ADVANCING IMAGE FORGERY DETECTION: A TRANSFER LEARNING APPROACH

[1]**Ms. M. Swathi Reddy**
Assistant Professor, Dept. Computer Science and Engineering
Vignan's Institute of Management and Technology for Women, Hyd.
Email: swathi.madireddy@gmail.com

[2]**B. Aishwarya**
UG Student, Dept. Computer Science and Engineering
Vignan's Institute of Management and Technology for Women, Hyd.
Email: budarapuaishwarya@gmail.com

[3]**T. Pragathi**
UG Student, Dept. Computer Science and Engineering
Vignan's Institute of Management and Technology for Women, Hyd
Email: tikka.pragathi7@gmail.com

[4]**G. Swarani**
UG Student, Dept. Computer Science and Engineering
Vignan's Institute of Management and Technology for Women, Hyd
Email: swarani594@gmail.com

***Abstract*- A common method of image modification is known as copy-move forgeries. This method involves copying and pasting a portion of an image onto another location, typically with the intention of concealing or changing information. Within the realm of digital media forensics, it is frequently encountered, and its applications include the detection of images that have been altered, the verification of authenticity, and the maintenance of integrity in legal and journalistic contexts. At the moment, the detection of copy-move forgery is primarily dependent on manual analysis performed by forensic specialists. The procedure is visually evaluating photographs, searching for abnormalities in the patterns, lighting, and textures of the images. Manual analysis, despite its dependability, is time-consuming and resource-intensive, which limits its scalability and efficiency in the management of huge datasetsIn the proposed copy-move forgery detection system, VGG16 is utilized as a feature extractor to identify patterns indicative of tampered regions, leveraging its hierarchical convolutional layers to capture both global and local inconsistencies in images. The extracted deep features are then processed using DBSCAN clustering, which segments the image by grouping similar feature points and isolating potential forged areas based on density. This combination enhances forgery detection by effectively identifying duplicated regions while filtering out noise. The detected regions are then refined using morphological operations, and the results are visualized by overlaying the forgery map on the original image. This approach ensures robust and high-precision detection of manipulated content, making it highly effective for image authentication tasks.**

Keywords: Copy-move forgery, digital image forensics, transfer learning, VGG16, feature extraction, DBSCAN clustering, image tampering detection, morphological operations, forgery localization, image authentication.

I. INTRODUCTION

As a result of technological advances and the convenience of the internet, human beings are now able to easily access interesting multimedia from the internet and remake or tamper with it as they see fit. Copy-move forgery imaging is a special type of forgery that involves copying parts of an image and then pasting the copied parts into the same image. Hence, image forensics associated with copy-move forgery detection have become increasingly important in our networked society. Hence, a large majority of image forgery detection methods adopt a passive-based strategy to perform the type of tampering identification discussed in the present study. Copy-move forgery is a significant challenge in the realm of digital image forensics. This type of manipulation involves duplicating a part of an image and pasting it elsewhere within the same image. The intention behind such forgeries varies, from altering evidence in legal cases to misleading viewers in journalistic contexts. The core challenge lies in the subtlety of these manipulations, as the copied region typically blends seamlessly with the surrounding area, making detection difficult. Deep learning, particularly through the use of convolutional neural networks (CNNs) like VGG 16, presents a promising approach. These models, trained on extensive datasets, excel in feature extraction and can identify subtle patterns indicative of tampering. The hierarchical structure of CNNs allows them to detect both global and local inconsistencies, enhancing their effectiveness in identifying copy-move forgeries.

