# Image Forgery Detection

**ABSTRACT:**
The prevalence of image modification software has increased due to its easy accessibility. Because the altered photos are indistinguishable with the human eye, they are spreading across multiple platforms, sparking rumors and deceiving a lot of people. This has prompted academics to develop a number of methods for more accurate image manipulation detection. The majority of traditional research on image forgery detection is focused on identifying basic traits that are unique to identifying a certain kind of fraud. Recent studies have demonstrated how highly successful neural network-based forgery detection is at spotting phony photos. Neural networks can retrieve complex hidden information from photos more precisely. Since a deep learning model automatically creates the necessary features in contrast to more conventional methods of forgery detection, it has emerged as the new field of study in image forgery

## 1 INTRODUCTION

Now a days, images are used in different sectors such as media, medical, education, sports, news, forensics etc. Images must be true and genuine as information is shared through images. But digital image manipulation takes place by using photo editing apps such as photoshop, pixlr, Adobe lightroom, etc. which mislead the image information. A human cannot tell whether an image has been altered with the unaided eye. Forging photographs canbe done for a variety of reasons, such as financial gain, rumors propagation, or making exaggerated statements to support oneself.

[10] There are two categories of methods for detecting photo tampering: (i) Active and (ii) Passive. In an active technique, an authorized individual embeds extra data (such a digital watermark) into the image either during the image collecting process or at a later time. This embedded information is utilized by the active technique to detect manipulation. The passive methods for detecting forgeries do not rely on the extra data. These methods are also known as "blind methods" since they don't make use of any extra data to identify forgeries.The passive techniques fall under the category of forgery type independenttechniques, which look for other forgeries such compression, resampling, and discrepancies.

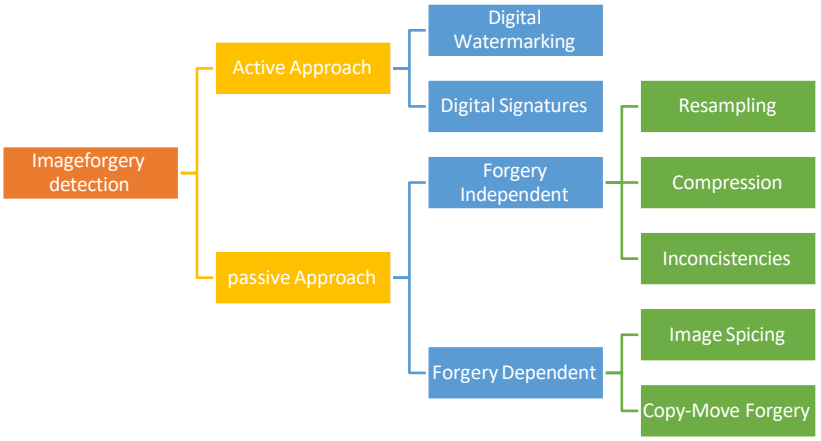



Fig 1: Classification of Image Forgery Detection Approaches

Regarding the forgery-dependent strategy, there are two categories:1) Image Splicing 2) Forgery by Copy-Move. The two distinct photos are combined in image splicing forgeries. Transfer-Copy Forging a picture is duplicating a section of it and repositioning it inside the original image at a different spot. The photos that are created using these techniques are difficult for humans to recognize.

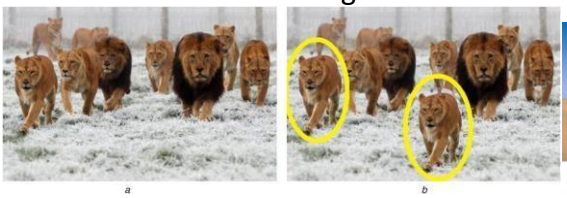


Fig 2: Copy-Move Image Forgery

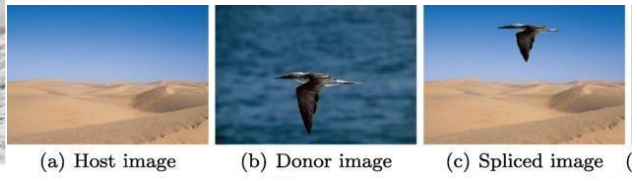



Fig 3: Image Splicing Image Forgery

Through the use of picture recompression, Ali, Syed Sadaf [11] proposed a method for identifying image forgeries that makes use of the variations in image sources.

The primary contributions of the proposed model are as follows:

- A densenet121 architecture detects picture fraud efficiently. Image recompression is used, and the model architecture considers the changes between images before and after compression.
- This technique detects copy-move forgery and image splicing, unlike other algorithms that only detect only type of forgery.

  The remainder of this paper is arranged as follows. Section 2 is a literature overview on image forgery . Section 3 contains information on the proposed model used to detect image forgeries. Section 4 covers the proposed technique and the model's comprehensive experimental results.

## 2 LITERATURE REVIEW

There are several methods for identifying image faking in this literature. There are two types of digital picture forensics: deep learning and traditional feature extraction. Most existing methods rely on specific artifacts from picture forgeries, but there are recently proposed ways based on CNNs and deep learning.

