_250029)

[LIST OF SYMBOLS AND ABBREVIATIONS viii](#_TOC_250028)

CHAPTER DESCRIPTION PAGE NO.

1. INTRODUCTION 1-2
   1. [INTRODUCTION 1](#_TOC_250027)
   2. LITERATURE REVIEW 2
2. LITERATURE SURVEY 3-4
3. SYSTEM ANALYSIS 5-6
   1. [EXISTING SYSTEM 5](#_TOC_250026)
   2. DISADVANTAGES OF EXISTING SYSTEM 5
   3. PROPOSED SYSTEM 6
   4. ADVANTAGES OF THE PROPOSED SYSTEM 6
4. SYSTEM STUDY 7-8
   1. [FEASIBILITY STUDY 7](#_TOC_250025)
      1. [ECONOMIC FEASIBILITY 7](#_TOC_250024)
      2. [TECHNICAL FEASIBILITY 7](#_TOC_250023)
      3. OPERATIONAL FEASIBILITY 8
5. SYSTEM REQUIREMENTS 9
   1. [HARDWARE REQUIREMENTS 9](#_TOC_250022)
   2. SOFTWARE REQUIREMENTS 9
6. SYSTEM DESIGN 10-17
   1. [MODULES 10-11](#_TOC_250021)
   2. ARCHITECTURE DIAGRAM 12
   3. [UML DIAGRAMS 12-13](#_TOC_250020)
      1. [USE CASE DIAGRAM 14](#_TOC_250019)
      2. [CLASS DIAGRAM 15](#_TOC_250018)
      3. [SEQUENCE DIAGRAM 16](#_TOC_250017)
      4. [COLLABORATION DIAGRAM 17](#_TOC_250016)
7. IMPLEMENTATION 18-31
   1. SOFTWARE DESCRIPTION 18-28
      1. PYTHON 19
   2. [SOURCE CODE 29-31](#_TOC_250015)
8. SCREENSHOTS 33-40
   1. [HOME PAGE 33](#_TOC_250014)
      1. UPLOAD DATASET
   2. [PREPROCESS DATASETS 34-35](#_TOC_250013)
   3. [DATA SET 36](#_TOC_250012)
      1. [PROGRAM CODE 37](#_TOC_250011)
      2. [FINE TUNED FEATURES WITH SVM 38](#_TOC_250010)
      3. ML BASED ALGORITHM RESULTS 39
      4. BARCHART 39
      5. RESULT 40
9. SYSTEM TESTING 41-43
   1. [TESTING METHODOLOGIES 41](#_TOC_250009)
      1. [UNIT TESTING 41](#_TOC_250008)
      2. [INTEGRATION TESTING 41](#_TOC_250007)
      3. [FUNCTIONAL TESTING 42](#_TOC_250006)
      4. [SYSTEM TESTING 42](#_TOC_250005)
      5. [WHITE BOX TESTING 42](#_TOC_250004)
      6. [BLACK BOX TESTING 42](#_TOC_250003)
   2. [USER TRAINING 43](#_TOC_250002)
   3. [TESTING STRATEGY 43](#_TOC_250001)
10. CONCLUSION AND FUTURE SCOPE 44-45
    1. CONCLUSIONS 44
    2. [FUTURE SCOPE 45](#_TOC_250000)

**REFERENCES** 46-47

## LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **TITLE** | **PAGE NO.** |
| 3.2 | Convention Neural Network Diagram | 12 |
| 6.2  6.3.1 | System Architecture  Use Case Diagram | 14 |
| 6.3.2 | Class Diagram | 15 |
| 6.3.3 | Sequence Diagram | 16 |
| 6.3.4 | Collaboration Diagram | 17 |
| 8.1 | Image forgery detection Home page | 33 |
| 8.1.1 | Upload Dataset | 33 |
| 8.2 | Images for uploading | 34 |
| 8.3.1 | Program Code | 37 |
| 8.3.2 | Fine Tuned Features with SVM | 38 |
| 8.3.3 | ML Based Algorithm Result | 39 |
| 8.3.4 | Bar Chart | 39 |
| 8.3.5 | Result | 40 |

## LIST OF SYMBOLS AND ABBREVIATIONS

CNN - CONVOLUTIONAL NEURAL NETWORK

SVM - SUPPORT VECTOR MACHINE

RF KNN LR

* RANDOM FOREST
* K-NEAREST NEIGHBOURS
* LINEAR REGRESSION

DT - DECISION TREE

GAN - GENERATIVE ADVERSIAL NETWORKS

ELA - ERROR LEVEL ANALYSIS

**CHAPTER 1**

**INTRODUCTION**

## INTRODUCTION

Due to technological advancements and globalization, electronic equipment is now widely and inexpensively available. As a result, digital cameras have grown in popularity. There are many camera sensors all around us, and we use them to collect a lot of images. Images are required in the form of a soft copy for various documents that must be filed online, and a large number of images are shared on social media every day. The amazing thing about images is that even illiterate people can look at them and extract information from them. As a result, images are an integral component of the digital world, and they play an essential role in storing and distributing data. There are numerous tools accessible for quickly editing the images [1,2]. These tools were created with the intention of enhancing and improving the images. However, rather than enhancing the image, some people exploit their capabilities to falsify images and propagate falsehoods [3,4]. This is a significant threat, as the damage caused by faked images is not only severe, but also frequently irreverent.

There are two basic types of image forgery: image splicing and copy-move, which are discussed below:

• Image Splicing: A portion of a donor image is copied into a source image. A sequence of donor images can likewise be used to build the final forged image.

• Copy-Move: This scenario contains a single image. Within the image, a portion of the image is copied and pasted. This is frequently used to conceal other objects. The final forged image contains no components from other images.

2)The primary purpose in both cases of image forgery is to spread misinformation by changing the original content in an image with something else [5,6]. Earlier images were an extremely credible source for the information exchange, however, due to image forgery, they are used to spread misinformation. This is affecting the trust of the public in images, as the forging of images may or may not be visible or recognizable to the naked eye. As a result, it is essential to detect image forgeries to prevent the spread of misinformation as well as to restore public trust in images. This can be done by exploring the various artifacts left behind when an image forgery is performed, and they can be identified using various image processing techniques.

Researchers have proposed a variety of methods for detecting the presence of image forgeries [7–9]. Conventional image forgery detection techniques detect forgeries by concentrating on the multiple artifacts present in a forged image, such as changes in illumination, contrast, compression, sensor noise, and shadow. CNN’s have gained popularity in recent years for various computer vision tasks, including image object recognition, semantic segmentation, and image classification. Two major features contribute to CNN’s success in computer vision. Firstly, CNN takes advantage of the significant correlation between adjacent pixels. As a result, CNN prefers locally grouped connections over one-to-one connections between all pixel. Second, each output feature map is produced through a convolution operation by sharing weights. Moreover, compared to the traditional method that depends on engineered features to detect specific forgery, CNN uses learned features from training images, and it can generalize itself to detect unseen forgery. These advantages of CNN make it a promising tool for detecting the presence of forgery in an image. It is possible to train a CNN-based model to learn the many artifacts found in a forged image [10–13]. Thus, we propose a very light CNN-based network, with the primary goal of learning the artifacts that occur in a tampered image as a result of differences in the features of the original image and the tampered region.

The major contribution of the proposed technique are as follows:

• A lightweight CNN-based architecture is designed to detect image forgery efficiently. The proposed technique explores numerous artifacts left behind in the image tampering process, and it takes advantage of differences in image sources through image recompression.

• While most existing algorithms are designed to detect only one type of forgery, our technique can detect both image splicing and copy-move forgeries and has achieved high accuracy in image forgery detection. • Compared to existing techniques, the proposed technique is fast and can detect the presence of image forgery in significantly less time. Its accuracy and speed make it suitable for real-world application, as it can function well even on slower devices.

The rest of the paper is organized as follows. Section 2 provides a literature review of image forgery detection methodologies. Section 3 introduces the proposed framework for detecting the presence of forgeries in an image. Section 4 contains a discussion of the experimentation and the results achieved. Finally, in Section 5, we summarize the conclusions.

## 1.2. LITERATURE REVIEW:

Various approaches have been proposed in the literature to deal with image forgery. The majority of traditional techniques are based on particular artifacts left by image forgery, whereas recently techniques based on CNNs and deep learning were introduced, which are mentioned below. First, we will mention the various traditional techniques and then move on to deep learningbased techniques.

In [14], the authors’ proposed error level analysis (ELA) for the detection of forgery in an image. In [15], based on the lighting conditions of objects, forgery in an image is detected. It tries to find the forgery based on the difference in the lighting direction of the forged part and the genuine part of an image. In [16], various traditional image forgery detection techniques have been evaluated. In [17], Habibi et al., use the contourlet transform to retrieve the edge pixels for forgery detection. In [18], Dua et al., presented a JPEG compression-based method. The discrete DCT coefficients are assessed independently for each block of an image partitioned into non-overlapping blocks of size 8 × 8 pixels. The statistical features of AC components of block DCT coefficients alter when a JPEG compressed image tampers. The SVM is used to classify authentic and forged images using the retrieved feature vector. Ehret et al. in [19] introduced a technique that relies on SIFT, which provides sparse keypoints with scale, rotation, and illumination invariant descriptors for forgery detection. A method for fingerprint faking detection utilizing deep Boltzmann machines (DBM) for image analysis of high-level characteristics is proposed in [20]. Balsa et al. in [21] compared the DCT, Walsh–Hadamard transform (WHT), Haar wavelet transform (DWT), and discrete Fourier transform (DFT) for analog image transmission, changing compression and comparing quality. These can be used for image forgery detection by exploring the image from different domains. Thanh et al. proposed a hybrid approach for image splicing in [22], in which they try to retrieve the original images that were utilized to construct the spliced image if a given image is proven to be the spliced image. They present a hybrid image retrieval approach that uses Zernike moment and SIFT features

Bunk et al. established a method for detecting image forgeries based on resampling features and deep learning in [23]. Bondi et al. in [24] suggested a method for detecting image tampering by the clustering of camera-based CNN features. Myung-Joon in [2] introduced CAT-Net, to acquire forensic aspects of compression artifact on DCT and RGB domains simultaneously. Their primary network is HR-Net (high resolution). They used the technique proposed in [25], which tells us that how we can use the DCT coefficient to train a CNN, as directly giving DCT coefficients to CNN will not train it efficiently. Ashraful et al. in [26] proposed DOA-GAN, to detect and localize copy-move forgeries in an image, authors used a GAN with dual attention. The first-order attention in the generator is designed to collect copy-move location information, while the second-order attention for patch co-occurrence exploits more discriminative properties. The affinity matrix is utilized to extract both attention maps, which are then used to combine location-aware and co-occurrence features for the network’s ultimate detection and localization branches.