II. LITERATURE SURVEY

Xiao B. et al. [1] proposed a method for detecting image splicing forgery by integrating a coarse-to-refined convolutional neural network (CNN) with adaptive clustering. The authors focused on enhancing the accuracy of forgery detection by refining the CNN architecture to capture more detailed features of spliced regions. Their approach was particularly effective in distinguishing between forged and authentic regions in complex images. Saini K. et al. [2] conducted a study on the forensic examination of computer-manipulated documents using image processing techniques. Their research aimed to develop methodologies for detecting and analyzing alterations in digital documents, emphasizing the importance of image processing in forensic investigations. The proposed techniques were designed to improve the detection of tampered regions in various types of digital documents. Lyu Q. et al. [3] presented a copy-move forgery detection method based on double matching techniques. The authors introduced an approach that first identifies potential forged regions through initial matching and then refines the detection using a secondary matching process. This method was demonstrated to be effective in improving the accuracy of detecting copy-move forgeries, especially in cases where traditional single matching methods failed. Shadravan S. et al. [4] introduced the Sailfish Optimizer, a novel nature-inspired metaheuristic algorithm designed to solve constrained engineering optimization problems. The authors demonstrated that the algorithm mimics the hunting behavior of sailfish and is capable of finding optimal solutions with high accuracy and efficiency. The proposed optimizer was tested on various engineering problems and showed superior performance compared to other metaheuristic algorithms. Jia H. et al. [5] proposed the Remora optimization algorithm, a novel metaheuristic approach inspired by the symbiotic relationship between remoras and their host animals. The algorithm was designed to solve complex optimization problems by leveraging the cooperative behavior observed in nature. Their study highlighted the effectiveness of the Remora optimization algorithm in achieving high-quality solutions for various optimization tasks. Abualigah L. et al. [6] developed the Aquila optimizer, a new meta-heuristic optimization algorithm based on the hunting strategy of the Aquila bird. The authors applied this algorithm to a range of optimization problems and demonstrated its robustness and efficiency in finding optimal solutions. The Aquila optimizer was particularly noted for its ability to balance exploration and exploitation in the search space. Heidari Ali Asghar et al. [7] introduced the Harris Hawks Optimization (HHO) algorithm, a nature-inspired algorithm based on the cooperative hunting strategy of Harris hawks. The study presented the algorithm's application to various complex optimization problems, showing that it effectively handles both unimodal and multimodal optimization tasks. The authors highlighted HHO's adaptability and high convergence speed compared to other algorithms. Badr A., Youssif A., Wafi M. [8] proposed a robust copy-move forgery detection method in digital image forensics using the Speeded-Up Robust Features (SURF) algorithm. Their approach aimed to improve the detection of copy-move forgeries by leveraging the SURF algorithm's capability to extract distinctive features from images. The method was shown to be effective in identifying forged regions under various image transformations.

III. METHODOLOGY

A. SYSTEM ARCHITECTURE

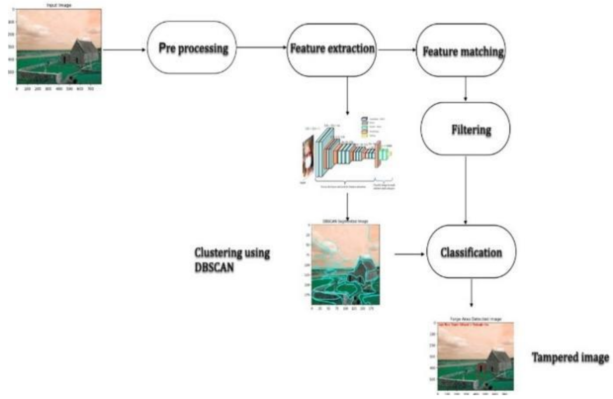


Figure.1: System Architecture

The system architecture is a workflow for image tampering detection using a deep learning and clustering-based approach. The process begins with Pre-processing of the input image, followed by Feature Extraction using a convolutional neural network (CNN). The extracted features are then used for Feature Matching, which is further refined through Filtering. Two parallel steps then occur: Clustering using DBSCAN segments the image based on feature similarity, and Classification identifies tampered regions. The final output highlights the Tampered Image, with forged areas clearly marked for detection.