The authors of [1] proposed pixel-based image forgery detection as one of the traditional techniques for identifying passive image forgery. Pixel-based methods highlight the individual pixels in the digital picture. Format-based picture fraud detection: This method primarily relies on image formats, mostly JPEG. Camera-based picture fraud detection is predicated on the notion that the model and artifacts of the camera may affect how the image is processed from capture to memory save. The rest consist of geometry-based and physical environment-based methods for detecting image forgeries. Several techniques have been proposed in [2] that focus on copy-move picture forgery detection. These techniques can be divided into two categories: block-based and key-point based. Using thecoefficients of the quantized discrete cosine transform (DCT), [3] proposed the first copy- move forgery detection technique. Another author suggested in [4] that principal component analysis (PCA) be used to identify copy move forgeries. [5] suggested a discretewavelet transform (DWT) that makes use of the pixel matching concept to detect copy- move forgeries effectively. In [6], a number of techniques for detecting picture splicing forgeries have been developed. To identify picture splicing forgeries, feature-based techniques such as principal component analysis (PCA) and color filter array (CFA) interpolation have been discussed here [6].

Deep learning techniques have improved in efficiency for detecting image forgeries in the modern day. Recurrent neural networks (RNN), convolutional neural networks (CNN), and deep neural networks (DNN) are examples of deep learning models that operate well in identifying image frauds. The author [7] has suggested using CNN-based deep learning techniques to identify image splicing and copy-move fraud. Here [7], RGB color photos are fed into CNN, which automatically learns hierarchical representations. Here, a supervised CNN is trained to recognize the new hierarchical features of the training image. This is the fundamental implementation. For the purpose of pre-processing the module, the supervised CNN's first convolutional layer is initialized. Next, from an image containing patch-size sliding windows, features are extracted. Finally, binary classification is employed to train the SVM classifier at the layer. It produces output regardless of whether the final feature was modified or built from scratch.

Copy-move photo forgeries: Copy-move Image forgery detection was proposed as A dual domain based CNN architecture (D-CNN)for the identification of copy-moved areas in photos [8]. Sub-SCNN is the CNN subnetwork whilst Sub-FCNN represent a Fully Convolutional Neural Network (JSonNet Implementation). For that to know and point the regions with malicious intervention, they are connected by these two sub-networks. The Sub-SCNN is responsible for detecting fake images from the RGB input image. The suggested method [8] when employed with pre-trained SCNN and FCNN networks, it provided superior results along with lower preprocessing time in the training. A method for picture splicing image forgery detection was proposed by [9] that exploit local phase quantization combined with a multiscale entropy filter. Highlights borders of forged regions by means of Entropy Filter applied on the chrominance channel The proposed method requires a conversion to YCbCr-based original image. Finally, to determine whether or not an image is attacked, a histogram of each property is obtained and fed into the SVM classifier. In the next step, authors stated that this method is efficient for detecting both copy-move as well picture splicing.

## 3 PROPOSED METHODOLOGY

In the study, I aim to employ DenseNet method known for its fastness, accuracy, low error rate and fewer numbers of epochs. The proposed way consists of three essential processes which are feature extraction, classification and pre-processing. Here the images are preprocessed to ELA (Error Level Analysis). To extract deep features from ELA photos, the DenseNet121 base model is employed, which was trained on the ImageNet dataset. This is accomplished by eliminating the top layers of the DenseNet121 and use its output as features for subsequent processing The collected features are then input into further layers in the classification stage, which consists of dropout for regularization, global average pooling, a dense layer with a ReLU activation, and a softmax layer for binary classification. The model is trained on the processed ELA images to distinguish between tampered and original images.

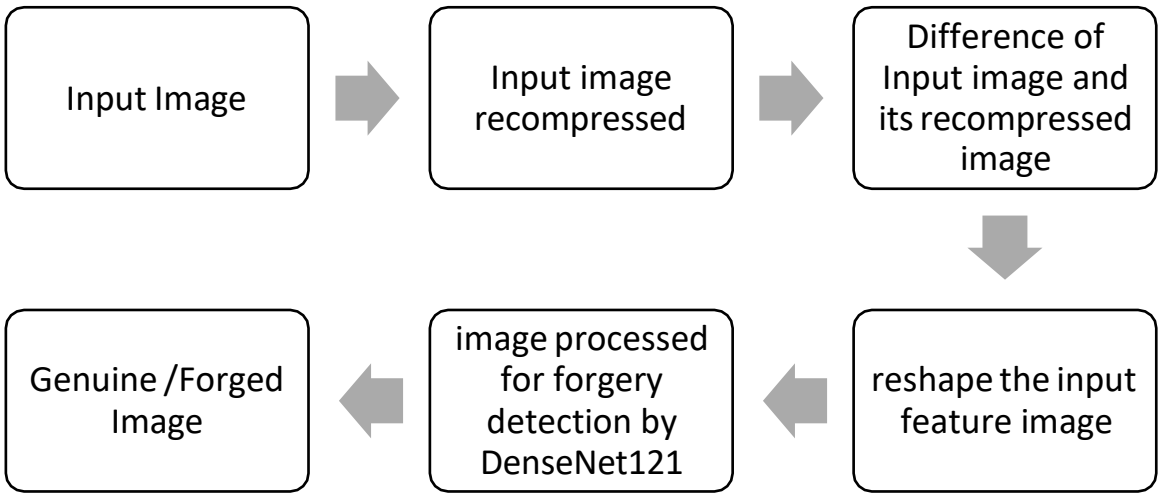



Fig 4: Flow chart of methodology used

### 3.1THE ACQUISITION OF DATA

The database's evaluation of picture tampering detection uses the publicly accessible CASIA V2.0 dataset [12]. The evaluation and comparison of tampering detection techniques are made possible by this easily accessible database for research purposes. It was designed especially to detect image forgery. There are 12,615 total photos in it, of which 7,492 are Authentic and 5,123 are tampered with.