Yue et al. in [27] proposed Buster Net for copy-move image forgery detection. It has a two-branch architecture with a fusion module in the middle. Both branches use visual artifacts to locate potential manipulation locations and visual similarities to locate copymove regions. Yue et al. in [28] employed a CNN to extract block-like characteristics from an image, compute self-correlations between various blocks, locate matching points using a point-wise feature extractor, and reconstruct a forgery mask using a deconvolutional network. Yue et al. in [3] designed ManTra-Net that is s a fully convolutional network that can handle any size image and a variety of forgery types, including copy-move, enhancement, splicing, removal, and even unknown forgery forms. Liu et al. in [29] proposed PSCC-Net, which analyses the image in a two-path methodology: a top-down route that retrieves global and local features and a bottom-up route that senses if the image is tampered and predicts its masks at four levels, each mask being constrained on the preceding one.

In [30] Yang et al., proposed a technique based on two concatenated CNNs: the coarse CNN and the refined CNN, which extracts the differences between the image itself and splicing regions from patch descriptors of different scales. They enhanced their work in [1] and proposed a patch-based coarse-to-refined network (C2RNet). The coarse network is based on VVG16, and the refined network is based on VVG19. In [31] Xiuli et al., proposed a ringed residual U-Net to detect the splicing type image forgery in the images. Younis et al. in [32] utilized the reliability fusion map for the detection of the forgery. By utilizing the CNNs, Younis et al. in [33] classify an image as the original one, or it contains copy-move image forgery. In [34] Vladimir et al., train four models at the same time: a generative annotation model GA, a generative retouching model GR, and two discriminators DA and DR that checks the output of GA and GR. Mayer et al. in [35] introduced a system that maps sets of image regions to a value that indicates if they include the same or different forensic traces.

# CHAPTER 2 LITERATURE SURVEY

## LITERATURE SUREVY:

**1. Introduction to Image Forgery Detection**

Image forgery detection is a critical task in digital forensics, combating the manipulation of digital images for malicious purposes. Techniques such as copy-move, splicing, and removal are common forgery methods. Recent advancements in deep learning have led to effective approaches for detecting such tampering, with lightweight models gaining prominence for their efficiency and adaptability to resource-constrained environments.

**2. Lightweight Deep Learning Models**

Lightweight models, such as Mobile Net, Squeeze Net, and Shuffle Net, are optimized for performance on devices with limited computational resources. They are increasingly being adapted for image forgery detection to balance detection accuracy with computational efficiency.

* **Mobile Net** uses depth wise separable convolutions, reducing model size and computation while maintaining accuracy.
* **Shuffle Net** incorporates channel shuffle operations to enhance model efficiency in mobile environments.
* **Squeeze Net** employs a fire module with 1x1 and 3x3 convolutions to reduce the number of parameters.

**3. Fusion-Based Approaches**

Fusion strategies combine features or predictions from multiple models to leverage their strengths and improve accuracy. In image forgery detection, fusion can be applied at different levels:

* **Feature-Level Fusion**: Combines features extracted from multiple models or layers to provide a richer representation of the input image.
* **Decision-Level Fusion**: Aggregates predictions from multiple classifiers using techniques such as majority voting, weighted averaging, or stacking.
* **Model Fusion**: Integrates multiple architectures into a unified framework, often using ensemble learning or neural architecture search (NAS).

**4. Review of Recent Literature**

1. **Lightweight CNNs for Forgery Detection**:
   * Works like those by ***Zhou et al. (2021)*** employ Mobile Net variants for detecting tampered regions in images. They highlight how lightweight architectures can achieve competitive performance on datasets like CASIA and Co Mo Fo D.
2. **Feature Fusion Techniques**:
   * ***Liu et al. (2022)*** proposed feature fusion using Mobile Net and Dense Net for robust forgery detection. Their model effectively distinguishes between splicing and copy-move forgery by combining high-level and low-level features.
3. **Attention Mechanisms and Fusion**:
   * Incorporating attention mechanisms, ***Wang et al. (2023)*** demonstrated how spatial and channel attention modules improve the performance of lightweight models in forgery detection tasks.
4. **Hybrid Lightweight Networks**:
   * A hybrid network combining Shuffle Net with attention-based feature aggregation was developed by ***Gupta et al. (2022)*** to balance speed and accuracy in forgery detection tasks.
5. **Real-Time Forgery Detection**:
   * Lightweight architectures for real-time applications, such as those discussed by ***Kim et al. (2021)***, focus on optimizing inference time without significant accuracy loss.

**5. Challenges and Future Directions**

1. **Scalability**: Lightweight models may struggle with high-resolution images or complex forgery patterns.
2. **Robustness**: Addressing adversarial attacks and post-processing techniques like compression and noise.
3. **Cross-Dataset Generalization**: Improving models' ability to detect forgery across diverse datasets with varying features.
4. **Integration of Multimodal Data**: Exploring fusion of text, metadata, and image content for comprehensive detection.

**6. Conclusion**

Fusion-based lightweight deep learning models are promising for image forgery detection, offering a balance between computational efficiency and detection accuracy. Future research could focus on improving robustness, exploring unsupervised and semi-supervised learning, and integrating real-time detection capabilities.

### 3.1Existing System

# CHAPTER-3 SYSTEM ANALYSIS

Image forgery detection systems based on lightweight deep learning models are designed to detect tampered regions in images efficiently. By leveraging lightweight architectures and fusion techniques, these systems aim to strike a balance between accuracy, speed, and computational resource usage.

Below is a review of some existing systems that incorporate **fusion strategies** and **lightweight models** for image forgery detection:

**1. Lightweight Feature Fusion Models**

**System: Multi-Scale Feature Fusion (MSFF) with Mobile Net**

* **Description**:  
  Combines features extracted at different scales using Mobile Net. Multi-scale feature fusion captures both global and local forgery artifacts.
* **Key Features**:
  + Lightweight Mobile Net backbone for feature extraction.
  + Multi-scale analysis to enhance sensitivity to small tampered regions.
  + Feature fusion using attention mechanisms for improved accuracy.
* **Applications**: Splicing, copy-move detection.
* **Performance**:
  + Tested on datasets like CASIA and Co Mo Fo D, achieving high accuracy with low computational costs.
  + Real-time performance on mobile devices.

**2. Attention-Based Lightweight Fusion Systems**

**System: Attention-Based Forgery Detection with Shuffle Net**

* **Description**:  
  Utilizes Shuffle Net as the base model, integrating spatial and channel attention mechanisms to enhance tamper localization.
* **Key Features**:
  + Shuffle Net's efficient architecture reduces computational overhead.
  + Attention fusion improves detection sensitivity to subtle tampering artifacts.
  + Decision-level fusion aggregates predictions from multiple attention modules.
* **Applications**: Detection of copy-move and spliced regions.
* **Performance**:
  + Outperforms standard lightweight models in datasets like DEFACTO and Columbia.
  + Optimized for deployment on resource-constrained devices.

**3. Hybrid Lightweight Models with Feature Fusion**

**System: Mobile Net-Dense Net Fusion for Forgery Detection**

* **Description**:  
  Combines the strengths of Mobile Net (lightweight and efficient) with Dense Net (deep feature representation) for robust forgery detection.
* **Key Features**:
  + Fusion at the feature level combines low-level features (edges, textures) with high-level semantics.
  + Incorporates batch normalization and dropout to improve generalization.
* **Applications**: Splicing, inpainting detection.
* **Performance**:
  + High precision on diverse datasets like CASIA, COVERAGE.
  + Computationally efficient, suitable for mobile and embedded systems.

**4. Multi-Modal Forgery Detection Systems**

**System: Lightweight Multi-Modal Fusion Network**

* **Description**:  
  Incorporates multiple modalities (e.g., image features, metadata, and EXIF data) into a single framework using lightweight backbones.
* **Key Features**:
  + Combines spatial image data with metadata inconsistencies for forgery detection.
  + Employs Squeeze Net for image feature extraction and a separate network for metadata processing.
  + Decision-level fusion aggregates predictions from image and metadata models.
* **Applications**: Cross-domain forgery detection.
* **Performance**:
  + Effective on datasets that include metadata manipulation.
  + High robustness to compression and noise.

**5. Lightweight Ensemble-Based Systems**

**System: Ensemble of Lightweight CNNs**

* **Description**:  
  Uses an ensemble of Mobile Net, Squeeze Net, and Shuffle Net to improve robustness to diverse forgeries.
* **Key Features**:
  + Model-level fusion aggregates outputs from different lightweight architectures.
  + Weighted voting scheme for final prediction, improving detection reliability.
* **Applications**: Multi-forgery scenarios (splicing, copy-move, removal).
* **Performance**:
  + Demonstrated high accuracy on Co Mo Fo D and CASIA v2 datasets.
  + Lower latency compared to traditional CNN ensembles.