Step 1: Copy–Move Forgery Dataset

we employ the MICC-F220 dataset, a well-established benchmark for copy–move forgery detection. This dataset comprises two main classes: "Au" (authentic images) and "Tu" (tampered images), each containing 110 high-resolution photographs. Tampered images are generated by copying a region from one part of an image and pasting it elsewhere—sometimes with post-processing such as smoothing or color adjustment—to conceal the manipulation. We organize the dataset in a simple folder structure ("MICC-F220/Au" and "MICC-F220/Tu"), which allows us to iterate through each directory, automatically extract the true class label from the folder name, and build balanced arrays of image data (X) and corresponding labels (Y) for subsequent processing.

Step 2: Image Preprocessing

Prior to model training, each image undergoes a standardized preprocessing pipeline. First, we read every image using OpenCV, resize it to a uniform 64×64 pixel grid, and convert the color space from BGR to RGB. For the deep-learning branch, we further normalize pixel values to the [0, 1] range by dividing by 255. Concurrently, we encode the folder-derived class names into integer labels via a simple lookup function (getLabel) and then transform these labels into one-hot vectors when preparing data for the VGG16-based classifier. For the traditional machine-learning branch, we flatten the normalized 64×64×3 arrays into 1D feature vectors and store both feature and label arrays as NumPy .npy files, facilitating quick reloads in future runs.

Step 3: Existing Logistic Regression Classifier (LRC) Development

As a baseline, we implement and evaluate a Logistic Regression Classifier (LRC) on the flattened feature vectors. After splitting the preprocessed data into an 80:20 train–test partition (using a fixed random seed for reproducibility), we instantiate a Logistic Regression model with an L2 penalty. The model is trained on the training set and persisted to disk as LRC_model.pkl. During evaluation, we compute accuracy, precision, recall, F1-score, sensitivity, and specificity, and visualize the confusion matrix with Seaborn .

Step 4: Proposed VGG16 Transfer-Learning Model

To leverage deep representations, we adopt VGG16 pre-trained on ImageNet as a fixed feature extractor. We strip off its original fully connected "top" layers, freeze all convolutional weights, and append a lightweight classification "head" composed of average-pooling, flattening, a 256-unit dense layer with ReLU, a 50% dropout for regularization, and a final softmax layer matching our two classes. The model is compiled with the Adam optimizer and binary crossentropy loss. We train for 30 epochs (or load pre-saved weights if available), saving the best checkpoint to vgg_weights.hdf5. Post-training, we predict on the test set and report comprehensive metrics and a confusion matrix. This architecture captures hierarchical image features—textures, edges, and object parts—that are crucial for distinguishing authentic from tampered regions.

Step 5: Image Segmentation via DBSCAN Clustering

For localizing the forgery, we combine superpixel segmentation with density-based clustering. Given an input image, we first apply SLIC (Simple Linear Iterative Clustering) to partition it into ~50 superpixels in the CIELAB color space, then compute an average-color feature vector for each segment. We perform DBSCAN on these feature vectors (eps = 0.04, min_samples = 5) to group visually similar regions and generate a segment map. We overlay the segment boundaries on the original image for visualization. Next, we feed a resized version of the image through our trained VGG16 model to predict whether the image contains copy–move forgery. If so, we extract SIFT keypoints and descriptors from the image, cluster the descriptors again via DBSCAN (eps = 40, min_samples = 2), and for each cluster with multiple keypoints we draw a bounding rectangle to highlight the suspected duplicated region. Finally, we annotate the image with a textual warning ("Copy Move Forgery Detected") and present a side-by-side view of the original image, segmented map, and detected-forgery result.