Table 1: Insights of the CASIA v2.0 dataset

| | Authenticated Images | Tampered Images | Total Images |
|---|---|---|---|
| CASIA 2.0 | 7492 | 5123 | 12,615 |
| Training (80%) | 5994 | 4098 | 10,092 |
| Testing (20%) | 1498 | 1025 | 2523 |

Image format: JPEG, BMP, TIFF

Dimensions : From 320x240 to 800x600 color images.

### 3.2DATA PREPROCESSING

As far as we are aware, there are some images in the CASIA v.2.0 dataset. This collection contains images with sizes ranging from 320x240 to 800x600. CASIA v.2.0's preprocessing and training are based on the various sources of altered regions and backdrop photos. The forged image is improved differently when we recompress such images because of the compression difference. The foundation of this idea is error-level analysis (ELA). converting photos to the ELA format in order to emphasize the distinctionsbetween the tampered and original parts.

If a photograph contains a counterfeit, the forged piece will compress differently from the rest of the image when recompressed, as the source differs from the original.This significantly highlights the counterfeitcomponent when comparing the original image to its recompressed counterpart. We thususe it to train our dense net-based image forgery detection model. Algorithm 1 provides this description of how the suggested technique works. The forged image A gets recompressed; we will call this new image Arecompressed. We now calculate the difference, or Adiff, between the original and recompressed pictures. Since the created area's source is different from the image's original component, it is now highlighted in Adiff. We identify Adiff as a prominent image and utilize it as our input feature to train a dense net-based network to distinguish between real and faked images.
We employ JPEG compression to create Arecompressed from A. JPEG compression is applied toImage A, resulting in Arecompressed. The pattern depicted in the fabricated portion of the image when there is only one compression, according to the histogram of the dequantized coefficients. After being scaled to 128 × 128 (Areshaped_diff), the feature image (Adiff) is fed intothe network. The feature images are how the network finds out if there has been any tampering. As it is trained, the suggested model discovers several artifacts left over from image fabrication that indicate the existence of the counterfeit. After JPEG compression is done, the dataset is divided into training and testing, with 80 percent train and 20 percent test.

### 3.3 EXISTING DENSENET ARCHITECTURE

The starting point for visual object detection, the reliable Dense Net architecture [13] is renowned for offering top-notch performance with fewer parameters. It's closely resembles Res Net with a few significant changes. ReNet mixes levels that come before and after using an additive attribute, whereas Dense Net adds a concatenation feature to combine the output of past and following layers [11, 12]. Conventional CNNs calculate the output of the lth layer by applying a nonlinear transformation denoted as Hl(.) to the previous layer's output.

xl = Hl (xl-1)  (1)

Dense Net deviates from the conventional method of concatenating layer output feature maps to inputs. Dense Net's unique approach offers a seamless paradigm of communication, enhancing the cross-layer flow of information. Specifically, the lth layer's inputs come from all of the layers that came before it:

xl = Hl ([[x0, x1, x ] l-1])  (2)

Concatenating the output maps from the previous layers yields the tensor [x0, x1, x2,... xl-1]. [10]. hl(.) is a non-linear transformation function. This function consists of three basic processes: batch normalization (BN), pooling and convolution combined, and activation (ReLU). The growth rate, represented by k, is essential to the lth layer's generalization in the following ways:

kl = k0+k(l-1)

where the array of the channels is denoted by k0.

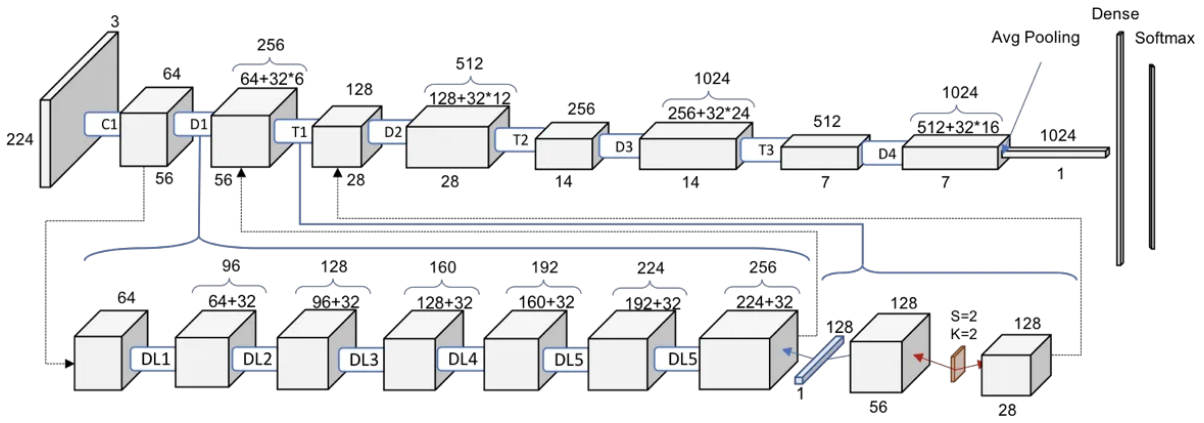



Figure-5

Figure-5 shows the model architecture of the existing DenseNet-121 network.

