**Image Forgery Detection Based on Fusion of Lightweight Deep Learning Models**

**ABSTARCT :**

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%.

**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 irreversible.

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.

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.

**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 learning based 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 BusterNet 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

**PROPOSED SYSTEM:**

CNNs, which are inspired by the human visual system, are designed to be non-linear interconnected neurons. They have already demonstrated extraordinary potential in a variety of computer vision applications, including image segmentation and object detection. They may be beneficial for a variety of additional purposes, including image forensics. With the various tools available today, image forgery is fairly simple to do, and because it is extremely dangerous, detecting it is crucial. When a fragment of an image is moved from one to another, a variety of artifacts occur due to the images’ disparate origins. While these artifacts may be undetectable to the naked eye, CNNs may detect their presence in faked images. Due to the fact that the source of the forged region and the background images are distinct, when we recompress such images, the forged is enhanced differently due to the compression difference. We use this concept in the proposed approach by training a CNN-based model to determine if an image is genuine or a fake.

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.

As previously stated, when an image is recompressed, if it contains a forgery, the forged portion of the image compresses differently from the remainder of the image due to the difference between the source of the original image and the source of the forged portion. When the difference between the original image and its recompressed version is analyzed, this considerably emphasizes the forgery component. As a result, we use it to train our CNN-based model for detecting image forgery.

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 Arecompressed (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 Adi f f (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 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 Adi f f 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 Arecompressed from A, we use JPEG compression. Image A undergoes JPEG compression and produces Arecompressed 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.

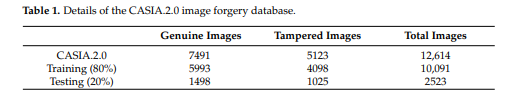
• The second convolutional layer consists of 32 filters of size 3-by-3, stride size one, and “relu” activation function.

• 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.

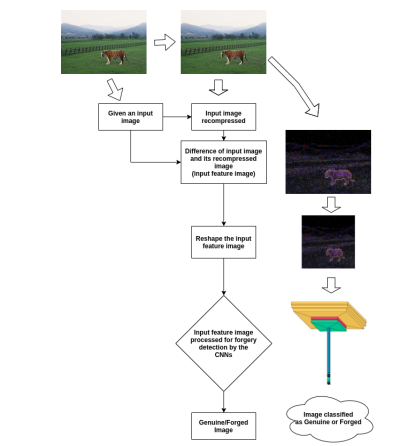
• Then we have the dense layer that has 256 neurons with “relu” activation function, finally which is connected to two neurons (output neurons) with “sigmoid” activation.

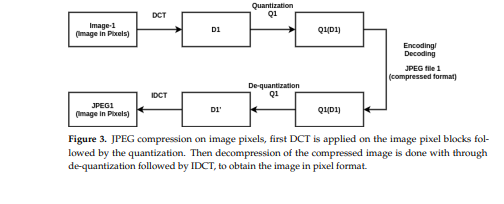


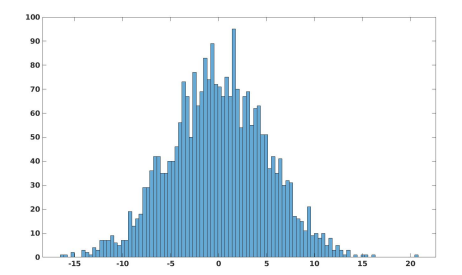
Various sample images and their processed forms: (a) the original images (RGB colour format); (b) the images with forgery (RGB colour format); (c) the recompressed forged images (RGB colour format); (d) ground truth of forgery (Binary format); (e) difference of the tampered image with its recompressed image (RGB colour format).

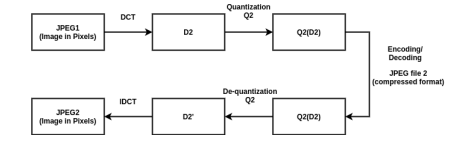


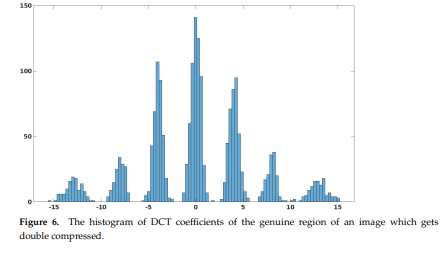
The feature image (Adi f f) is resized to 128 × 128 (Areshaped\_di f f) and then fed to the network. The network learns the presence of any tampering present through the feature images (images shown in Figure 1e). During training, the proposed model learns the existence of the forgery in an image through the numerous artifacts left behind during image forgery. The trained model can identify tampering with high accuracy, discussed in the next section.











**EXITING SYSTEM :**

This section describes the training and testing environment for the proposed approach. Aside from that, we’ll examine and contrast its performance with that of other techniques.

1. **Experimental Setup :**

We examined the proposed technique on a popular CASIA 2.0 image forgery database [22,49], to evaluate how efficient it is. There are a total of 12,614 images (in BMP, JPG, and TIF format), out of which 7491 are genuine images and 5123 tamper images. CASIA 2.0 includes images from various categories, including animals, architecture, articles, characters, plants, nature, scenes, textures, and indoor images. There are different-different sizes of the images present in the database; the resolution of images varies from 800 × 600 pixels to 384 × 256 pixels. The details about the CASIA 2.0 database are given in Table 1. A processor (Intel(R) Core(TM) i5-2400 CPU @ 3.1 GHz) having 16 GB RAM has been used for the experimentation.

Following terms are initially calculated for the evaluation:

• Total\_Images: The total number of images that were tested.

• TP (true positive): Correctly identified tampered images.

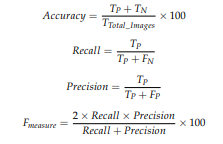
• TN (true negative): Correctly identified genuine images.

• FN (false negative): Wrongly identified tampered images, the tampered images which have been identified as genuine images.

• FP (false positive): Wrongly identified genuine images, the genuine images which have been identified as tampered images.

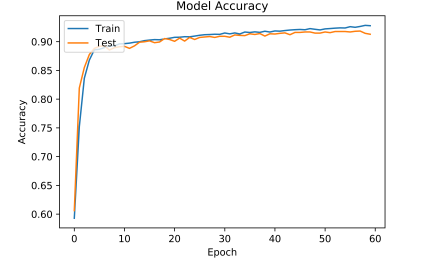
We calculate the accuracy, precision, recall, and Fmeasure [1] for the evaluation and the comparison of the proposed method with others. These are calculated as given below:

Now the accuracy is defined as given below:



1. **Model Training and Testing :**

To evaluate the proposed technique, we randomly divided the CASIA 2.0 database in the ratio of 80% and 20% (Table 1), we used 80% of the images (5993 authentic images, 4099 tampered images, total 10,092 images) for training the model. We used Adam optimizer with an initial learning rate of 1 × 10−5 and a batch size of 64. The remaining 20% images (1498 genuine images, 1024 tampered images, total 2522 images) are for testing the proposed model and comparing it with the other existing frameworks. Figure 7 illustrates the training and testing accuracy of the proposed model when trained on the CASIA 2.0 database with the settings mentioned above.