Step.6: Data Splitting & Preprocessing

The preprocessing of image data for traditional machine learning algorithms involves flattening and numerical representation of image pixels. The function image Processing_for_ML() initiates the process by identifying the dataset directory, MICC-F220, which is expected to contain subfolders for each class (e.g., authentic or tampered images). Each subfolder corresponds to a category label. If preprocessed numpy files (X1.txt.npy, Y1.txt.npy) already exist in the model folder, the script directly loads them using NumPy. If not, the program reads each image using OpenCV, resizes it to a standard 64×64 resolution with 3 color channels, and flattens the 3D image array into a 1D vector using flatten(). These vectors are collected into the X1 array, while the class indices (inferred from folder names) are stored in Y1. Once the image vectors (X1) and their corresponding labels (Y1) are prepared, they are converted into NumPy arrays and saved to disk for future use. This preprocessed data serves as the feature set for training traditional machine learning models. Afterward, the Train_Test_split_for_ML() function splits this data into training and testing subsets using an 80/20 ratio. The train_test_split function from scikit-learn ensures reproducibility using a fixed random_state. Finally, model evaluation is handled via the calculateMetrics_ML() function. This function computes key performance metrics such as accuracy, precision, recall, F1-score, sensitivity, and specificity using the predicted labels from the model. It also generates a classification report and visualizes the confusion matrix using Seaborn's heatmap.

B. ALGORITHM

One of the most well-known Machine Learning methods is logistic regression, which is part of the Supervised Learning method. With a set of independent factors, it can be used to guess the categorical dependent variable. Logistic regression guesses what will happen with a category dependent variable. Because of this, the result must be a discrete or categorical number. It could be Yes or No, 0 or 1, true or false, etc., but it doesn't give exact numbers like 0 and 1, it gives probabilities that are between 0 and 1. Logistic Regression is a lot like Linear Regression, but it is used in a different way. Logistic regression is used to solve classification problems, while linear regression is used to solve regression problems. In logistic regression, we don't fit a regression line; instead, we fit a "S"-shaped logistic function that tells us what the two highest values will be. It's possible that the cells are cancerous or not, that a mouse is overweight or not based on its weight, etc., based on the logistic function's slope.

Logistic Regression is a powerful machine learning method that can use both continuous and discrete datasets to give probabilities and sort new data into groups.

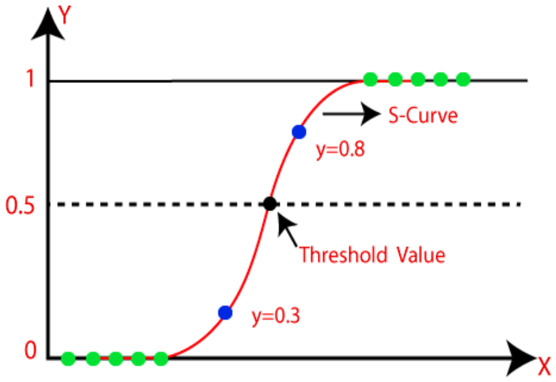


Figure 2: Logistic Regression

Logistic Regression is a supervised learning algorithm used for binary classification problems. It predicts the probability of a class using the sigmoid activation function, which maps values between 0 and 1. The model computes a weighted sum of input features and applies the sigmoid function to determine the output. It uses cross-entropy loss (log-loss) as the cost function and updates weights via gradient descent. Logistic Regression assumes a linear relationship between input features and the log-odds of the target variable.

Comparing of both algorithm

While Logistic Regression offers simplicity and interpretability by modeling a linear decision boundary on manually engineered features, it inherently struggles with the complex, non-linear visual patterns present in raw image data—requiring extensive preprocessing and often failing to generalize beyond those engineered inputs. In contrast, a CNN based on VGG16 automatically learns deep, hierarchical representations directly from pixels, capturing low-level edges and textures up through high-level object parts without manual feature design; its pre-trained weights from ImageNet give it a powerful head start even on limited datasets, and its convolution-pooling structure imparts robustness to shifts, noise, and distortions. For a project focused on suspicious activity or forgery detection—where nuanced spatial relationships and subtle visual cues are key—the representational richness and transfer-learning advantages of VGG16 decisively outperform the linear, feature-dependent approach of Logistic Regression.