## 3.4 PROPOSED DENSENET ARCHITECTURE FOR IMAGE FORGERY DETECTION

In this work, we performed the picture classification job using an advanced convolutional neural network, also called a CNN, architecture. The central component of our methodology is the integration of a pre-trained DenseNet-121 model for feature extraction procedures. We present a thorough description of our model design below, beginning with the DenseNet-121 model's initialization and continuing through to the output layer's conclusion

Our suggested model is predicated on the DenseNet-121 network, as was previously stated. The network's proficiency in image categorization tasks is widely recognized. We initialized the DenseNet-121 model in our implementation using the weights that were previously trained from ImageNet for the first layer of our model. This implementation enables the use of transfer learning efficiently. To obtain the output layer of the network for our task, we disabled the top fully connected layers of the DenseNet-121 model. Given that the input form is defined as (128,128,3), the model is designed to process images with three RGB color channels that have a resolution of128x128 pixels. Then, for feature extraction, we added the DenseNet-121 base model because of its dense connective pattern; each layer receives inputs from preceding layers,increasing the gradient flow and enabling the reuse of features. After initializing the DenseNet-121 base model, we constructed the overall model using a sequential model. Thesequential model allows for the linear stacking of layers, facilitating straightforward construction and modification of the network.

For the first layer in our proposed sequential model is the pre-trained DenseNet-121 base model, ensuring robust feature extraction capabilities right from the beginning. This layer model processes the input images and outputs a high dimensional feature map that captures the essential characteristics of the images required for the classification taskWe added a GAP layer to move from the layer that uses convolution to the layer that is entirely connected. This global average pooling (GAP) layer enables the transformation of feature maps into asingle vector per feature map, reducing the output size while retaining important spatial information. Global Average Pooling can help reduce overfitting by significantly reducing the number of parameters. A completely connected (dense) layer with 256 units was introduced after the Global Average Pooling (GAP) layer. (ReLU) activation function for this layer. ReLU was chosen due to its excellent computational efficiency and ability to mitigate the vanishing gradient problem. The final layer of our proposed model is a two-unit dense layer that represents the two classes in our binary classification problem. By using the softmax activation function on this layer, we were able to generate a distribution of probability over both classes. The model's predictions may be clearly understood in terms of probability thanks to the softmax function, which makes sure that the output values add up to one.