1. **Comparison with Other Techniques :**

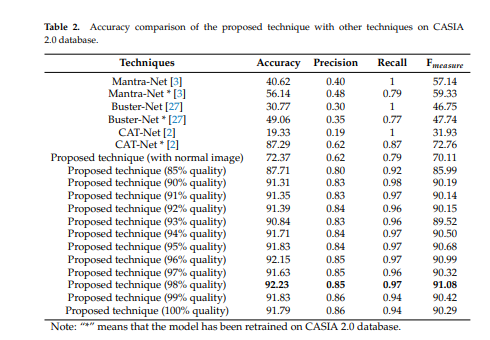
We compare the proposed technique to the other techniques in terms of accuracy and time required for image forgery detection.

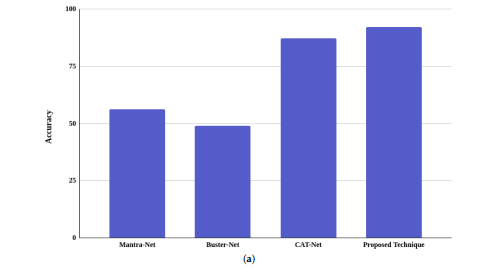
* 1. **Accuracy Comparison :**

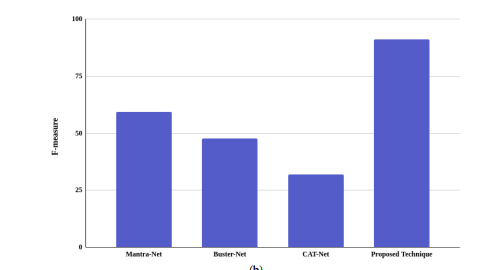
Table 2 shows the image forgery detection accuracy by the various techniques. These techniques and the proposed method have been evaluated on the same set of imagesof the CASIA 2.0 database. For the techniques mentioned in [2,3,27], when these techniquesprocess an input image, then, if the mask generated from them reports forgery, then we categorize the input image as tampered else; it is considered as genuine image. It must be noted that Buster-Net [27] is basically for copy-move type image forgeries, so we have used copy-move forged images to report its results. Whereas CAT-Net [2] is basically for splicing type of image forgeries, so we have used the images with splicing to report its results. Mantra-Net [3] can handle both image splicing and copy-move type image forgeries. It can be observed that we chose techniques that can handle either image splicing or copy-move, but also techniques that can handle both image splicing and copy-move type of image forgeries. All the techniques have CASIA 2.0 database as a common database for the evaluation. We have used these techniques’ publicly available trained model for evaluation. Apart from this, we retrained their models on the same CASIA 2.0 database images, on which the proposed model has been trained. The results obtained by retraining these models are also given in Table 2, along with their original models. After retraining, these techniques’ accuracy has improved; however, the proposed technique still outperforms them. CAT-Net [2], Buster-Net [27], and Mantra-Net [3] concentrate more on where the forgery is present (localization, where the output is pixel-level forgery detection) in the given image rather than focusing on that is the image is tampered or genuine (detection, where the output is a binary classification). However, the proposed technique focuses on whether the given image is tampered with or genuine.

The proposed technique achieved better forgery detection accuracy due to the fact that instead of directly using the original pixel image, it uses the feature image, which is the difference of the image with its recompressed image. This helps to detect image forgery better because it can be observed that in the feature image the forged part gets highlighted. Hence, it has resulted in achieving high accuracy. On the other hand, [2,3,27] show poor accuracy in image forgery detection as these techniques try to find image forgery at the pixel level, and due to this there are false positive pixels reported which reduces their overall forgery detection accuracy at the image level.

As mentioned in the previous section, we used JPEG compression to recompress the image; now, various quality factors are available while recompressing the image. So we have evaluated the proposed model for different JPEG quality factors and reported them as well. It is observed that the accuracy is better if the quality factor is kept at more than 90. The proposed technique achieved better accuracy as it utilizes better input features rather than directly using the original image as input features. To verify this we have trained our model by directly using the original images (instead of the better processed features), and its results are also reported in Table 2. It can be observed that in such a scenario the accuracy of the model drops from 92.23% to 72.37%, this shows the effectiveness of the processed input features (the difference of the original image with its recompressed version). Figure 8 show the comparison of the accuracy and the Fmeasure for the proposed method and the other techniques.





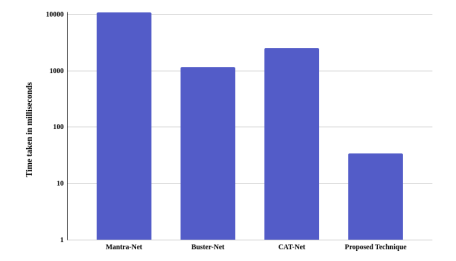


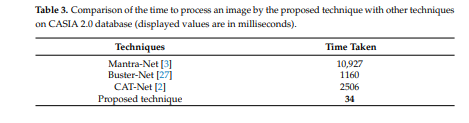
* 1. **Processing Time Comparison :**

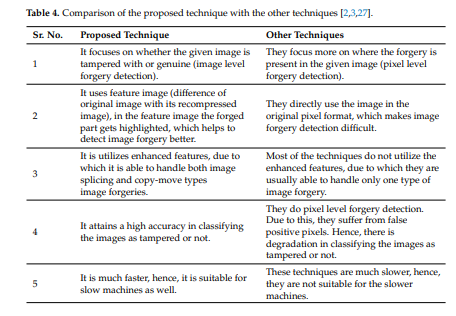
Table 3 shows the average time taken by the various techniques to process an image and to predict whether it is a genuine image or a tampered one. In terms of predicting the presence of forgery in an image, it can be noted that the proposed technique is fast and more efficient than the other state-of-the-art techniques. Figure 10 pictorially shows the comparison of the average time taken by the proposed method and the other techniques for the forgery detection in an image

This is because we provide an efficient feature image to our model, and the proposed CNN-based model is relatively light compared to other techniques. As a result, it can provide predictions in a much shorter amount of time. This makes our model will be advantageous in real-world scenarios. Table 4 shows the comparison of the proposed technique with the other techniques.









Hence, from the experimental results, the following observations can be made:

• Unlike other techniques, the proposed technique works well for both image splicing and copy-move types of image forgeries.

• It is highly efficient for image forgery detection and has exhibited significantly better performance than the other techniques.

• The difference in the compression of the forged part and the genuine part of the image is a good feature that can be learned by our CNN based model efficiently, which makes the proposed technique more robust in comparison to the other techniques.

• The proposed model is much faster than the other techniques, making it ideal and suitable for real-world usage, as it can be implemented even on slower machines.

**HARDWARE & SOFTWARE REQUIREMENTS:**

**HARD REQUIRMENTS :**

* System    :   Pentium IV 2.4 GHz.
* Hard Disk  :   40 GB.
* Floppy Drive :   1.44 Mb.
* Monitor   :   15 VGA Colour.
* Mouse    :   Logitech.
* Ram    :   512 MB.

**SOFTWARE REQUIRMENTS :**

* Operating system   : Windows 8Professional.
* Coding Language  : python

# 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

## ECONOMICAL 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.

# 4.SYSTEM DESIGN

## 4.1 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 objects 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.



# 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.



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 timing diagrams.



COLLABRATION 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.