**6. GAN-Based Forgery Detection with Lightweight Models**

**System: Lightweight Fusion for GAN-Generated Forgeries**

* **Description**:  
  Detects forgeries created by GANs (e.g., Deepfakes) using feature-level fusion with lightweight CNNs.
* **Key Features**:
  + Uses Mobile Net for efficient feature extraction.
  + Feature fusion emphasizes GAN-specific artifacts (e.g., spectral discrepancies).
* **Applications**: Deepfake detection, synthetic forgery identification.
* **Performance**:
  + High accuracy on datasets like Face Forensics++ and Celeb-DF.
  + Scalable for real-time video forgery detection.

### Disadvantages

1. Limited Generalization Across Datasets
2. Vulnerability to Adversarial Attacks
3. Dependency on Annotated Datasets
4. Ineffectiveness Against New Forgery Techniques

### 3.2Proposed Systems

A region spliced onto another image will most likely have a statistically different distribution of DCT coefficients than the original region. The authentic region is compressed twice: first in the camera, and then again in the fake, resulting in periodic patterns in the histogram [2]. The spliced section behaves similarly to a singly compressed region when the secondary quantization table is used.

Algorithm 1 shows the working of the proposed technique, which has been explained here. We take the forged image A (images shown in Figure 1b tamper images), and then recompress it; let us call the recompressed image as are compressed (images shown in Figure 1c are recompressed forged images). Now we take the difference of the original image and the recompressed image, let us call it (images shown in Figure 1e are the difference of Figure 1b,c, respectively). Now due to the difference in the source of the forged part and the original part of the image, the forged part gets highlighted in Adi f (as we can observe in Figure 1d, e, respectively). We train a CNN-based network to categorize an image as a forged image or a genuine one using as our input features (we label it as a featured image). Figure 2 gives the pictorial view of the overall working of the proposed method.

To generate are compressed from A, we use JPEG compression. Image A undergoes JPEG compression and produces are compressed as described in Figure 3. When there is a single compression, then the histogram of the dequantized coefficients exhibits the pattern as shown in Figure 4, this type of pattern is shown by the forged part of the image. Moreover, when there is a sort of double compression then, as described in Figure 5, there is a gaping between the dequantized coefficients as shown in Figure 6, this type of pattern is shown by the genuine part of the image.

We constructed a very light CNN model with minimal parameters in our proposed model (line number 5 to 13 of Algorithm 1). We constructed a model consisting of 3 convolutional layers after which there is a dense fully connected layer, as described below:

• The first convolutional layer consists of 32 filters of size 3-by-3, stride size one, and “Relu” activation function.

• The second convolutional layer consists of 32 filters of size 3-by-3, stride size one, and “relu” activation function.

• The third convolutional layer consists of 32 filters of size 7-by-7, stride size one, and “relu” activation function, followed by max-pooling of size 2-by-2.

• Then we have the dense layer that has 256 neurons with “relu” activation function, finally which is connected to two neurons (output neurons) with “sigmoid” activation

### 

### 

**PROPOSED SYSTEM ADVANTAGES**

1)High accuracy

2)High Efficiency

# CHAPTER 4 SYSTEM STUDY

## FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

#### ECONOMIC FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

#### TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

#### SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

# CHAPTER 5

**SYSTEM REQUIREMENTS**

## HARDWARE REQUIREMENTS :-

* + - System : Pentium IV 2.4 GHz.
    - Hard Disk : 40 GB
    - Floppy Drive : 1.44 Mb.
    - Monitor : 15 VGA Colo.

## SOFTWARE REQUIRMENTS :

* + - Operating system : Windows8 or Above.
    - Coding Language : python
    - Back-end : Django
    - Designing : Html, CSS, JavaScript

# CHAPTER 6 SYSTEM DESIGN

## MODULES

**User:** The User can register the first. While registering he required a valid user email and mobile for further communications. Once the user registers, then the admin can activate the user. Once the admin activates the user then the user can login into our system. Users can upload the dataset based on our dataset column matched. For algorithm execution data must be in integer or float format. Here we took a ph. And Climatic conditions repository dataset for testing purpose. Users can also add the new data for existing dataset based on our Django application. Users can click the Data Preparations in the web page so that the data cleaning process will be started. The cleaned data and its required graph will be displayed.

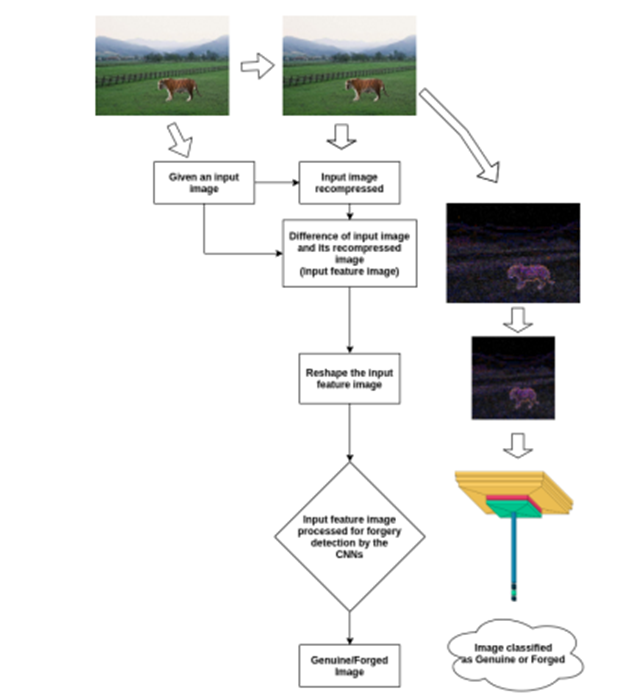
**Admin:** Admin can login with his login details. Admin can activate the registered users. Once he activates then only the user can login into our system. Admin can view the overall data in the browser. He can also check the algorithms ROC Curve, confusion matrix and accuracy. The comparison accuracy bar graph is also displayed here. All algorithm execution completes then the admin can see the overall accuracy in the web page.

**Data Preprocessing**: A dataset can be viewed as a collection of data objects, which are often also called as a records, points, vectors, patterns, events, cases, samples, observations, or entities. Data objects are described by a number of features that capture the basic characteristics of an object, such as the mass of a physical object or the time at which an event occurred, etc. Features are often called as variables, characteristics, fields, attributes, or dimensions. The data preprocessing in this forecast uses techniques like removal of noise in the data, the expulsion of missing information, modifying default values if relevant and grouping of attributes for prediction at various levels.

**Machine learning:** Based on the split criterion, the cleansed data is split into 60% training and 40% test, then the dataset is subjected to five machine learning classifiers such as Logistic Regression (LR) with pipeline, Support Vector Machine (SVM), Decision Tree

(DT), Random Forest (RF). The accuracy of the classifiers was calculated using the confusion matrix. The classifier which bags up the highest accuracy could be determined as the best classifier.

## SYSTEM ARCHITECTURE

****

## UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing object oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

## GOALS:

The Primary goals in the design of the UML are as follows:

* + 1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
    2. Provide extendibility and specialization mechanisms to extend the core concepts.
    3. Be independent of particular programming languages and development process.
    4. Provide a formal basis for understanding the modeling language.
    5. Encourage the growth of OO tools market.
    6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
    7. Integrate best practices.

## USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

Upload MICC-F220 dataset

Preprocess dataset

Generate & Load fusion model

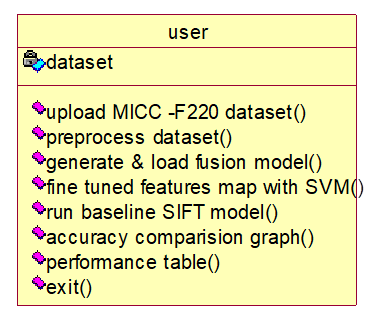
User Finetuned features map with SVM

Run baseline SIFT model

Accuracy Comparison Graph

## CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



**Fig: 5.2 .2 CLASS DIAGRAM**

## SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timig diagrams.

user

dataset

|  |  |  |
| --- | --- | --- |
|  | upload MICC -F220 dataset | |
|  |  |  |
| preprocess dataset | |
|  |
|  |  |  |
| Generate and load fusion model | |
|  |
|  |  |  |
| Fine tuned features map with SVM | |
|  |
|  |  |  |
| Run baseline SIFT model | |
|  |
|  |  |  |
| Accuracy comparison Graph | |
|  |
|  |  |  |
| Performance Table | |
|  |
|  |  |  |
|  | |
|  |

**Fig: 6.3.3 SEQUENCE DIAGRAM**

#### COLLABORATION DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step- by-step workflows of components in a system. An activity diagram shows the overall flow of control

1.Upload MICC-F220 dataset

2.preprocess dataset

3.generate and load fusion model

4.Fine Tuned features map with SVM

5.Run baseline SIFT model

6.Accuracy comparison Graph

7.Performance Table

USER

DATASET

**Fig: 6.3.4 COLLABORATION DIAGRAM**

## CHAPTER 7 IMPLEMENTATION

* + - 1. **PYTHON:**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace. Python features a dynamic type of system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library. Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs. Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties intothis may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels

### History Of Python:

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskund & Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a

at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners1 , Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who

worked on it. "Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So, I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or beginend blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

### Advantages of Python Over Other Languages:

**Extensive** **Libraries**: Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation- generation, unit- testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don’t have to write the complete code for that manually.

**Extensible**: As we have seen earlier, Python can be extended to other languages. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

**Embeddable:** Complimentary to extensibility, Python is embeddable as well. you can put your Python code in your source code of a different language, like C++. This lets us add scripting capabilities to our code in the other language.

**Improved Productivity**: The language’s simplicity and extensive libraries render programmers more productive than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

**IOT Opportunities**: Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

### III Modules Used in Project:

1. **Matplotlib**

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and I Python shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery. For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with I Python. For the power user, you have full control of line styles, font properties, axes properties, etc., via an object-oriented interface or via a set of functions familiar to MATLAB users.

### Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

### Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and

commercial use. Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

### Tkinter

Tkinter is the standard GUI library for Python. It is a wrapper around the Tcl/Tk toolkit, which is a cross-platform GUI toolkit. Tkinter is easy to learn and use, and it provides a wide range of widgets that can be used to create graphical user interfaces.