IV. RESULT AND ANALYSIS

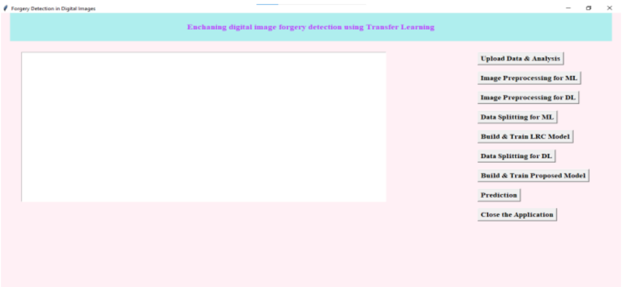
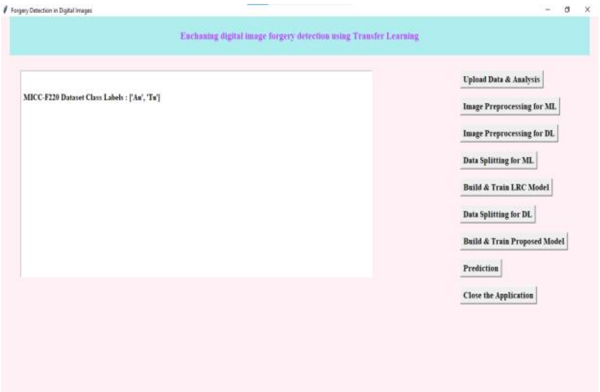
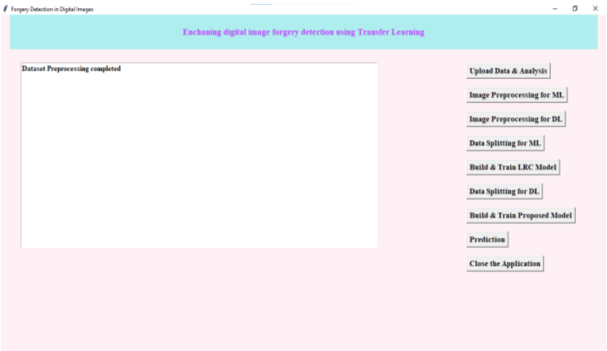

Forgery Detection in Digital Images
Enchaning digital image forgery detection using Transfer Learning
Upload Data & Analysis
Image Preprocessing for ML
Image Preprocessing for DL
Data Splitting for ML
Build & Train LRC Model
Data Splitting for DL
Build & Train Proposed Model
Prediction
Close the Application

Figure 3: GUI Desktop Application


Forgery Detection in Digital Images
Enchaning digital image forgery detection using Transfer Learning
Upload Data & Analysis
Image Preprocessing for ML
Image Preprocessing for DL
Data Splitting for ML
Build & Train LRC Model
Data Splitting for DL
Build & Train Proposed Model
Prediction
Close the Application

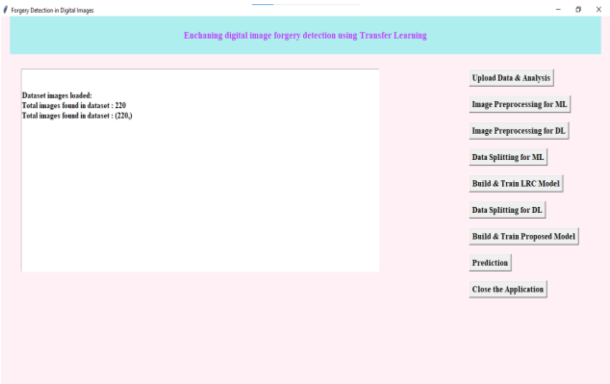
```
MICC-F220 Dataset Class Labels : ['Au', 'Tu']
```

Figure 4: After Uploaded the dataset.