## IMPLEMENTATION:

## MODULES:

1. Upload MICC-F220 Dataset: using this module we will upload dataset to application
2. Preprocess Dataset: using this module we will read all images and then normalize their pixel values and then resize them to equal size
3. Generate & Load Fusion Model: using this module we will train 3 algorithms called SqueezeNet, MobileNetV2 and ShuffleNet and then extract features from it to train fusion model. All algorithms prediction accuracy will be calculated on test data
4. Fine Tuned Features Map with SVM: using this module we will extract features from all 3 algorithms to form a fusion model and then fusion data get trained with SVM and then calculate its prediction accuracy.
5. Run Baseline SIFT Model: using this module we will extract SIFT existing technique features from images and then train with SVM and get its prediction accuracy
6. Accuracy Comparison Graph: using this module we will plot accuracy graph of all algorithms
7. Performance Table: using this module we will display all algorithms performance table.

# SOFTWARE ENVIRONMENT

## What is Python :

Below are some facts about Python.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

* + [Machine Learning](https://www.geeksforgeeks.org/machine-learning/)
  + GUI Applications (like Kivy, Tkinter, PyQt etc. )
  + Web frameworks like Django (used by YouTube, Instagram, Dropbox)
  + Image processing (like Opencv, Pillow)
  + Web scraping (like Scrapy, BeautifulSoup, Selenium)
  + Test frameworks
  + Multimedia

## Advantages of Python :-

Let’s see how Python dominates over other languages.

## 1. 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.

## 2. 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.

## 3. 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.

## 4. 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.

## 5. 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.

When working with Java, you may have to create a class to print **‘Hello World’**. But in Python, just a print statement will do. It is also quite **easy to learn, understand,** and**code.** This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

## 7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and **indentation is mandatory.** This further aids the readability of the code.

## 8. Object-Oriented

This language supports both the **procedural and object-oriented**programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the **encapsulation of data** and functions into one.

## 9. Free and Open-Source

Like we said earlier, Python is **freely available.** But not only can you[**download Python**](https://data-flair.training/blogs/install-python-windows/) for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

#### 10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn’t the same with Python. Here, you need to**code only once**, and you can run it anywhere. This is called **Write Once Run Anywhere (WORA)**. However, you need to be careful enough not to include any system-dependent features.

## 11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, **debugging is easier** than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

# Advantages of Python Over Other Languages :

## 1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don’t have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

## 2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

**The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.**

## 3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and [**machine learning**](https://data-flair.training/blogs/machine-learning-tutorials-home/), automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

## Disadvantages of Python

So far, we’ve seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let’s now see the downsides of choosing Python over another language.

#### 1. Speed Limitations

We have seen that Python code is executed line by line. But since [Python](https://www.python.org/) is interpreted, it often results in **slow execution**. This, however, isn’t a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

#### 2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the **client-side**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbonnelle**.

The reason it is not so famous despite the existence of Brython is that it isn’t that secure.

#### 3. Design Restrictions

As you know, Python is **dynamically-typed**. This means that you don’t need to declare the type of variable while writing the code. It uses **duck-typing**. But wait, what’s that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can**raise run-time errors**.

#### 4. Underdeveloped Database Access Layers

Compared to more widely used technologies like **JDBC (Java DataBase Connectivity)** and **ODBC (Open DataBase Connectivity)**, Python’s database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

#### 5. Simple

No, we’re not kidding. Python’s simplicity can indeed be a problem. Take my example. I don’t do Java, I’m more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

## 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 Wiskunde &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 project 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 begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

## 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.

#### (a) 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.

#### (b) 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.

#### (c) 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/" \t "_blank), [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.

#### 2. 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 softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

#### 3. Continuous Improvement

As [**ML algorithms**](https://data-flair.training/blogs/machine-learning-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.

#### 4. 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.

#### 5. 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.

### Disadvantages of Machine Learning :-

#### 1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

#### 2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

#### 3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

#### 4. High error-susceptibility

[Machine Learning](https://en.wikipedia.org/wiki/Machine_learning) is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

**Python Development Steps : -**

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system.  
Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked.Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode.Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it."Some changes in Python 7.3:

* Print is now a function
* Views and iterators instead of lists
* The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
* There is only one integer type left, i.e. int. long is int as well.
* The division of two integers returns a float instead of an integer. "//" can be used to have the "old" behaviour.
* Text Vs. Data Instead Of Unicode Vs. 8-bit

**Purpose :-**

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

**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 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 into this 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. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

**Modules Used in Project :-**

**Tensorflow**

TensorFlow is a [free](https://en.wikipedia.org/wiki/Free_software) and [open-source](https://en.wikipedia.org/wiki/Open-source_software) [software library for dataflow and differentiable programming](https://en.wikipedia.org/wiki/Library_(computing)) across a range of tasks. It is a symbolic math library, and is also used for [machine learning](https://en.wikipedia.org/wiki/Machine_learning) applications such as [neural networks](https://en.wikipedia.org/wiki/Neural_networks). It is used for both research and production at [Google](https://en.wikipedia.org/wiki/Google).‍

TensorFlow was developed by the [Google Brain](https://en.wikipedia.org/wiki/Google_Brain) team for internal Google use. It was released under the [Apache 2.0](https://en.wikipedia.org/wiki/Apache_License) [open-source license](https://en.wikipedia.org/wiki/Open-source_license) on November 9, 2015.

**Numpy**

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

* A powerful N-dimensional array object
* Sophisticated (broadcasting) functions
* Tools for integrating C/C++ and Fortran code
* Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

**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.

**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 [IPython](http://ipython.org/) shells, the [Jupyter](http://jupyter.org/) 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](https://matplotlib.org/tutorials/introductory/sample_plots.html) and [thumbnail gallery](https://matplotlib.org/gallery/index.html).

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. 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.

**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**

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 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 into this 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. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

**Install Python Step-by-Step in Windows and Mac :**

Python a versatile programming language doesn’t come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

## How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

**Note:** The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. [Download the Python Cheatsheet here.](https://myelearninghub.com/python-cheat-sheet/)The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

### Download the Correct version into the system

**Step 1:** Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: [https://www.python.org](https://www.python.org/)



Now, check for the latest and the correct version for your operating system.

**Step 2:** Click on the Download Tab.

****

**Step 3:** You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

****

**Step 4:** Scroll down the page until you find the Files option.

**Step 5:** Here you see a different version of python along with the operating system.



• To download Windows 32-bit python, you can select any one from the three options: Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.

•To download Windows 64-bit python, you can select any one from the three options: Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

**Note:** To know the changes or updates that are made in the version you can click on the Release Note Option.

### Installation of Python

**Step 1:** Go to Download and Open the downloaded python version to carry out the installation process.



**Step 2:** Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



**Step 3:** Click on Install NOW After the installation is successful. Click on Close.



With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

**Note:** The installation process might take a couple of minutes.

### Verify the Python Installation

**Step 1:** Click on Start

**Step 2:** In the Windows Run Command, type “cmd”.



**Step 3:** Open the Command prompt option.

**Step 4:** Let us test whether the python is correctly installed. Type **python –V** and press Enter.



**Step 5:** You will get the answer as 3.7.4

**Note:** If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

### Check how the Python IDLE works

**Step 1:** Click on Start

**Step 2:** In the Windows Run command, type “python idle”.



**Step 3:** Click on IDLE (Python 3.7 64-bit) and launch the program

**Step 4:** To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**



**Step 5:** Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

**Step 6:** Now for e.g. **enter print**

**6.SYSTEM TEST**

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, sub assemblies, 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.