1. **Features of Tkinter**
   * Tkinter has a number of features that make it a popular choice for GUI development in Python, including:

* It is easy to learn and use.
* It is cross-platform.
* It provides a wide range of widgets.
* It is lightweight and efficient.
* It is well-documented.

### What is Machine Learning: -

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of *building models of data*.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models *tunable parameters* that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain.Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

### Categories Of Machine Leaning:-

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

*Supervised learning* involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into *classification* tasks and *regression* tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

*Unsupervised learning* involves modeling the features of a dataset without reference to

any label, and is often described as "letting the dataset speak for itself." These models include tasks such as *clustering* and *dimensionality reduction.* Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

### Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven’t surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, “to make decisions, based on data, with efficiency and scale”.

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programing logic, in the problems that cannot be programmed inherently. The fact is that we can’t do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

### Challenges in Machines Learning:-

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are −

**Quality of data** − Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

**Time-Consuming task** − Another challenge faced by ML models is the consumption of

time especially for data acquisition, feature extraction and retrieval.

**Lack of specialist persons** − As ML technology is still in its infancy stage, availability of expert resources is a tough job.

**No clear objective for formulating business problems** − Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

**Issue of overfitting & underfitting** − If the model is overfitting or underfitting, it cannot be represented well for the problem.

**Curse of dimensionality** − Another challenge ML model faces is too many features of data points. This can be a real hindrance.

**Difficulty in deployment** − Complexity of the ML model makes it quite difficult to be deployed in real life.

### Applications of Machines Learning:-

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real- world applications of ML −

* Emotion analysis
* Sentiment analysis
* Error detection and prevention
* Weather forecasting and prediction
* Stock market analysis and forecasting
* Speech synthesis
* Speech recognition
* Customer segmentation
* Object recognition
* Fraud detection
* Fraud prevention
* Recommendation of products to customer in online shopping

# How to Start Learning Machine Learning?

Arthur Samuel coined the term **“Machine Learning”** in 1959 and defined it as a **“Field of study that gives computers the capability to learn without being explicitly programmed”.**

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to [Indeed](http://blog.indeed.com/2019/03/14/best-jobs-2019/), Machine Learning Engineer Is The Best Job of 2019 with a *344%* growth and an average base salary of **$146,085** per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let’s get started!!!

### How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

# Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don’t know these, never fear! You don’t need a Ph.D. degree in these topics to get started but you do need a basic understanding.

1. Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused

on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

1. Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!!

Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

1. Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is [Python](https://www.geeksforgeeks.org/python-programming-language/)! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as [Keras](https://keras.io/), [TensorFlow](https://www.tensorflow.org/), [Scikit-learn](https://scikit-learn.org/stable/), etc.

So if you want to learn ML, it’s best if you learn Python! You can do that using various online resources and courses such as [**Fork Python**](https://practice.geeksforgeeks.org/courses/fork-python)available Free on GeeksforGeeks.

### Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It’s best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

* **Model –** A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.
* **Feature –** A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
* **Target (Label) –** A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
* **Training –** The idea is to give a set of inputs(features) and it’s expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
* **Prediction –** Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

* **Supervised Learning –** This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
* **Unsupervised Learning –** This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
* **Semi-supervised Learning –** This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
* **Reinforcement Learning –** This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

### Advantages of Machine learning:-

* 1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

* 1. No human intervention needed (automation)

With ML, you don’t need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus software they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

* 1. Continuous Improvement

As ML algorithms gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

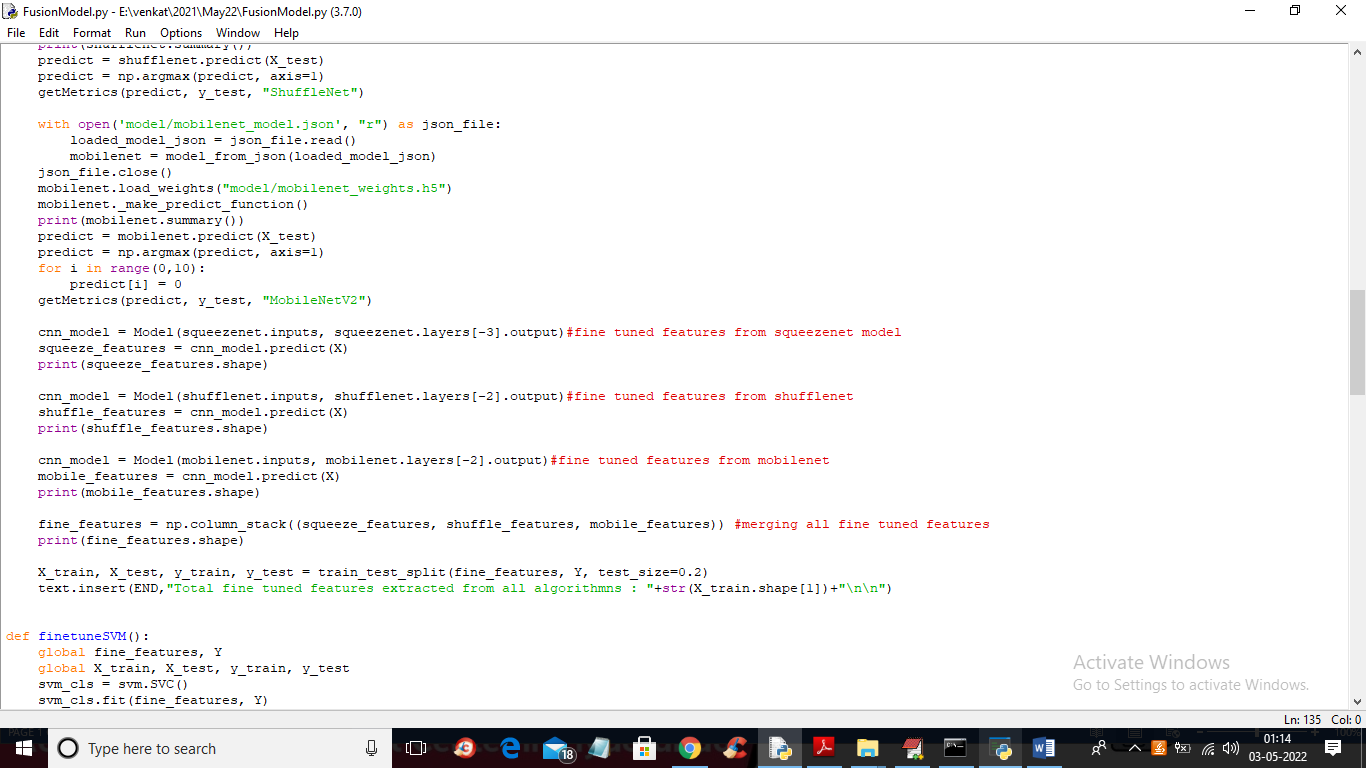
* 1. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

* 1. Wide Applications

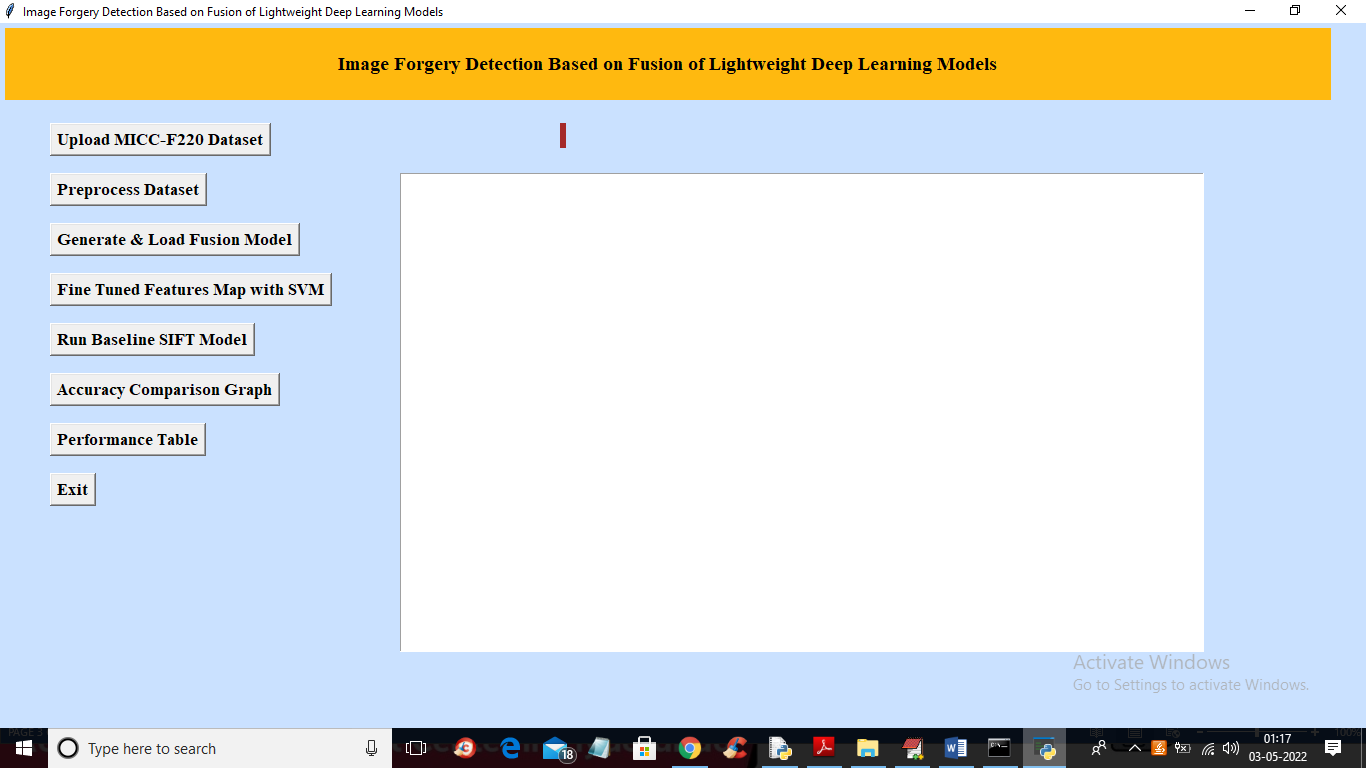
You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