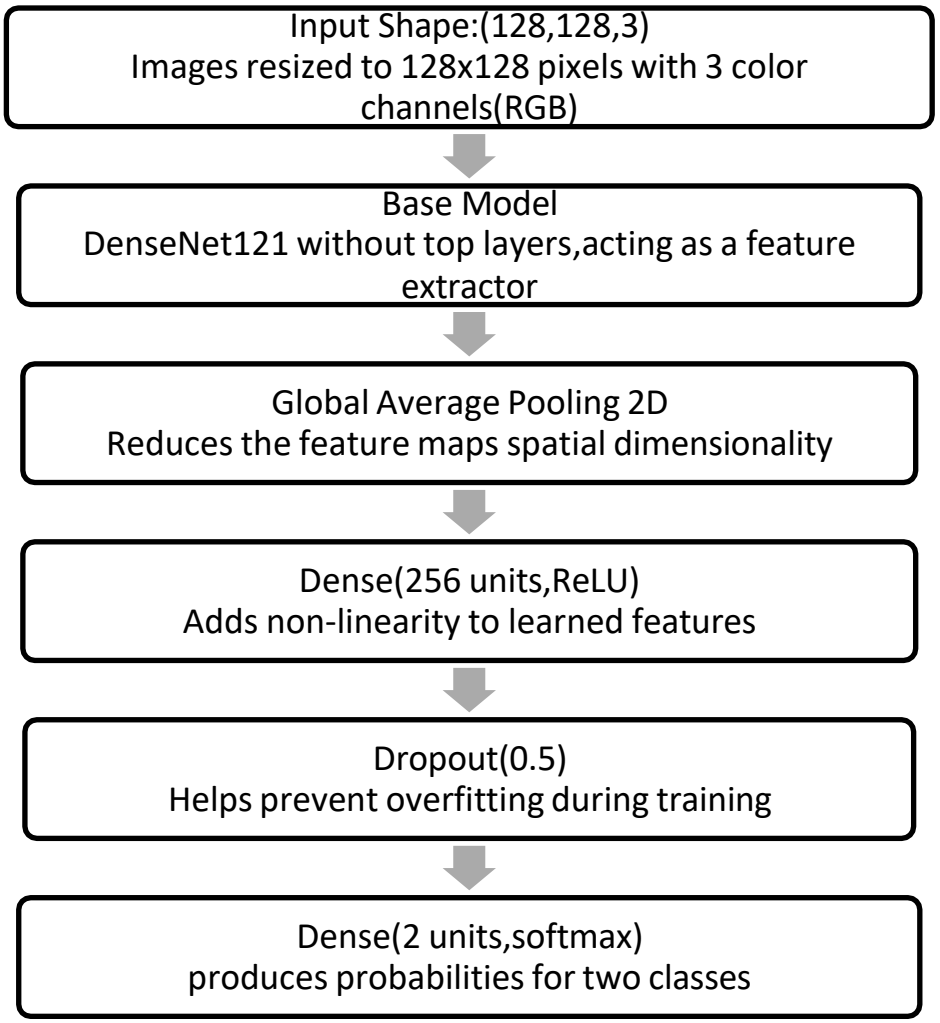



Fig-6 Model Architecture of our proposed Method

The following pseudocode describes the steps of the proposed system:

```
Define convert_to_ela_image(path, quality):
Open image and convert to 'RGB'
    Save image as 'JPEG' with specified quality
    Open the saved JPEG image
    Calculate the pixel-wise difference between original and JPEG image
    Enhance the difference image
    Return the enhanced difference image
Define prepare_image(image_path):
    Convert image to ELA format
    Resize ELA image to 128x128
    Flatten and normalize the ELA image
    Return the processed image
Initialize X_train, X_test, Y_train, Y_test as empty lists
Define paths for Au_Train, Tp_Train, Au_Test, Tp_Test
load the dataset
```

Convert X_train, X_test to numpy arrays
One-hot encode Y_train, Y_test
Reshape X_train, X_test to (-1, 128, 128, 3)
Define build_model():
Load DenseNet121 model pre-trained on ImageNet without top layers
Create a Sequential model
Add DenseNet121 as base model
Add GlobalAveragePooling2D layer
Add Dense layer with 256 units and 'relu' activation
Add Dropout layer with 0.5 dropout rate
Add Dense layer with 2 units and 'softmax' activation
Return the model
Initialize model by calling build_model()
Print model summary
Define training parameters: epochs, batch_size, learning_rate
Compile model with Adam optimizer, binary cross-entropy loss, and accuracy metric
Define ModelCheckpoint and EarlyStopping callbacks
Train model using model.fit() with X_train, Y_train, batch_size, epochs, validation_data, and callbacks
Predict Y_pred using model.predict() on X_test
Convert Y_pred to class labels using argmax
Convert Y_test to class labels using argmax
Compute confusion matrix (cm) using confusion_matrix() with Y_true_classes and Y_pred_classes
Extract TN, FP, FN, TP from confusion matrix
Calculate precision, recall (TPR), F1 score, error rate, TNR, FPR, and accuracy using appropriate formulas
Print precision, recall, F1 score, error rate, TNR, FPR, and overall accuracy
Plot confusion matrix:
Normalize confusion matrix
Set up plot with titles and axis labels
Fill plot with values from confusion matrix
Display plot

We set the following parameters for our proposed model's training procedure using this implementation:

| | |
|---|---|
| Training | For the training dataset, we took 80 percent of the data from CASIA v2.0. |
| Batch size | We utilized a 32-person batch size for the training. |
| Testing | 20% of the data are in the testing dataset. |
| Dropout | Based on our observations, we employed a dropout rate of 0.7. Thus, overfitting is avoided |
| Loss | The loss function used was binary cross-entropy. In scenarios involving binary categorization, it is useful. |
| Optimizer | Adam Optimizer was employed. |
| Learning | Our initial implementation has a 1e-5 learning rate |

| | |
|---|---|
| Epochs | For the purposes of our model implementation, we have chosen a maximum of 10 epochs. Because it provides greater accuracy even when the smallest number of epochs is run. |

TABLE II. SUMMARY OF PARAMETERS OF THE PROPOSED MODEL

| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| Densenet121(Functional) | (None, 4, 4, 10124) | 7037504 | input_1[0][0] |
| Global_average_pooling_2d | (None, 1024) | 0 | densenet121[0][0] |
| Dense (Dense) | (None, 256) | 262400 | global_average_pooling2d[0][0] |
| Dropout (Dropout) | (None,256) | 0 | dense[0][0] |
| Dense_1 (Dense) | (None,2) | 514 | dropout[0][0] |

Total parameters: 7,300,418

Trainable parameters: 7,216,770

Non-trainable parameters: 83,648

## 4 EXPERIMENT RESULTS AND DISCUSSION

This section covers the training and evaluation conditions for the proposed approach Moreover, we examine and contrast its effectiveness with alternative approaches.

### EXPERIMENTAL CONFIGURATION

We evaluated the proposed method's efficacy with the popular CASIA 2.0 image forgeries database. The database contains a total of 12,615 images in the BMP, JPG, and TIF formats; 5,123 of them are photoshopped images, while 7,492 are actual images. CASIA 2.0 contains images from many different genres, such as individuals, wildlife, buildings, articles, plant life, the environment, scenery, materials, and interior pictures. The images in the database vary in size and resolution from 800 × 600 pixels to 384 × 256 pixels. Table 1 contains information on the CASIA 2.0 database. VSCode is used to implement the suggested work in Python. All processing was done on a typical PC with a 12th Gen Intel(R) Core(TM) i5-1240P 1.70 GHz processor and 16 GB of RAM.

### EVALUATION METRICS:

To evaluate and compare the recommended strategy with other ways, we compute the accuracy, recall, precision, and F-measure. The following is how these metrics are defined and computed:

1. **Accuracy:** The percentage of authentic and manipulated images that were successfully identified out of all the images that were evaluated.