### TYPES OF TESTS

**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 test**

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.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**System Test**

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.

**Unit Testing**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

# Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

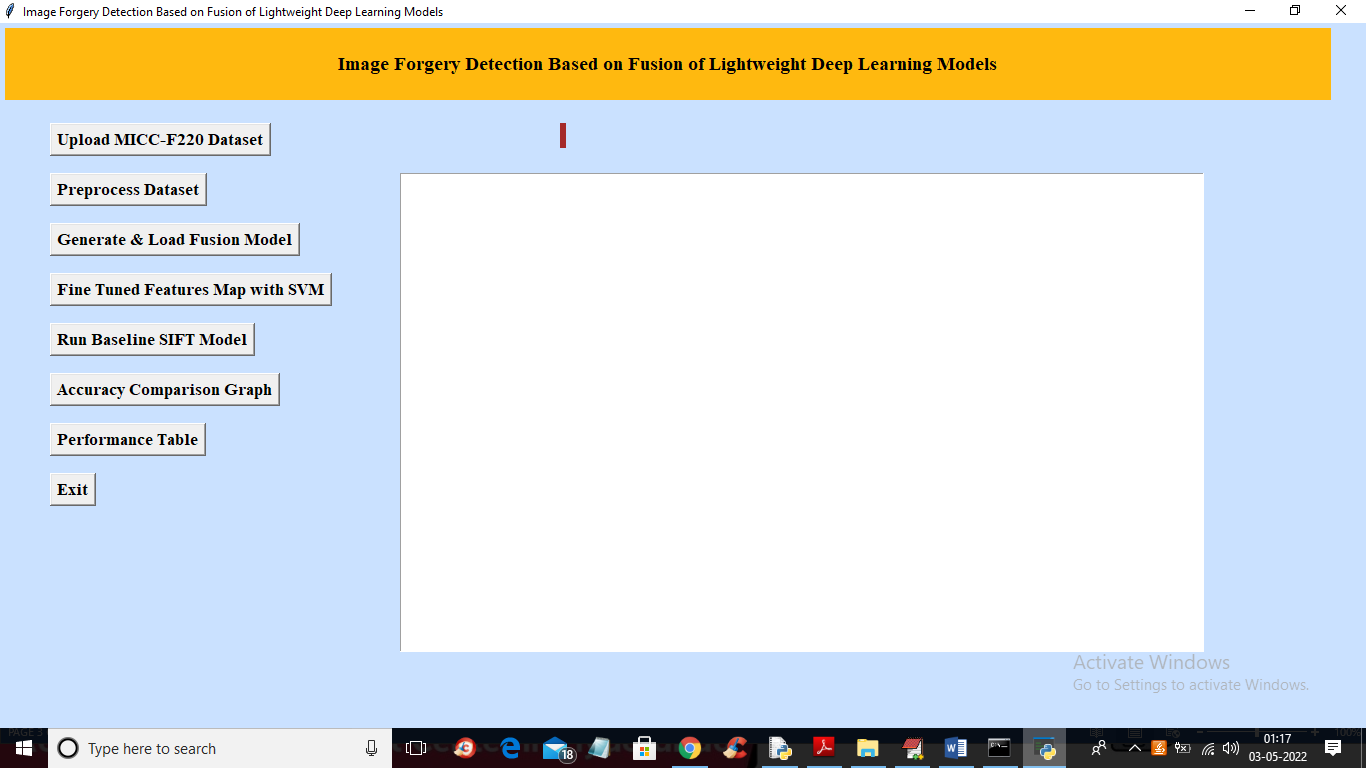
**Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

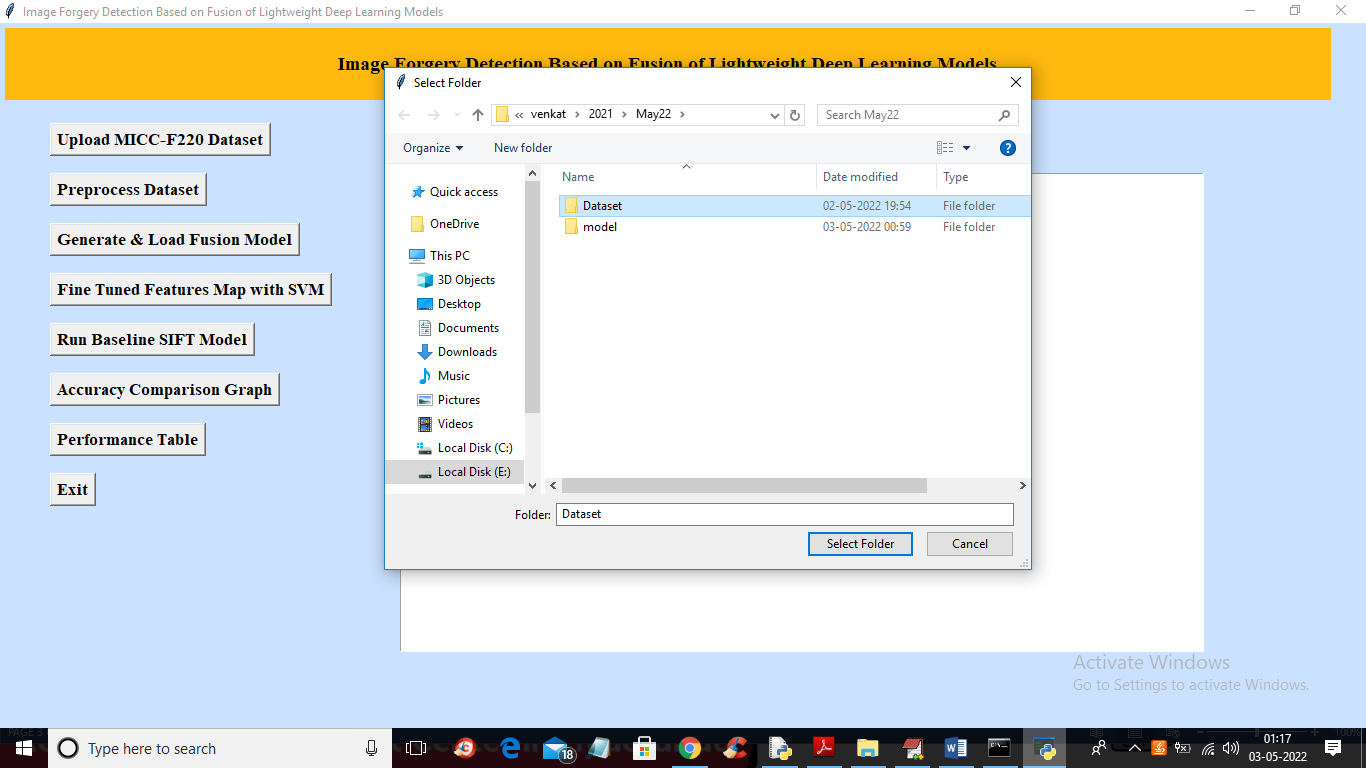
**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**SCREENSHOTS :**