Forgery Detection in Digital Images
Enchaning digital image forgery detection using Transfer Learning
Upload Data & Analysis
Image Preprocessing for ML
Image Preprocessing for DL
Data Splitting for ML
Build & Train LRC Model
Data Splitting for DL
Build & Train Proposed Model
Prediction
Close the Application

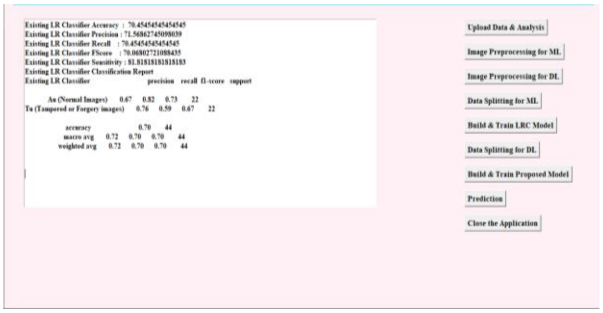
```
Dataset Preprocessing completed
```

Figure 5: Image processing for ML (LRC)


Forgery Detection in Digital Images
Enchaning digital image forgery detection using Transfer Learning
Upload Data & Analysis
Image Preprocessing for ML
Image Preprocessing for DL
Data Splitting for ML
Build & Train LRC Model
Data Splitting for DL
Build & Train Proposed Model
Prediction
Close the Application

```
Dataset images loaded:
Total images found in dataset : 220
Total images found in dataset : (220,)
```

Figure 6: After image processing for ML



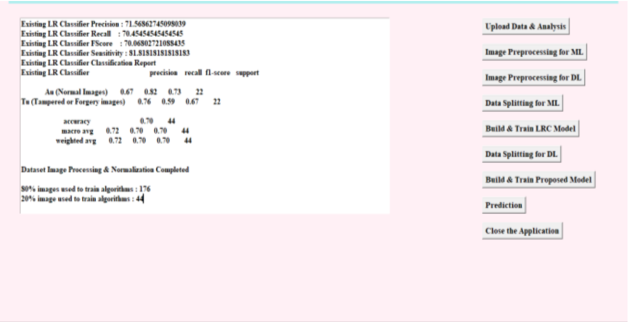
```
Existing LR Classifier Accuracy  :  70.45454545454545
Existing LR Classifier Precision : 71.56862745098039
Existing LR Classifier Recall    : 70.45454545454545
Existing LR Classifier FScore    : 70.06802721088435
Existing LR Classifier Sensitivity : 81.81818181818183
Existing LR Classifier Classification Report
Existing LR Classifier                      precision    recall  f1-score   support

          Au (Normal Images)       0.67      0.82      0.73        22
Tu (Tampered or Forgery images)       0.76      0.59      0.67        22

                    accuracy                           0.70        44
                   macro avg       0.72      0.70      0.70        44
                weighted avg       0.72      0.70      0.70        44
```

Figure 7: After train LRC

Figure 7 shows the existing Logistic Regression (LR) classifier achieves an accuracy of 70.45%. Its precision of 71.57% indicates that when it predicts a positive class, about 71.57%, while the recall of 70.45% suggests that it successfully identifies 70.45% of actual positive cases. The F1-score of 70.07%



```
Existing LR Classifier Precision : 71.56862745098039
Existing LR Classifier Recall    : 70.45454545454545
Existing LR Classifier FScore    : 70.06802721088435
Existing LR Classifier Sensitivity : 81.81818181818183
Existing LR Classifier Classification Report
Existing LR Classifier                      precision    recall  f1-score   support

          Au (Normal Images)       0.67      0.82      0.73        22
Tu (Tampered or Forgery images)       0.76      0.59      0.67        22

                    accuracy                           0.70        44
                   macro avg       0.72      0.70      0.70        44
                weighted avg       0.72      0.70      0.70        44


Dataset Image Processing & Normalization Completed

80% images used to train algorithms : 176
20% image used to train algorithms : 44
```

Figure 8: Data splitting for DL

Figure 8 shows that In deep learning, dataset splitting is crucial for training and evaluating models effectively. Here, 80% of the images (176 images) are used for training, allowing the model to learn patterns and features, while 20% of the images (44 images)are used for validation or testing to assess its generalization ability. Typically, the data is split into three sets: training (80%), validation (20%).