Accuracy= (TruePositive + TrueNegative /T total_Images)x100

1. **Recall:** The proportion of correctly identified truly modified photographs among all the tampered regions

Recall= TruePositive /( TruePositive +FalseNegative)

2. **Precision:** the percentage of altered images that were accurately identified among all images that were tampered with.

Precision= TruePositive / TruePositive +FalsePositive)

**4.F-measure(F1 Score):** The two metrics are provided as a ratio via the harmonic mean of precision and recall.

F – measure = ((2 x Recall x Precision)/(Recall + Precision))x100

## 4. 2 RESULTS

Instead of our proposed DenseNet-121 model, we also evaluated several other existing models. Among them, our modified DenseNet-121 model yielded the best results.

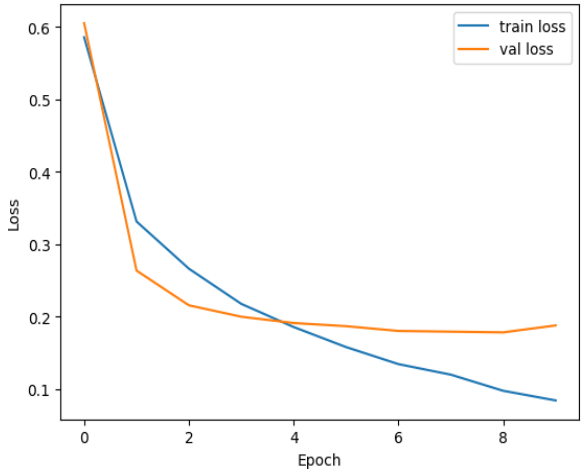
| Model | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| VGG16 | 0.8970 | 0.8468 | 0.9112 | 0.8778 |
| VGG19 | 0.8839 | 0.8667 | 0.8439 | 0.8552 |
| InceptionV3 | 0.809 | 0.7856 | 0.7688 | 0.771 |
| MobileNetV2 | 0.9191 | 0.9131 | 0.9122 | 0.9126 |
| Resneet50 | 0.9239 | 0.8874 | 0.9207 | 0.9086 |
| Proposed Model | **0.9303** | **0.9028** | **0.9239** | **0.9132** |

Using max-pooling layers and 3x3 convolution filters, VGG16 is a 16-layer deep convolutional neural network. Strong recall is accompanied by lower precision, which increases false positives. VGG19 keeps the same concept while expanding VGG16 to 19 layers. It marginally underperforms VGG16 in recall but has good precision and reasonable recall.
Using parallel convolutions of various sizes, InceptionV3 uses inception modules to collect multi-scale characteristics. It is less suited for applications requiring a high degree of accuracy and precision because it performs the worst overall out of all the models.
Depth-wise separable convolutions are used by MobileNetV2, a mobile application framework, to lower parameters and computational costs without sacrificing robust performance in any measure.ResNet50 is a skip-connected deep residual network that consistently achieves excellent recall and accuracy, giving it a dependable method for detecting true positives. When compared to the suggested model, its precision is marginally less. The proposed model, a newly constructed neural network, outperforms the most advanced models in terms of accuracy, precision, recall, and F1-Score. It achieves this higher performance while maintaining a strong balance between decreasing false positives and negatives and recognizing meaningful occurrences.

### 4.3 Results in terms of testing accuracy and testing loss:

Testing Loss and Testing Accuracy are two essential parameters that are widely used to evaluate how well machine learning models function.
Testing Accuracy: This tests the accuracy of a model by identifying what proportion of actual positives were correctly identified from amongst all positive cases in test dataset. It is calculated by the quotient of the total number of forecasts over correct predictions.
Testing Loss: This metric evaluates how well the model is performing by determining a loss using this test dataset and seeing error for prediction made relative to actual outcomes.

Model Loss (Figure-7)




Model Accuracy (Figure-8)

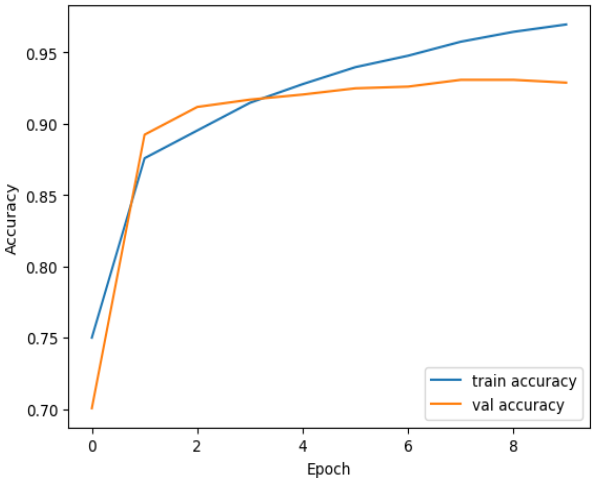



Figure 7: Training loss and validation loss graph over ten epochs. While the validation loss peaks at epoch 4, the training loss continuously declines. This implies that the training data have caused the model to overfit. The Figuer-8 displays a model's accuracy throughout ten epochs of training and validation. The training accuracy gradually rises to approximately 0.97 from a starting point of roughly 0.75. The validation accuracy gradually rises to approximately 0.93 from a starting point of roughly 0.70.