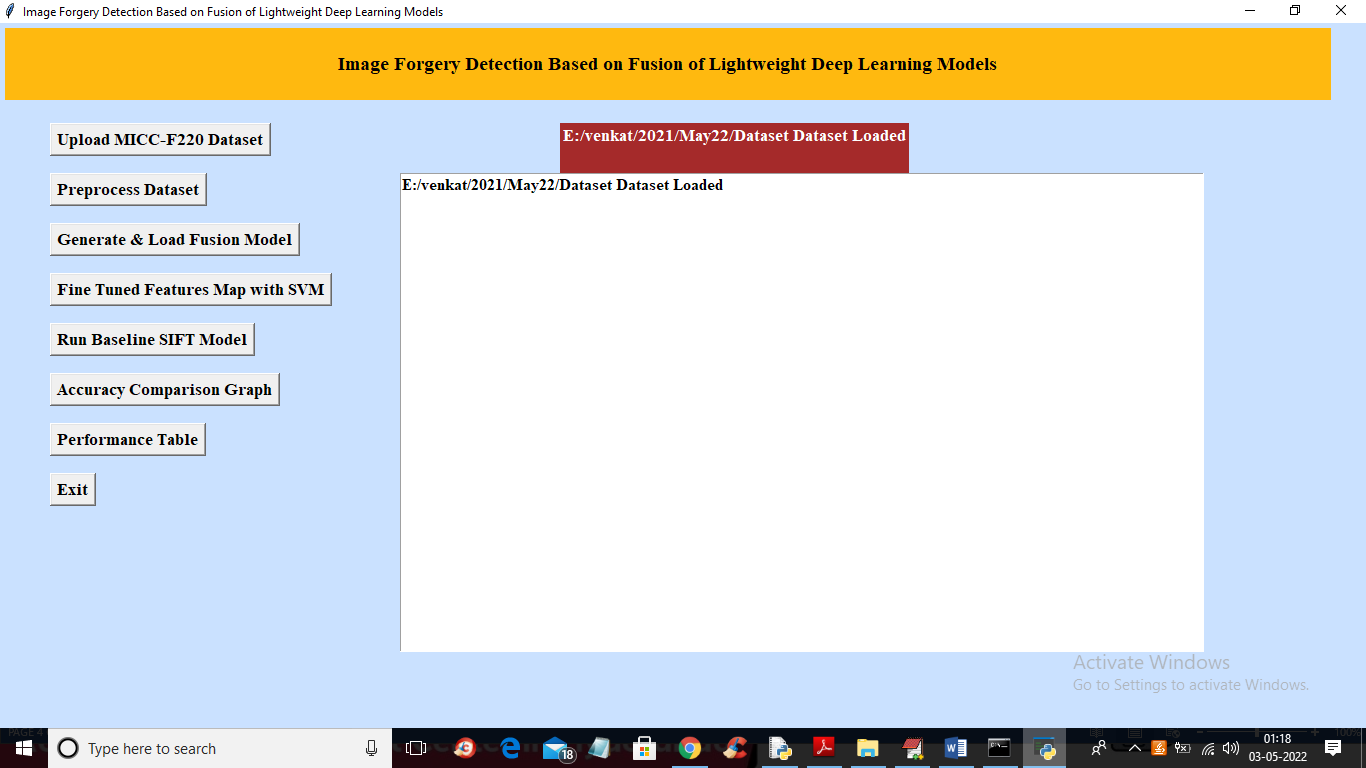
To run project double click on ‘run.bat’ file to get below output



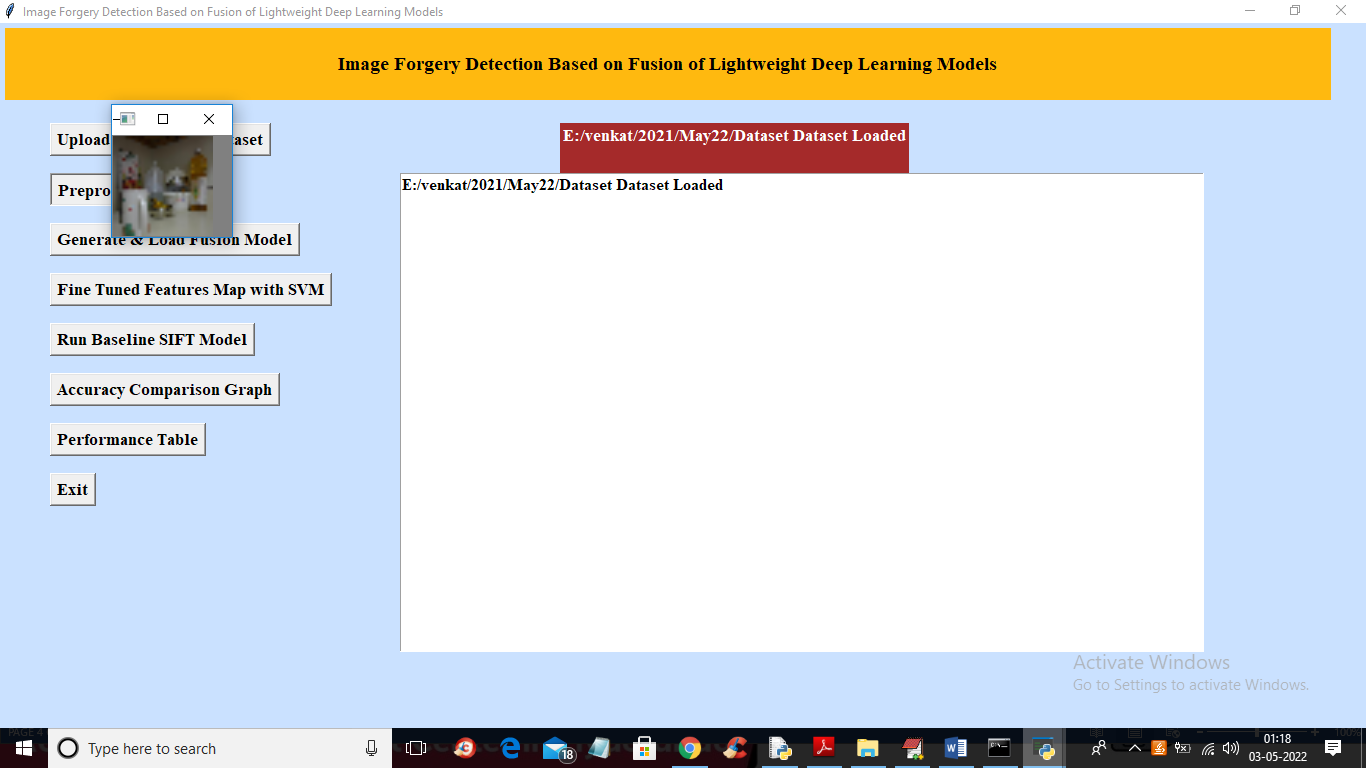
In above screen click on ‘Upload MICC-F220 Dataset’ button to upload dataset and get below output