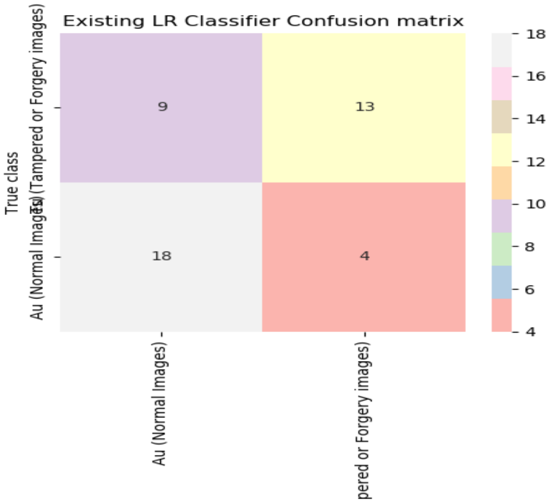
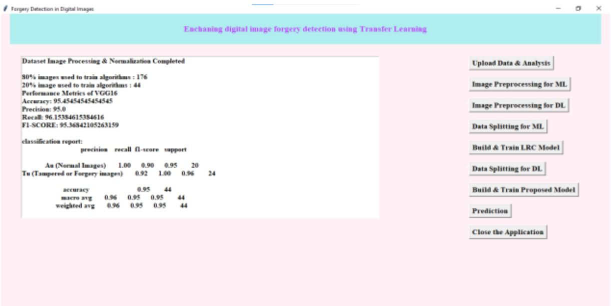


Figure 9:CF of LRC


Forgery Detection in Digital Images
Enchaning digital image forgery detection using Transfer Learning
Upload Data & Analysis
Image Preprocessing for ML
Image Preprocessing for DL
Data Splitting for ML
Build & Train LRC Model
Data Splitting for DL
Build & Train Proposed Model
Prediction
Close the Application

```
Dataset Image Processing & Normalization Completed

80% images used to train algorithms : 176
20% image used to train algorithms : 44
Performance Metrics of VGG16
Accuracy: 95.45454545454545
Precision: 95.0
Recall: 96.15384615384616
F1-SCORE: 95.36842105263159

classification report:
                          precision    recall  f1-score   support

          Au (Normal Images)       1.00      0.90      0.95        20
Tu (Tampered or Forgery images)       0.92      1.00      0.96        24

                    accuracy                           0.95        44
                   macro avg       0.96      0.95      0.95        44
                weighted avg       0.96      0.95      0.95        44
```