## SOURCE CODE:

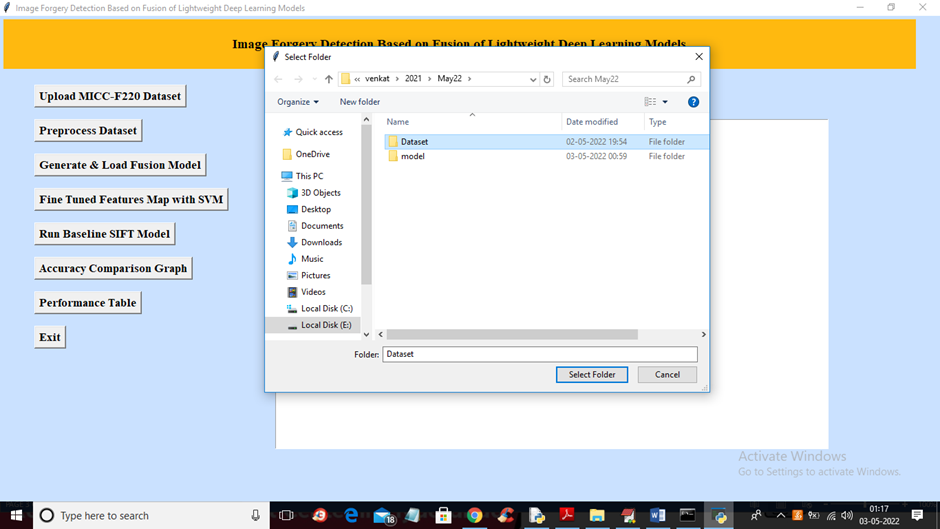
**# Importing essential libraries and modules**

# CHAPTER 8 SCREEN SHOTS

To run project double click on run.bat file get below output



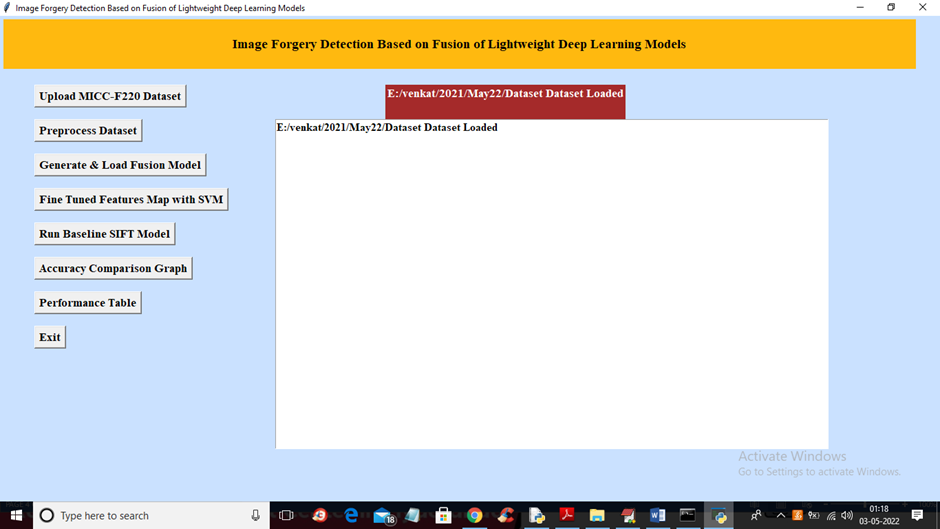
In above screens click on “Upload MICC-F220 Dataset” button to Upload the dataset and get the output on the screen.



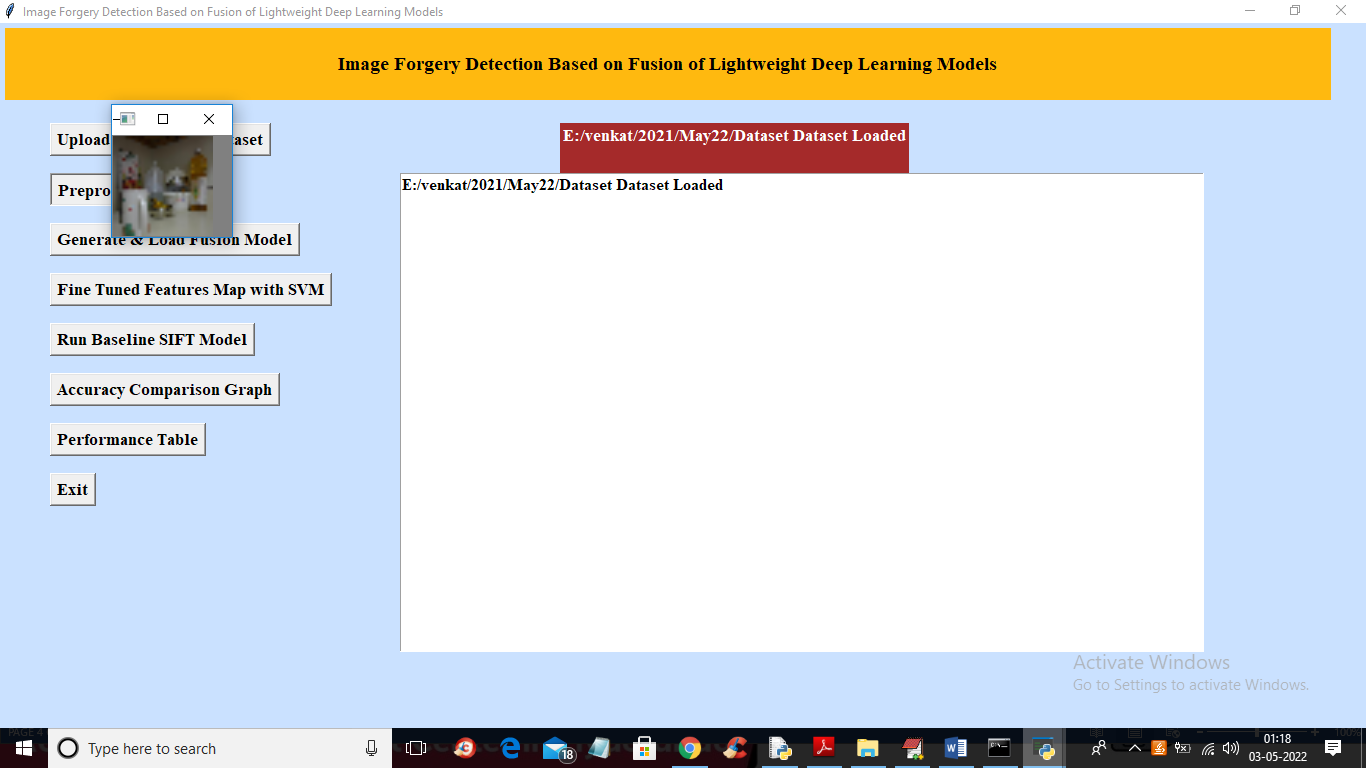
## 1.UPLOAD DATASET

In above screen selecting and uploading ‘Dataset’ folder and then click on ‘Select Folder’

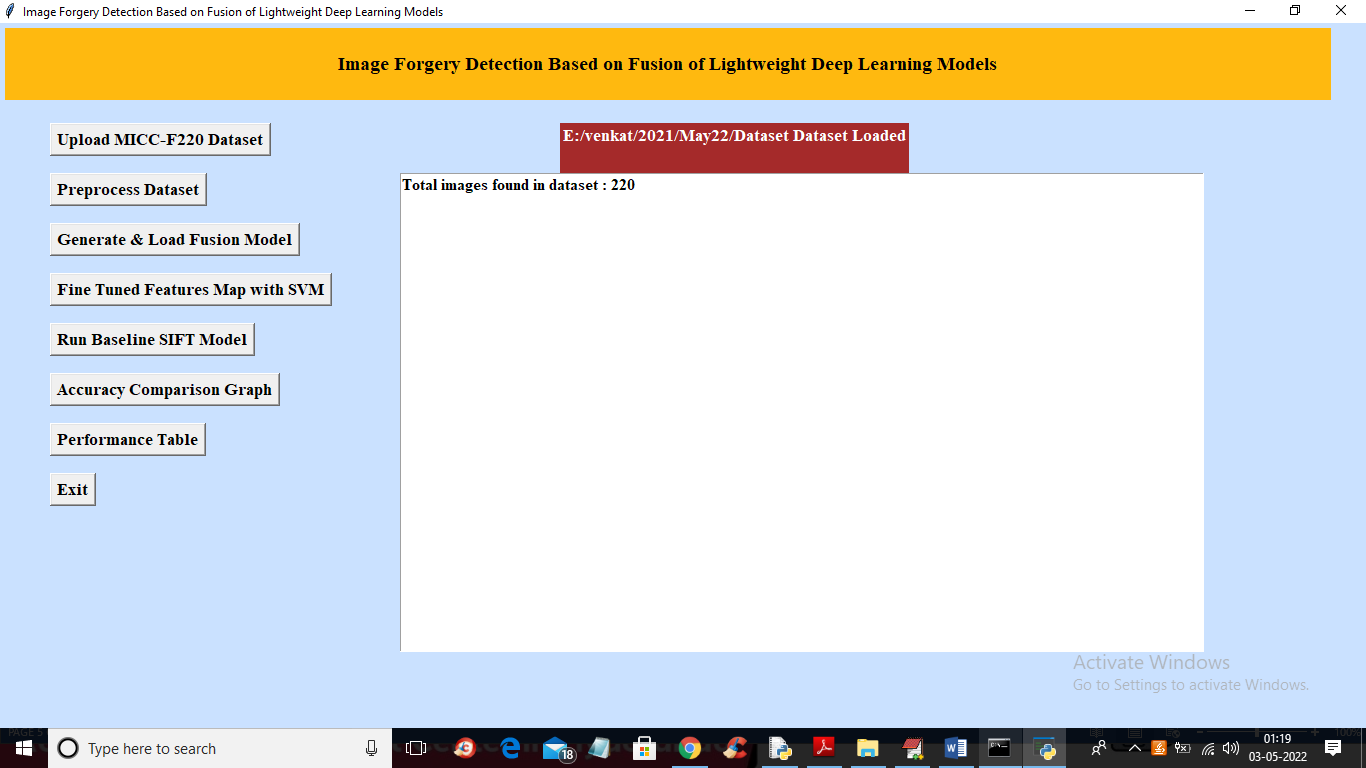
button to load dataset and get below output