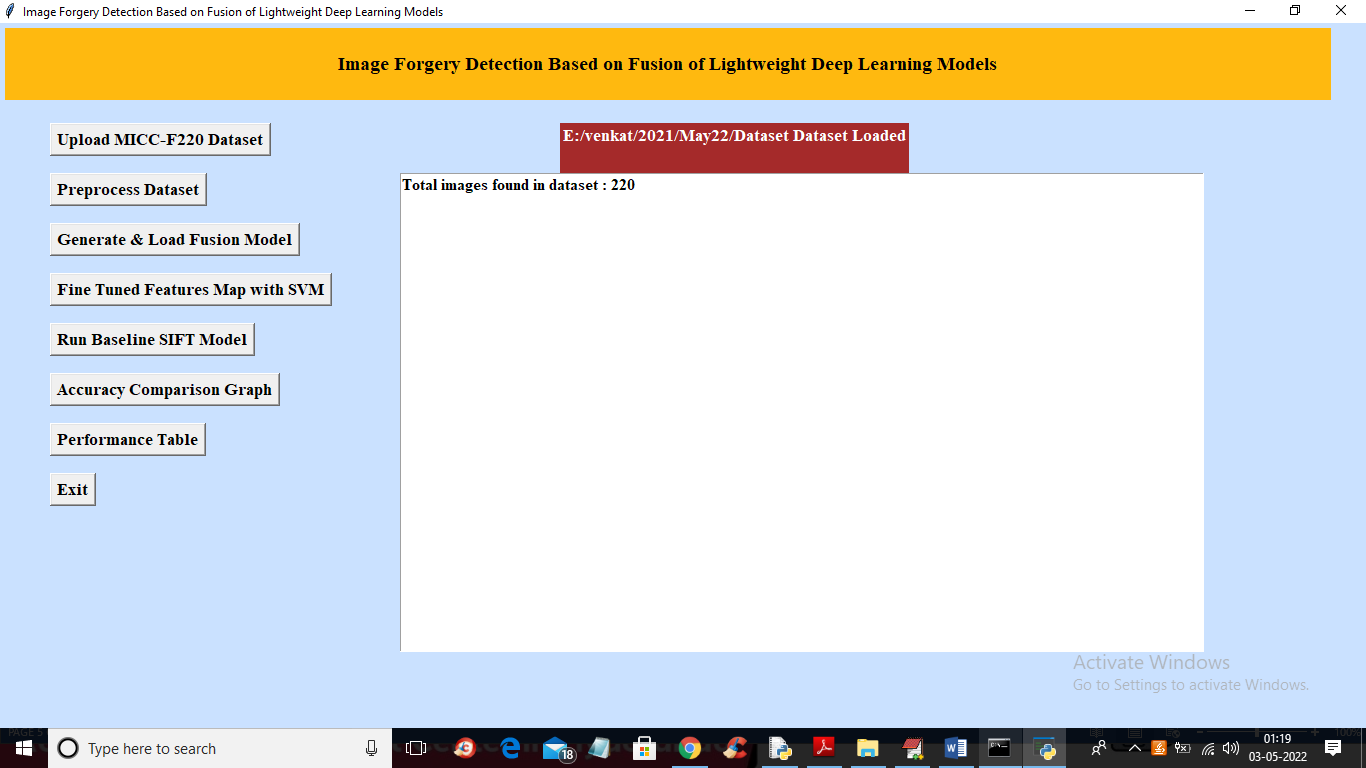
In above screen selecting and uploading ‘Dataset’ folder and then click on ‘Select Folder’ button to load dataset and get below output



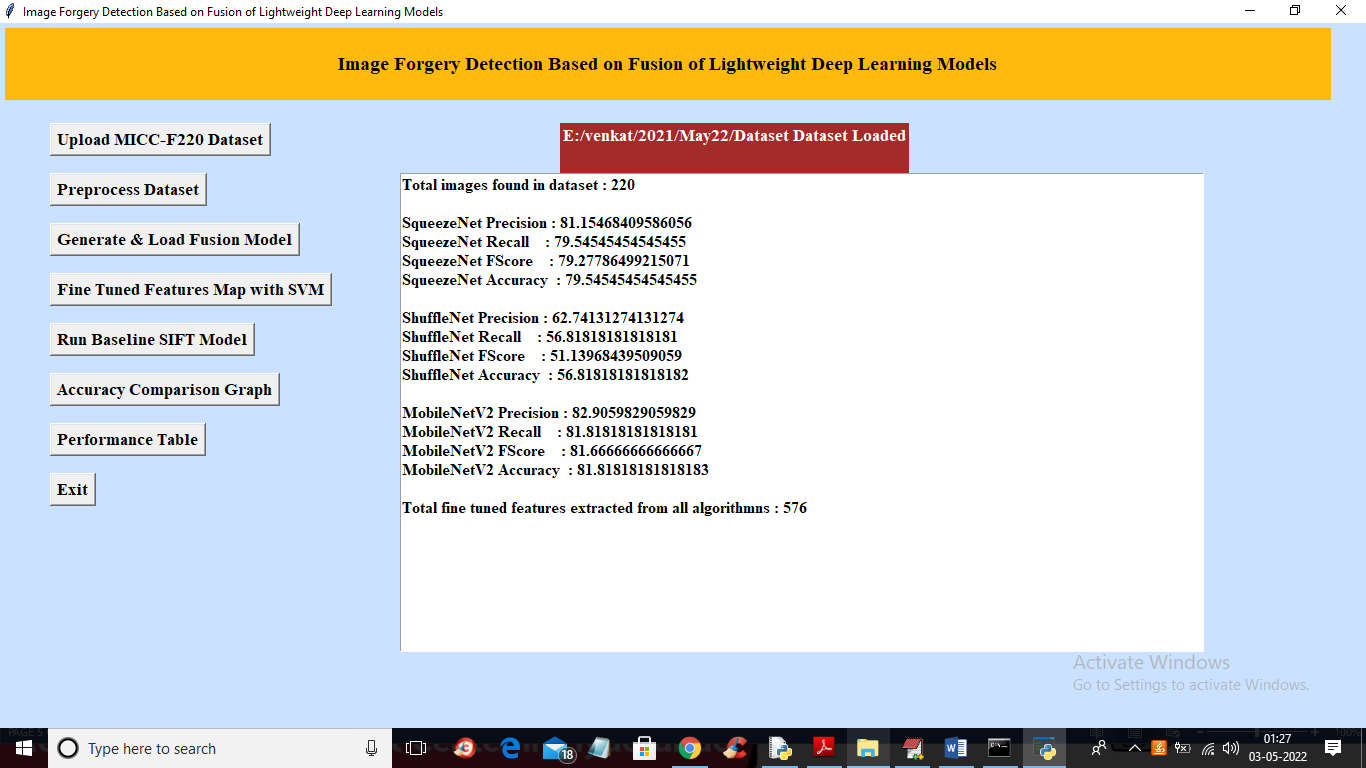
In above screen dataset loaded and now click on ‘Preprocess Dataset’ button to read all images and normalize them and get below output