**TABLE III: ACCURACY, TEST LOSS, AND TRAINING DURATION OF PROPOSED MODEL**

| Epochs (E) | Time (s) | Training Loss | Training Accuracy | Testing Loss | Testing Accuracy |
|---|---|---|---|---|---|
| Epoch-1 | 908 | 0.75 | 0.64 | 0.60 | 0.7005 |
| Epoch-2 | 792 | 0.35 | 0.86 | 0.26 | 0.8922 |
| Epoch-3 | 743 | 0.28 | 0.88 | 0.21 | 0.9116 |
| Epoch-4 | 615 | 0.21 | 0.91 | 0.19 | 0.9168 |
| Epoch-5 | 445 | 0.18 | 0.92 | 0.19 | 0.9204 |
| Epoch-6 | 473 | 0.16 | 0.93 | 0.18 | 0.9247 |
| Epoch-7 | 460 | 0.12 | 0.93 | 0.17 | 0.9259 |
| Epoch-8 | 465 | 0.12 | 0.95 | 0.17 | 0.9387 |
| Epoch-9 | 461 | 0.10 | 0.96 | 0.17 | 0.9307 |
| Epoch-10 | 466 | 0.08 | 0.96 | 0.16 | **0.9330** |

Epoch 1: At the beginning of the model, the training accuracy is 0.64 and the training loss is 0.75. Testing accuracy is 0.7005 and testing loss is 0.60. Epoch-2: A notable improvement is observed, with the training accuracy rising to 0.86 and the training loss decreasing to 0.35. Testing accuracy increases to 0.8922 while testing loss drops to 0.26. From Epoch 3 to Epoch 10, the model shows consistent improvement, with training accuracy rising from 0.88 to 0.96 and training loss gradually falling from 0.28 to 0.08. As a result, testing accuracy increases from 0.9116 to 0.9330, while testing loss drops from 0.21 to 0.16. Epoch-10 has the highest testing accuracy, with a rating of 0.9330.

## TABLE IV: CONFUSION MATRIX FOR THE BINARY CLASSIFICATION OF THE PROPOSED MODEL

**Confusion Matrix:**

A confusion matrix is a table used to evaluate the performance of a categorization model by comparing expected and actual values. It includes.

1. True Positives (TP): Positives that were correctly predicted.
2. True Negative (TN): Negatives that were correctly predicted.
3. False Positive (FP): An incorrect positive prediction made.
4. False Negative(FN): A prediction that was made incorrectly.

To support performance evaluation, each row displays the actual classes, and the columns show the anticipated classes. The following metrics are derived: F1-score (harmonic mean of precision and recall), recall (true positives in all actual positives), specificity (true negatives in all actual negatives), accuracy (right classifications), and precision (true positives in all projected positives). **Figure 9** demonstrates the high prediction accuracy of the model for both manipulated and unaltered photos. 947 out of 1025 genuine photos and 1397 out of 1499 manipulated images were accurately categorized by the model. This suggests that the model does a decent job of differentiating between altered and original photos. Although there are some false positives and false negatives in the model, they are not particularly common, which suggests the model does its purpose quite well.

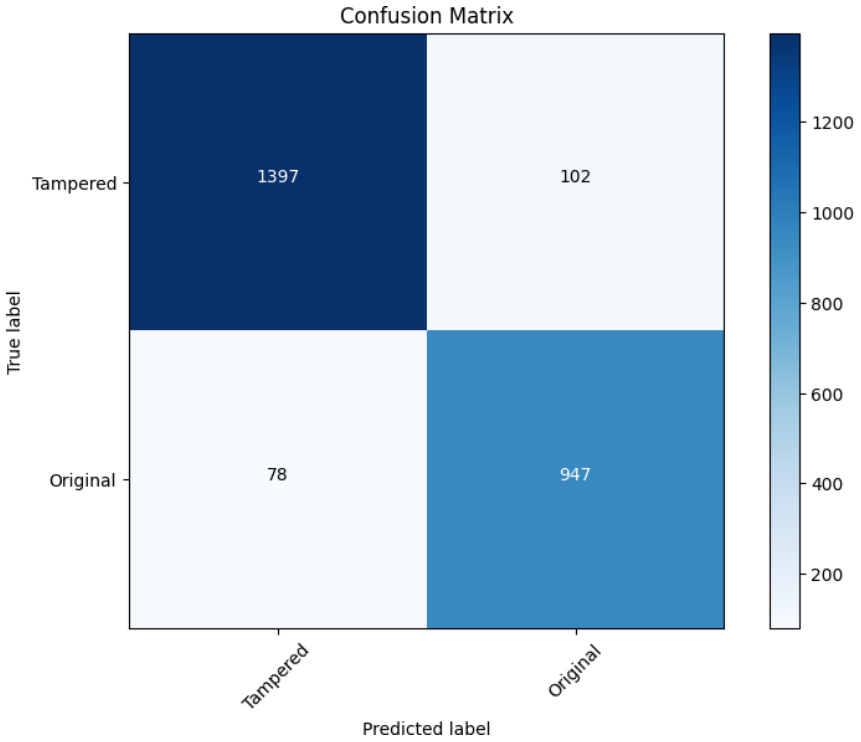



**Figure-9**

## 4.3 COMPARISION WITH OTHER TECHNIQUES

The accuracy of the suggested methodology was compared to other current methods. The authors have published all of the findings in this section in their individual research.