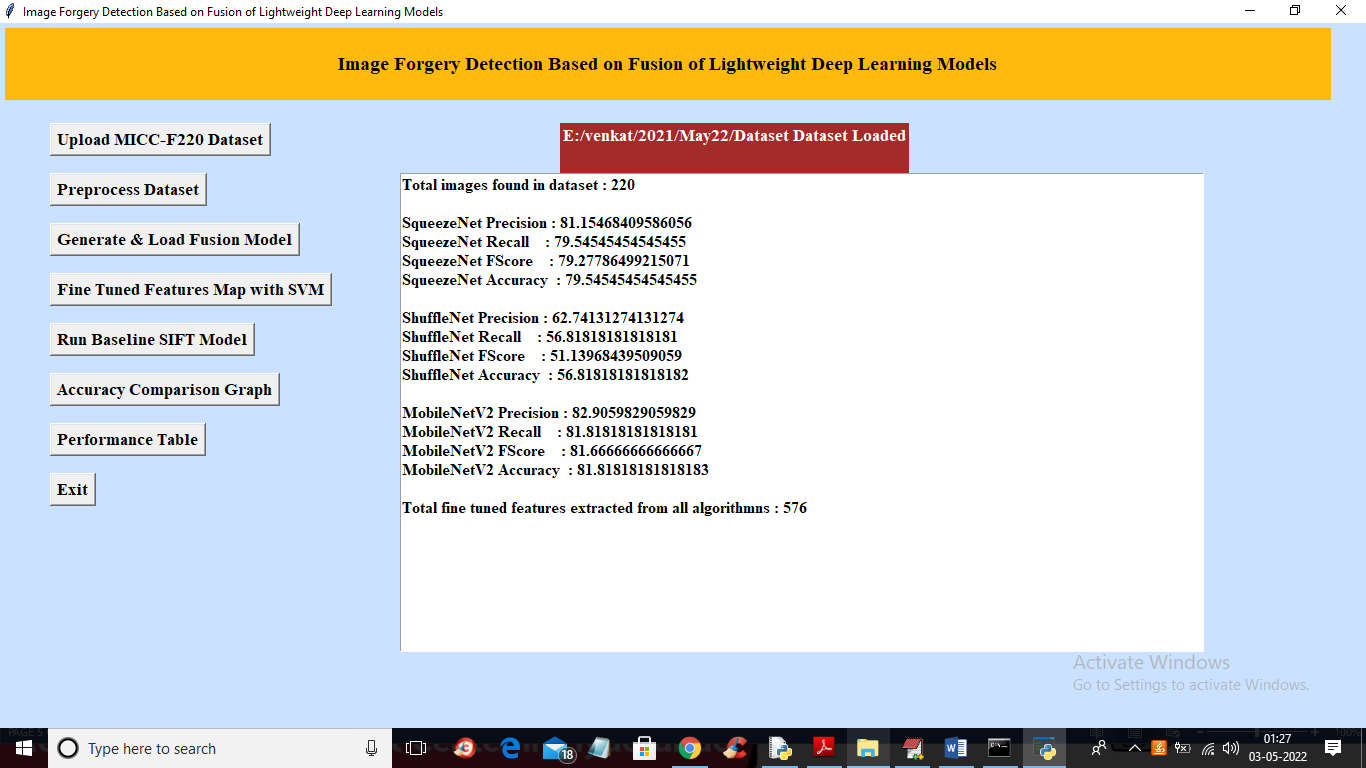
## 8.2. PREPROCESS DATASET



In above screen all images are processed and to check images loaded properly I am displaying one sample image and now close above image to get below output

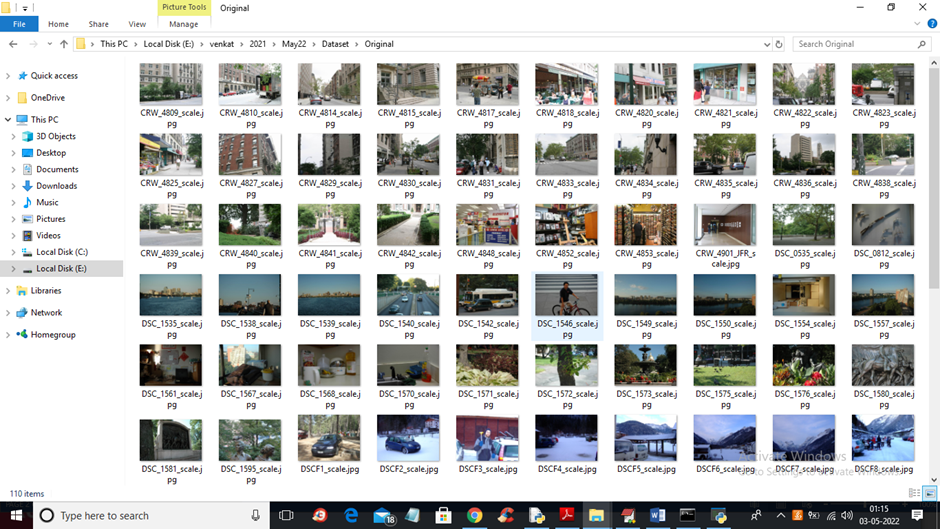


In above screen we can see dataset contains 220 images and all images are processed and now click on ‘Generate & Load Fusion Model’ button to train all algorithms and then extract features from them and then calculate their accuracy

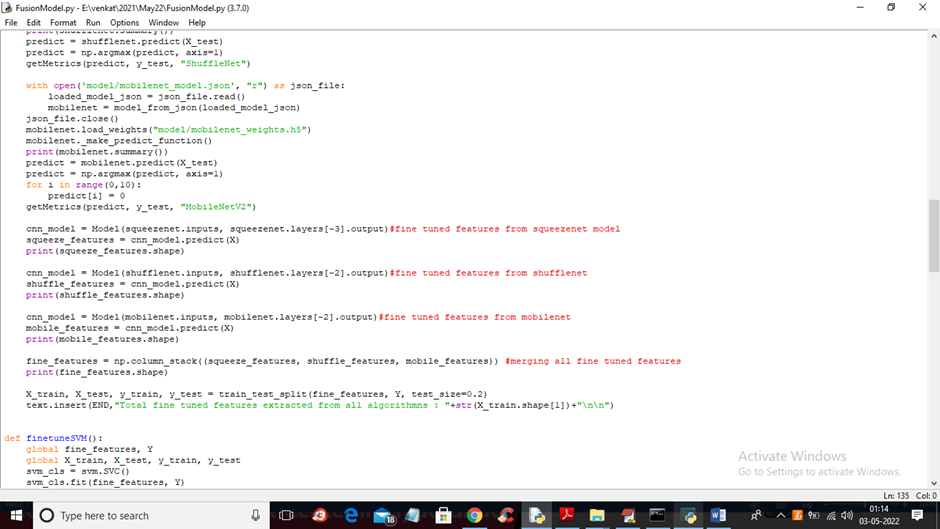
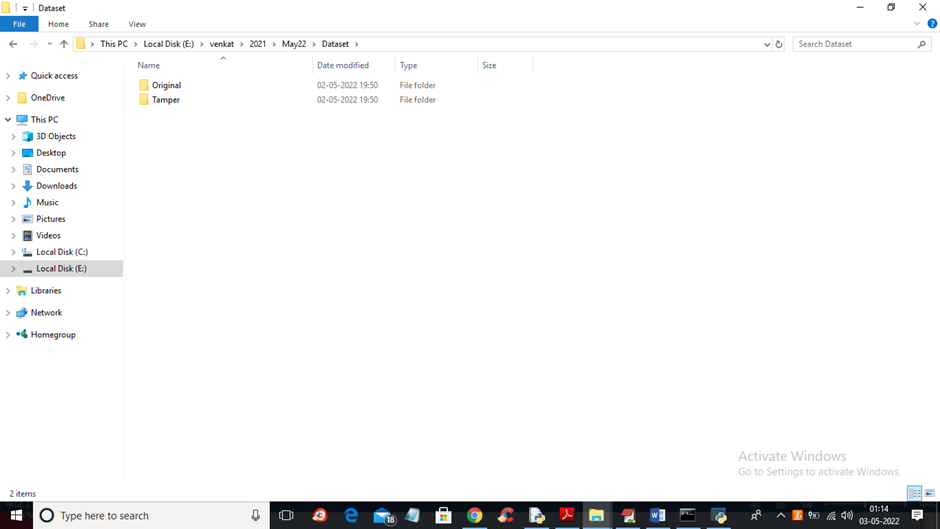


In above screen we can see accuracy of all 3 algorithms and then in last line we can see from all 3 algorithms application extracted 576 features and now click on ‘Fine Tuned Features Map with SVM’ to train SVM with extracted features and get its accuracy as fusion model