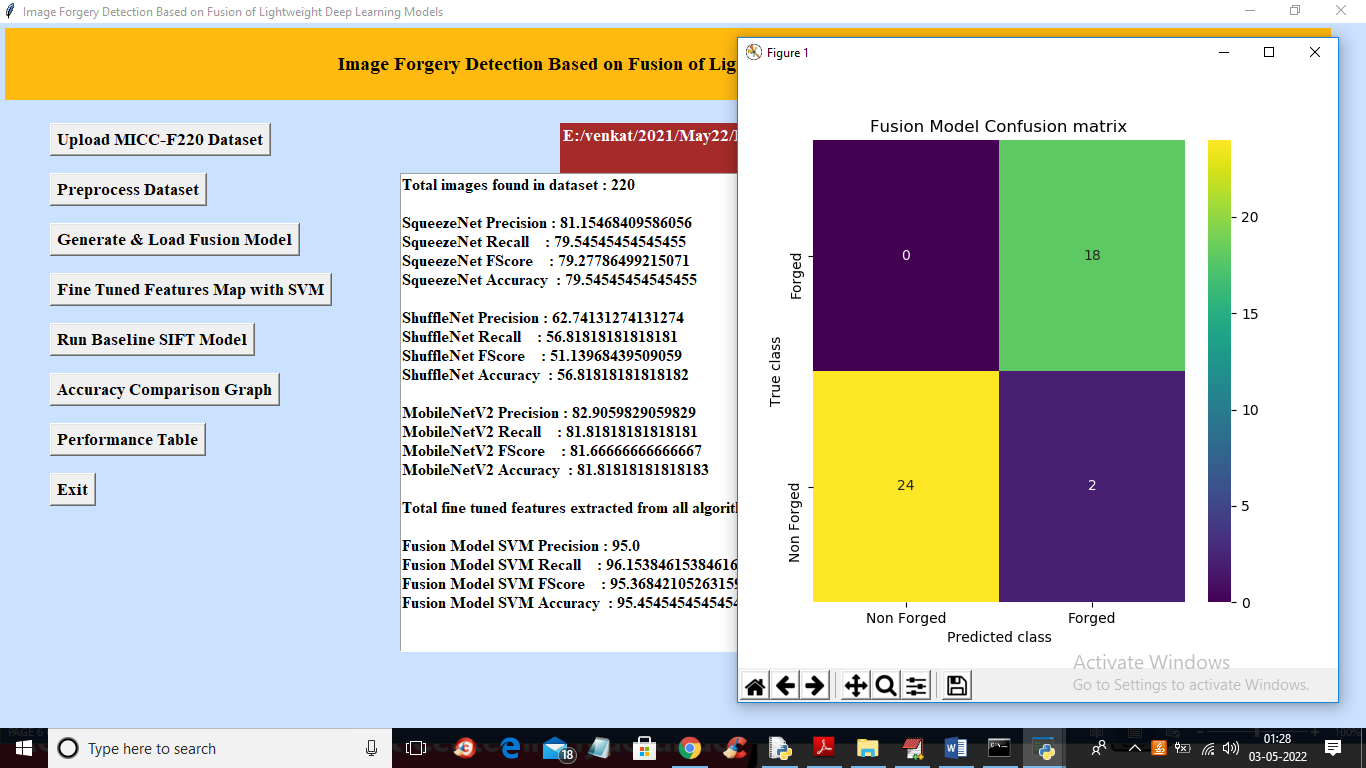
In above screen all images are processed and to check images loaded properly I am displaying one sample image and now close above image to get below output



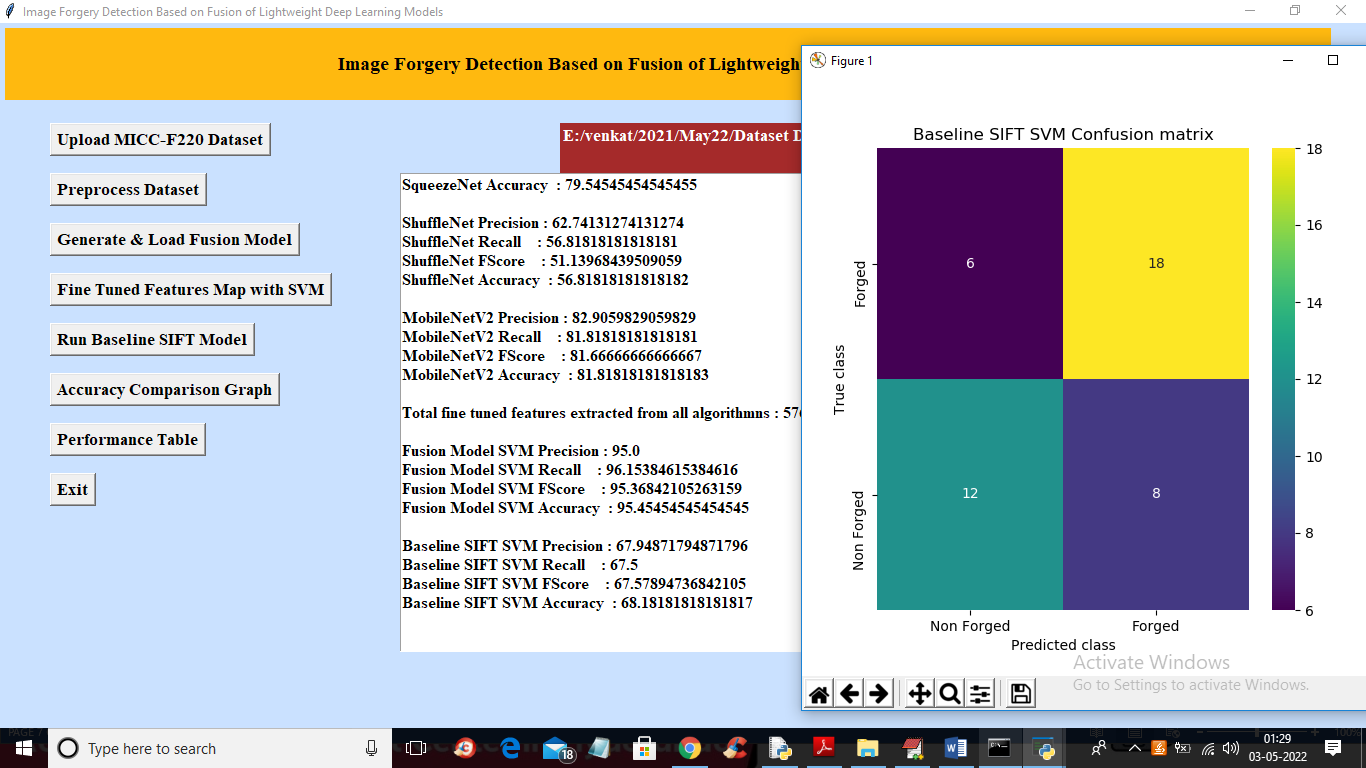
In above screen we can see dataset contains 220 images and all images are processed and now click on ‘Generate & Load Fusion Model’ button to train all algorithms and then extract features from them and then calculate their accuracy