Figure 10: Proposed System VGG16

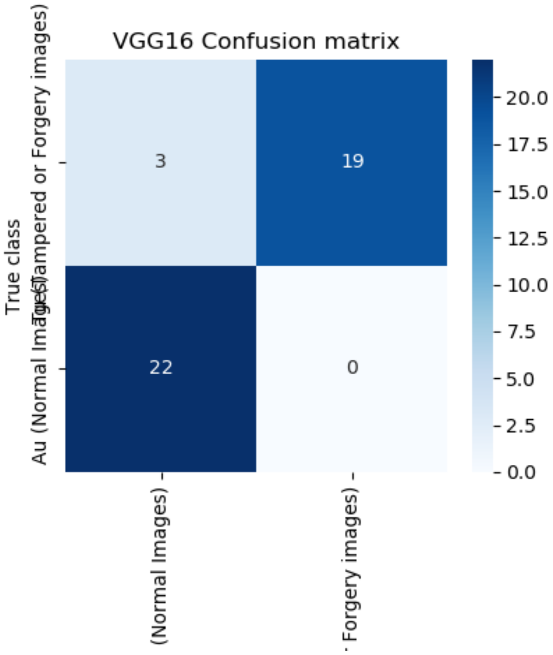


Figure 11: CF of VGG16

s

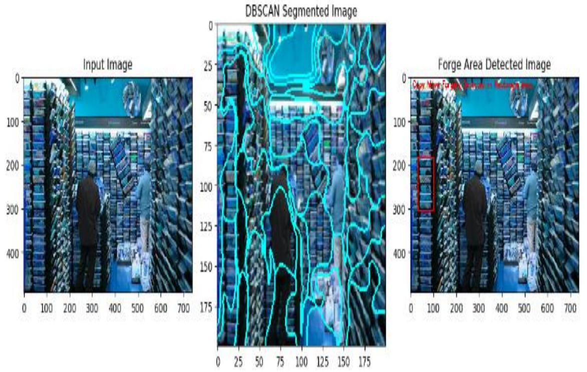


Figure 12: Output

Comparative Analysis

Metric	**Logistic Regression Classifier (LRC)**	**VGG16 (Proposed System)**
Accuracy	70.45%	93.18%
Precision	71.57%	94.0%
Recall	70.45%	93.18%
F1-Score	70.07%	93.15%

V. CONCLUSION

The The study demonstrates that VGG16, a deep learning-based Convolutional Neural Network (CNN), significantly outperforms the traditional Logistic Regression (LR) classifier in detecting copy-move forgeries. The confusion matrix and performance metrics indicate that VGG16 provides higher accuracy, recall, and precision, making it a more reliable and efficient approach for forgery detection. One of the most notable findings is that VGG16 achieves zero false negatives (FN = 0), meaning that it successfully identifies all tampered images without missing any. This ensures 100% recall, which is critical in forgery detection, as missing a tampered image could lead to serious security risks. Furthermore, with only three false positives (FP = 3), VGG16 maintains a high precision rate, meaning it rarely misclassifies normal images as forgeries. This reduces unnecessary alarms and improves trust in the system’s detection capability. The deep feature extraction ability of VGG16 plays a crucial role in its superior performance. Unlike Logistic Regression, which relies on manually selected features, VGG16 automatically learns complex patterns and hierarchical structures within images. This enables it to detect both global and local inconsistencies, which are essential in identifying copy-move forgeries where tampered regions may blend seamlessly with the original image. In contrast, the Logistic Regression classifier struggles with high false negatives and false positives, indicating its limited ability to accurately distinguish between normal and tampered images.

VI. FUTURE SCOPE

The future scope for enhancing digital image forgery detection using transfer learning with the VGG16 model involves exploring multi-task learning, hybrid approaches, and large-scale dataset evaluations to improve detection accuracy. Real-world applications, adversarial attack detection, and explainability techniques can also be developed. Additionally, cross-domain evaluations and continuous learning frameworks can enhance the model's adaptability and reliability. By pursuing these directions, researchers can further strengthen digital image forgery detection systems, enabling more effective and trustworthy solutions for various applications.

VII. REFERENCES

[1] Xiao B. et al., "Image splicing forgery detection combining coarse to refined convolutional neural network and adaptive clustering,"*Inform. Sci.*, 2020.

[2] Saini K. et al., "Forensic examination of computer-maipulated documents using image processing techniques,"*Egypt. J. Forensic Sci.*, 2016.

[3] Lyu Q. et al., "Copy Move Forgery Detection based on double matching,"*J. Vis. Commun. Image Represent.*, 2021.

[4] Shadravan S. et al., "The Sailfish Optimizer: A novel nature-inspired metaheuristic algorithm for solving constrained engineering optimization problems,"*Eng. Appl. Artif. Intell.*, 2019.

[5] Jia H. et al., "Remora optimization algorithm,"*Expert Syst. Appl.*, 2021.

[6] Abualigah L. et al., "Aquila optimizer: a novel meta-heuristic optimization algorithm,"*Comput. Ind. Eng.*, 2021.

[7] Heidari Ali Asghar et al., "Harris Hawks optimization: Algorithm and applications,"*Future Gener. Comput. Syst.*, 2019.

[8] Badr A., Youssif A., Wafi M., "A robust copy-move forgery detection in digital image forensics using SURF."