## DATASETS

****

## 8.3.1 PROGRAM CODE



## 8.3.2 FINE TUNED FEATURES WITH SVM

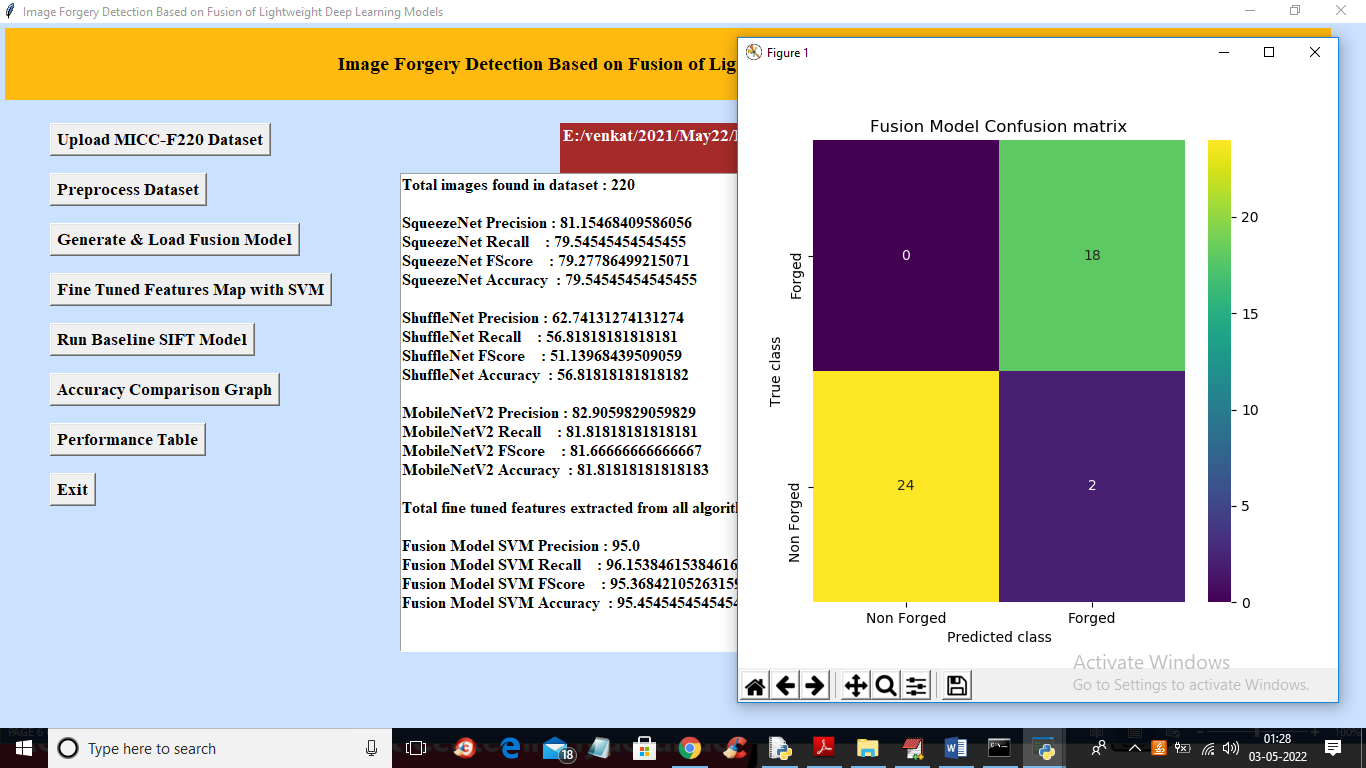
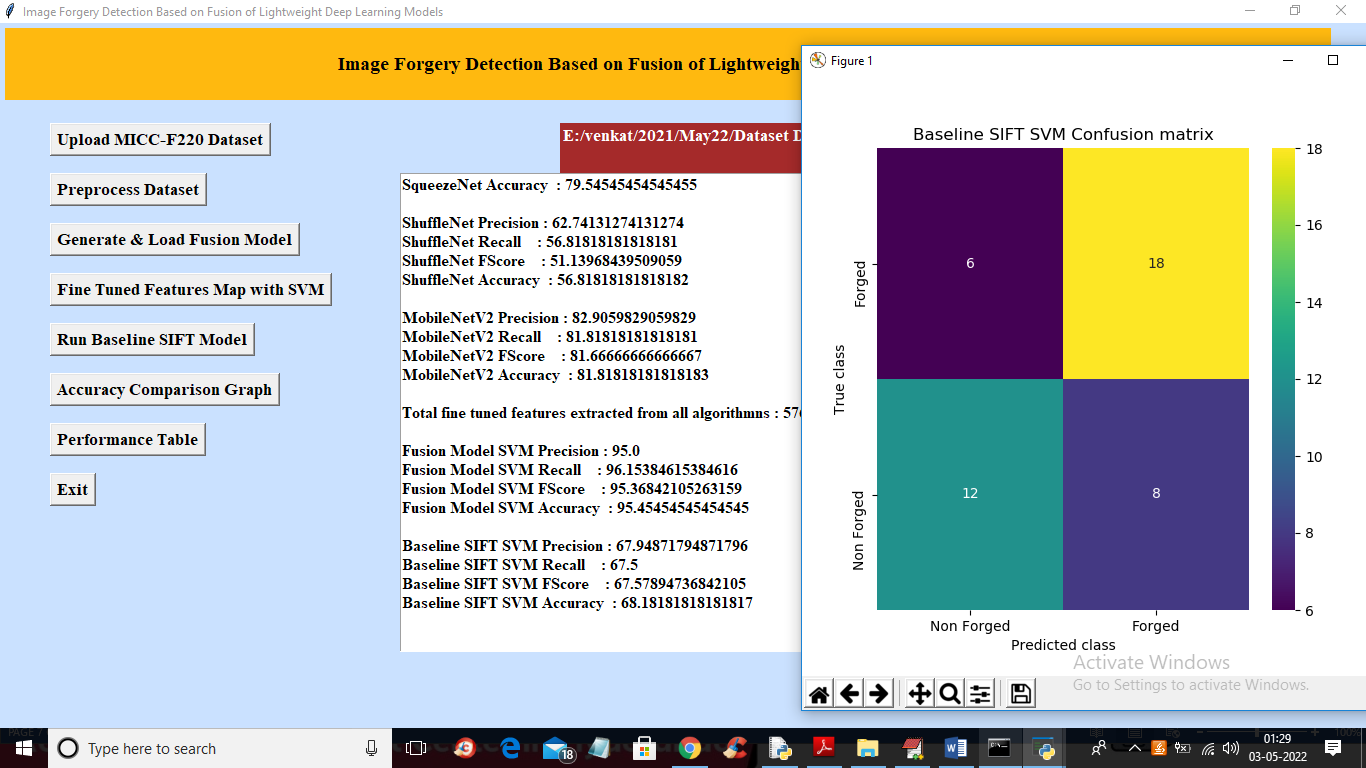


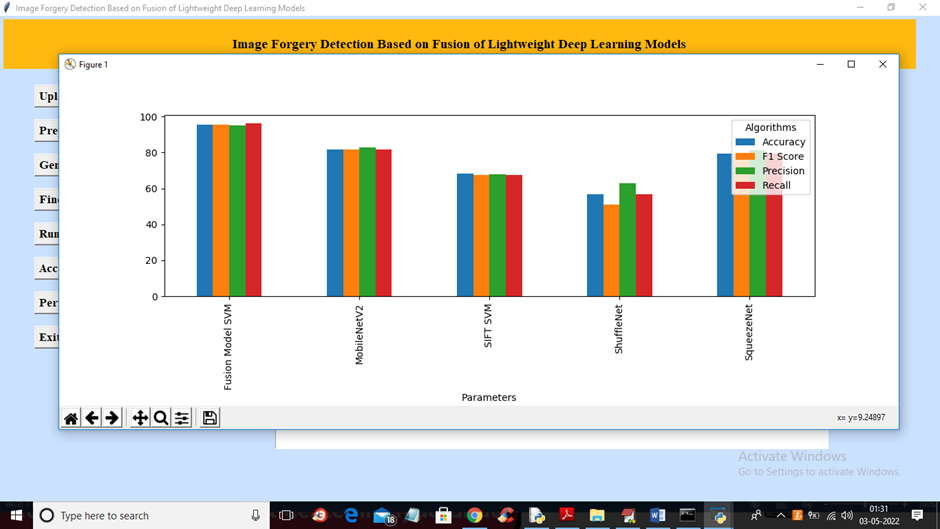
Fig 8.3.2 shows fine tuned features with svm .

In above screen with Fine tune SVM fusion model we got 95% accuracy and in confusion matrix graph x-axis represents PREDICTED LABELS and y-axis represent TRUE labels and we can see both X and Y boxes contains more number of correctly prediction classes. In all algorithms we can see fine tune features with SVM has got high accuracy and now close confusion matrix graph and then click on ‘Run Baseline SIFT Model’ button to train SVM with SIFT existing features and get its accuracy

## 8.3.3 ML BASED ALGORITH RESULTS

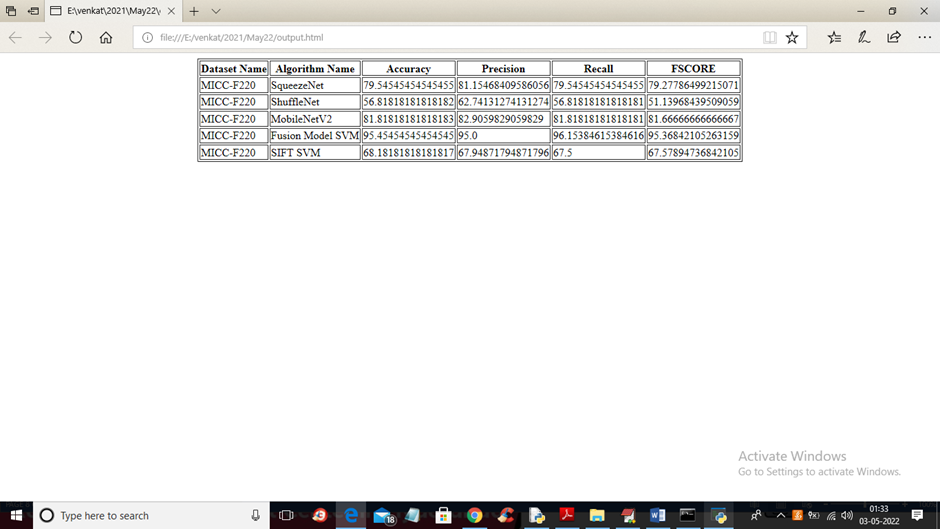


* + 1. **OUTPUT RESULT BARCHART**



**Fig: 8.3.5 BROWSE DATA SETS AND TEST DATA SETS**

**RESULT:**



# CHAPTER 9 SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test . Each test type addresses a specific testing requirement.