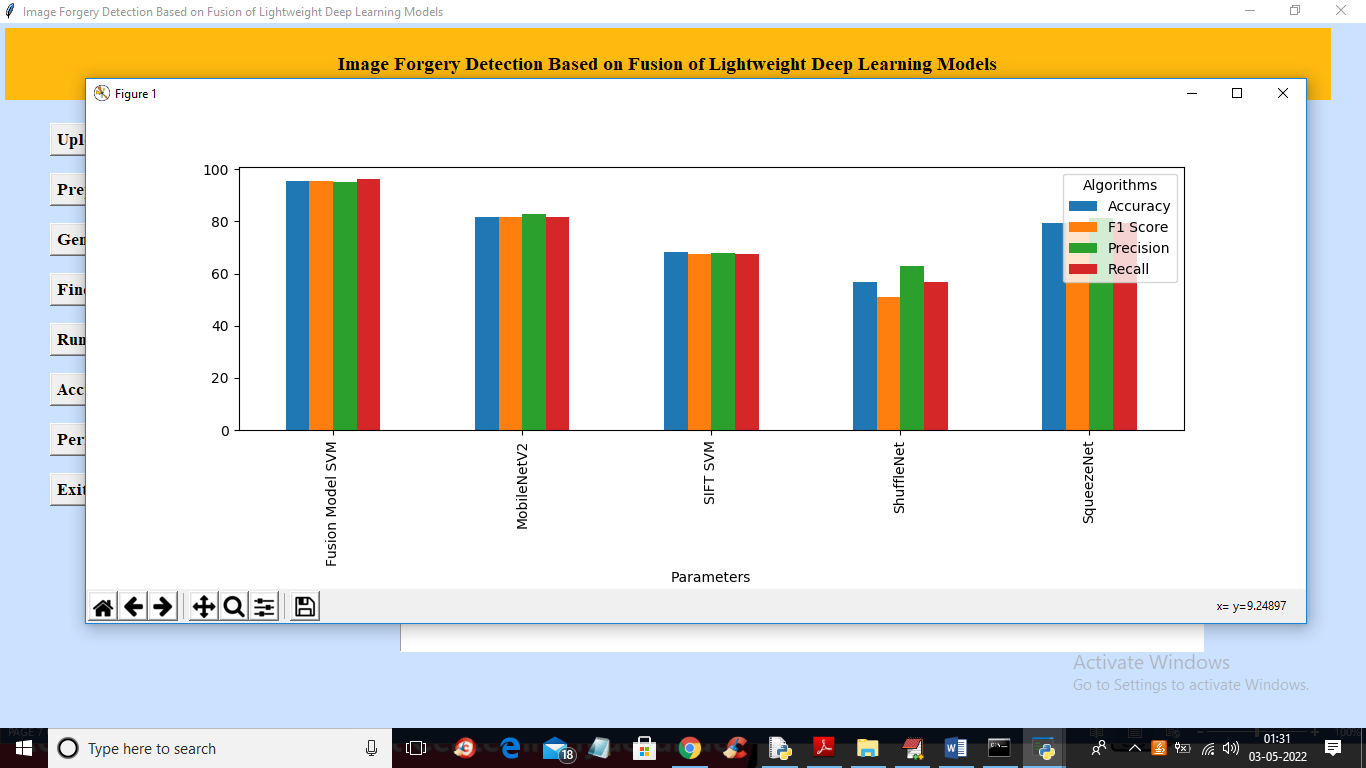
In above screen we can see accuracy of all 3 algorithms and then in last line we can see from all 3 algorithms application extracted 576 features and now click on ‘Fine Tuned Features Map with SVM’ to train SVM with extracted features and get its accuracy as fusion model



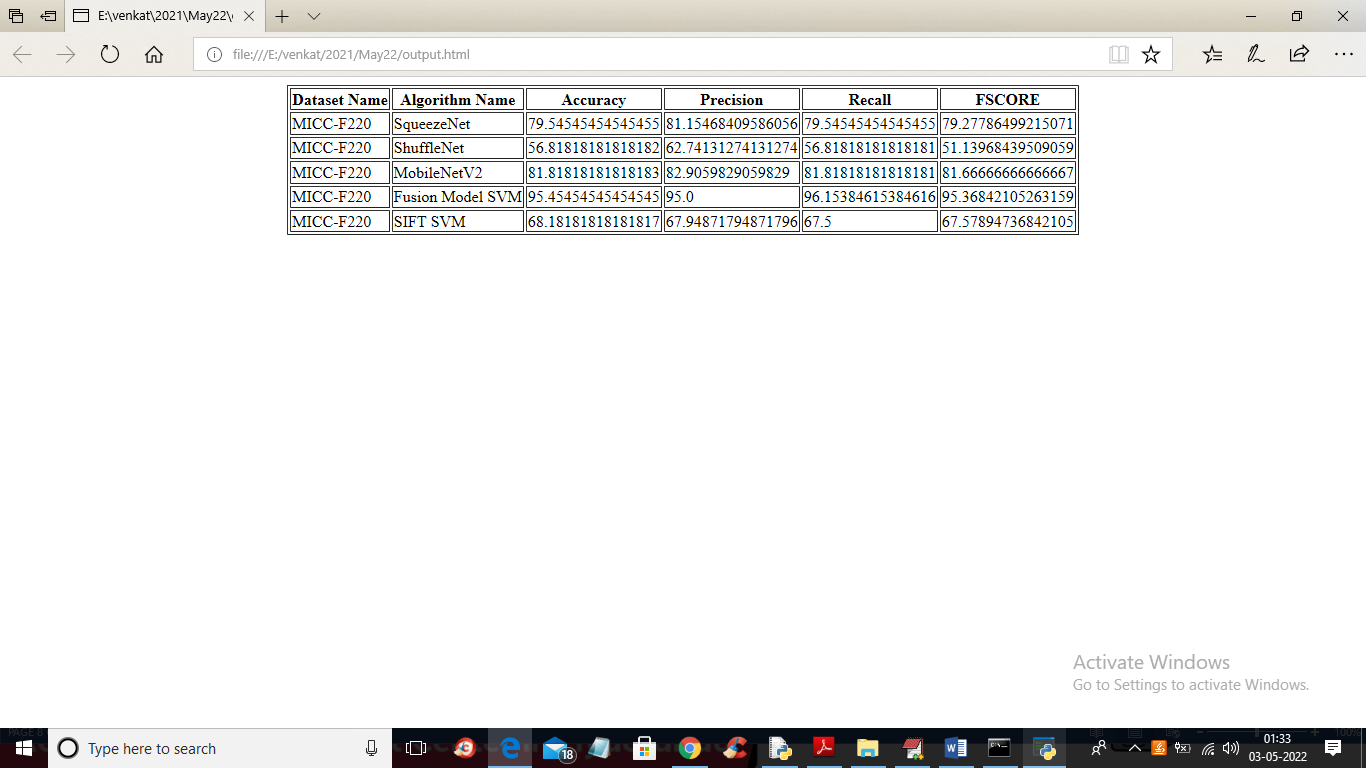
In above screen with Fine tune SVM fusion model we got 95% accuracy and in confusion matrix graph x-axis represents PREDICTED LABELS and y-axis represent TRUE labels and we can see both X and Y boxes contains more number of correctly prediction classes. In all algorithms we can see fine tune features 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



In above screen with existing SIFT SVM features we got 68% accuracy and in confusion matrix graph we can see existing SIFT predicted 6 and 8 instances incorrectly. So we can say existing SIFT features are not good in prediction and now close above graph and then click on ‘Accuracy Comparison Graph’ button to get below graph



In above graph x-axis represents algorithm names and y-axis represents accuracy and other metrics where each different colour bar represents different metrics like precision, recall etc. Now close above graph and then click on ‘Performance Table’ button to get result in below tabular format



In above screen we can see propose fusion model SVM with fine tune features has got 95% accuracy which is better than all other algorithms

**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.

**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