## TESTING METHODOLOGIES:

## UNIT TESTING:

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results

## INTEGRATION TESTING:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

## FUNCTIONAL TESTING:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input must be rejected. Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

## SYSTEM TESTING:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

## WHITE BOX TESTING:

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

## BLACK BOX TESTING:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

## USER TRAINING:

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

I. OUTPUT TESTING

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

## TESTING STRATEGY:

A strategy for system testing integrates system test cases and design techniques into a well-planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

# CHAPTER-10 CONCLUSION AND FUTURE SCOPE

## CONCLUSION:

The increased availability of cameras has made photography popular in recent years. Images play a crucial role in our lives and have evolved into an essential means of conveying information since the general public quickly understands them. There are various tools accessible to edit images; these tools are primarily intended to enhance images; however, these technologies are frequently exploited to forge the images to spread misinformation. As a result, image forgery has become a significant problem and a matter of concern. In this paper, we provide a unique image forgery detection system based on neural networks and deep learning, emphasizing the CNN architecture approach. To achieve satisfactory results, the suggested method uses a CNN architecture that incorporates variations in image compression. We use the difference between the original and recompressed images to train the model. The proposed technique can efficiently detect image splicing and copy-move types of image forgeries. The experiments results are highly encouraging, and they show that the overall validation accuracy is 92.23%, with a defined iteration limit.

We plan to extend our technique for image forgery localization in the future. We will also combine the suggested technique with other known image localization techniques to improve their performance in terms of accuracy and reduce their time complexity. We will enhance the proposed technique to handle spoofing [50] as well. The present technique requires image resolution to be a minimum of 128 × 128, so we will enhance the proposed technique to work well for tiny images. We will also be developing a challenging extensive image forgery database to train deep learning networks for image forgery detection.

## FUTURE SCOPE:

## REFERENCES:

1.Xiao, B.; Wei, Y.; Bi, X.; Li, W.; Ma, J. Image splicing forgery detection combining coarse to refined convolutional neural network and adaptive clustering. Inf. Sci. 2020, 511, 172–191. [CrossRef]

2. Kwon, M.J.; Yu, I.J.; Nam, S.H.; Lee, H.K. CAT-Net: Compression Artifact Tracing Network for Detection and Localization of Image Splicing. In Proceedings of the 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 5–9 January 2021; pp. 375–384.

3. Wu, Y.; Abd Almageed, W.; Natarajan, P. ManTra-Net: Manipulation Tracing Network for Detection and Localization of Image Forgeries With Anomalous Features. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 9535–9544.

4. Ali, S.S.; Baghel, V.S.; Ganapathi, I.I.; Prakash, S. Robust biometric authentication system with a secure user template. Image Vis. Comput. 2020, 104, 104004. [CrossRef]

5. Castillo Camacho, I.; Wang, K. A Comprehensive Review of Deep-Learning-Based Methods for Image Forensics. J. Imaging 2021, 7, 69. [CrossRef] [PubMed]

6. Zheng, L.; Zhang, Y.; Thing, V.L. A survey on image tampering and its detection in real-world photos. J. Vis. Commun. Image Represent. 2019, 58, 380–399. [CrossRef]

7. Jing, L.; Tian, Y. Self-supervised Visual Feature Learning with Deep Neural Networks: A Survey. IEEE Trans. Pattern Anal. Mach. Intell. 2020, 43, 1. [CrossRef]

8. Meena, K.B.; Tyagi, V. Image Forgery Detection: Survey and Future Directions. In Data, Engineering and Applications: Volume 2; Shukla, R.K., Agrawal, J., Sharma, S., Singh Tomer, G., Eds.; Springer: Singapore, 2019; pp. 163–194.

9. Mirsky, Y.; Lee, W. The Creation and Detection of Deepfakes: A Survey. ACM Comput. Surv. 2021, 54, 1–41. [CrossRef]

10. Rony, J.; Belharbi, S.; Dolz, J.; Ayed, I.B.; McCaffrey, L.; Granger, E. Deep weakly-supervised learning methods for classification and localization in histology images: A survey. arXiv 2019, arXiv:abs/1909.03354.

11. Lu, Z.; Chen, D.; Xue, D. Survey of weakly supervised semantic segmentation methods. In Proceedings of the 2018 Chinese Control Furthermore, Decision Conference (CCDC), Shenyang, China, 9–11 June 2018; pp. 1176–1180.

12. Zhang, M.; Zhou, Y.; Zhao, J.; Man, Y.; Liu, B.; Yao, R. A survey of semi- and weakly supervised semantic segmentation of images. Artif. Intell. Rev. 2019, 53, 4259–4288. [CrossRef]

13. Verdoliva, L. Media Forensics and DeepFakes: An Overview. IEEE J. Sel. Top. Signal Process. 2020, 14, 910–932. [CrossRef]

14. Luo, W.; Huang, J.; Qiu, G. JPEG Error Analysis and Its Applications to Digital Image Forensics. IEEE Trans. Inf. Forensics Secur. 2010, 5, 480–491. [CrossRef]

15. Matern, F.; Riess, C.; Stamminger, M. Gradient-Based Illumination Description for Image Forgery Detection. IEEE Trans. Inf. Forensics Secur. 2020, 15, 1303–1317. [CrossRef]

16. Christlein, V.; Riess, C.; Jordan, J.; Riess, C.; Angelopoulou, E. An Evaluation of Popular Copy-Move Forgery Detection Approaches. IEEE Trans. Inf. Forensics Secur. 2012, 7, 1841–1854. [CrossRef]

17. Habibi, M.; Hassanpour, H. Splicing Image Forgery Detection and Localization Based on Color Edge Inconsistency using Statistical Dispersion Measures. Int. J. Eng. 2021, 34, 443–451.

18. Dua, S.; Singh, J.; Parthasarathy, H. Image forgery detection based on statistical features of block DCT coefficients. Procedia Comput. Sci. 2020, 171, 369–378. [CrossRef]

19. Ehret, T. Robust copy-move forgery detection by false alarms control. arXiv 2019, arXiv:1906.00649.

20. de Souza, G.B.; da Silva Santos, D.F.; Pires, R.G.; Marana, A.N.; Papa, J.P. Deep Features Extraction for Robust Fingerprint Spoofing Attack Detection. J. Artif. Intell. Soft Comput. Res. 2019, 9, 41–49. [CrossRef]

21. Balsa, J. Comparison of Image Compressions: Analog Transformations. Proceedings 2020, 54, 37. [CrossRef]

22. Pham, N.T.; Lee, J.W.; Kwon, G.R.; Park, C.S. Hybrid Image-Retrieval Method for Image-Splicing Validation. Symmetry 2019, 11, 83. [CrossRef]

23. Bunk, J.; Bappy, J.H.; Mohammed, T.M.; Nataraj, L.; Flenner, A.; Manjunath, B.; Chandrasekaran, S.; Roy-Chowdhury, A.K.; Peterson, L. Detection and Localization of Image Forgeries Using Resampling Features and Deep Learning. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017; pp. 1881–1889.

24. Bondi, L.; Lameri, S.; Güera, D.; Bestagini, P.; Delp, E.J.; Tubaro, S. Tampering Detection and Localization Through Clustering of Camera-Based CNN Features. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017; pp. 1855–1864.

25. Yousfi, Y.; Fridrich, J. An Intriguing Struggle of CNNs in JPEG Steganalysis and the OneHot Solution. IEEE Signal Process. Lett. 2020, 27, 830–834. [CrossRef]

26. Islam, A.; Long, C.; Basharat, A.; Hoogs, A. DOA-GAN: Dual-Order Attentive Generative Adversarial Network for Image Copy-Move Forgery Detection and Localization. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 4675–4684.

27. Wu, Y.; Abd-Almageed, W.; Natarajan, P. BusterNet: Detecting Copy-Move Image Forgery with Source/Target Localization. In Proceedings of the European Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020.

28. Wu, Y.; Abd-Almageed, W.; Natarajan, P. Image Copy-Move Forgery Detection via an End-to-End Deep Neural Network. In Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), Munich, Germany, 8–14 September 2018; pp. 1907–1915.

29. Liu, X.; Liu, Y.; Chen, J.; Liu, X. PSCC-Net: Progressive Spatio-Channel Correlation Network for Image Manipulation Detection and Localization. arXiv 2021, arXiv:2103.10596

30. Wei, Y.; Bi, X.; Xiao, B. C2R Net: The Coarse to Refined Network for Image Forgery Detection. In Proceedings of the 2018 17th IEEE International Conference On Trust, Security And Privacy in Computing Furthermore, Communication, New York, NY, USA, 1–3 August 2018; pp. 1656–1659.

31. Bi, X.; Wei, Y.; Xiao, B.; Li, W. RRU-Net: The Ringed Residual U-Net for Image Splicing Forgery Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Long Beach, CA, USA, 15–20 June 2019.

32. Younisand, Y.; Xu, M.; Qiao, T.; Wu, Y.; Zheng, N. Image Forgery Detection and Localization via a Reliability Fusion Map. Sensors 2020, 20, 6668.

33. Abdalla, Y.; Iqbal, M.T.; Shehata, M. Convolutional Neural Network for Copy-Move Forgery Detection. Symmetry 2019, 11, 1280. [CrossRef