| Model | Accuracy | Precision | F1-measure |
|---|---|---|---|
| S. S. Ali [14] | 92.23 | 85 | 0.91 |
| Reference [15] | 92.23 | 85 | 0.91 |
| 1DDCT & DWT [16] | 90.00 | 84 | 0.87 |
| Z. He, W. Lu [17] | 89.76 | - | - |
| Kubal, P., Mane [18] | 92.10 | - | - |
| Proposed Model | **93.30** | **90** | **0.91** |

**Table-5:Accuracy, Precision, F1-Measure corresponding to architectures**

A number of studies have provided good grounds for comparing the efficiency in terms of images forged detection widely. Ali et al. built their unique deep learning approach by employing picture recompression for on Data Changeset Query The results reported in (2022) attained an accuracy rate of 92.23%, a precision of 85% and F1-score =0.91, which represent significant progress accomplished by the end-to-end baseline INSALON. Given this, Al_Azrak et al. Deep learning techniques and trigonometric transforms were also introduced by Saqlan Naqvi et al. wrote (2020) they achieved an excellent accuracy of 90% in their study. However, their previous work (Blom et al., 2012) concentrated on detection of digital picture splicing and adopted Markov features in DCT and DWT domains with an accuracy of approx. More recently, Kubal et al. Using deep learning methods combined with ELA, they managed a thorough evaluation of 92.10% accuracy in the robust examination (2023).

The proposed DenseNet121 model which is better than above approaches gives F1-measure of 0.91, Precision=90% and Accuracy=93.3%. The proposed model capable of detecting two well known types copy-move and image splicing forgeries solidify its versatility and robustness. DenseNet121 model is a cool way to detect the forgery among images, and by using transfer learning agile training (training time dropped significantly but without reducing accuracy).

**CONCLUSION AND FUTURE SCOPE**

This study is the same work but with different topologies in a dataset to recognize image fraud quickly and accurately. Based on our findings we decide which is the best model and better performance metric but only with a few epochs of training. There are many ways to detect fake photos. They used the DenseNet121 model, which is much more accurate than CNN. DenseNet121 achieves 93.3% accuracy in less than 20 epochs, where our model has to run around ~100 epoch to reach the same level ()dense)immigrate-123). Although the proposed technique is based on picture recompression for detecting properties of full-frame copy-move and image splicing frauds with different configuration parameters (different levels JPEG quality).

Although the given model seems promising, there are several domains that could be expanded. Next time research could address:

1.Analyzing diverse deep learning architectures - Whether different kinds of DLs such as ResNet, VGGNet and/or Inception Net are good in detecting forged images.

2.Advertisements for boosting robustness against adversarial attack - Ways to harden a model (strategies that use the changes in input images) so it can better defend attempts of fooling deep learning models.

3.Building a real-time forgery detection system: Lighter weight models or optimized algorithms to be employed for online social media platforms/live streaming apps so that they can detect and discard these attacks in almost realtime.
Improving generalization to a variety of picture datasets: Looking into methods to train the model on a wider range of image datasets with a variety of attributes in order to improve its generalization capacity.
Overall, this study demonstrates how well deep learning works for detecting image forgeries.

# References


1. Ansari, Mohd Dilshad, Satya Prakash Ghrera, and Vipin Tyagi. "Pixel-based image forgery detection: A review." *IETE journal of education* 55, no. 1 (2014): 40-46.
2. Abidin, A. B. Z., Majid, H. B. A., Samah, A. B. A., & Hashim, H. B. (2019, December). Copy-move image forgery detection using deep learning methods: a review. In *2019 6th international conference on research and innovation in information systems (ICRIIS)* (pp. 1-6). IEEE.
3. Fridrich, J., Soukal, D. and Lukas, J., 2003, August. Detection of copy-move forgery in digital images. In *Proceedings of digital forensic research workshop* (Vol. 3, No. 2, pp. 652-63).
4. Popescu, A. C., & Farid, H. (n.d.). Exposing Digital Forgeries by Detecting Duplicated Image Regions. (2000), 1–11.
5. Zhang, J., Feng, Z. and Su, Y., 2008, November. A new approach for detecting copy-move forgery in digital images. In *2008 11th IEEE Singapore International Conference on Communication Systems* (pp. 362-366). IEEE.
6. Kumari, Ritesh, and Hitendra Garg. "Image splicing forgery detection: A review." *Multimedia Tools and Applications* (2024): 1-39.

7. Rao, Yuan, and Jiangqun Ni. "A deep learning approach to detection of splicing and copy-move forgeries in images." In *2016 IEEE international workshop on information forensics and security (WIFS)*, pp. 1-6. IEEE, 2016.
8. Shi, Zenan, Xuanjing Shen, Hui Kang, and Yingda Lv. "Image manipulation detection and localization based on the dual-domain convolutional neural networks." *IEEE Access* 6 (2018): 76437-76453.
9. Agarwal, Saurabh, and Satish Chand. "Image forgery detection using multi scale entropy filter and local phase quantization." *International journal of image, graphics and signal processing* 7, no. 10 (2015): 78.
10. Barad, Zankhana J., and Mukesh M. Goswami. "Image forgery detection using deep learning: a survey." *2020 6th international conference on advanced computing and communication systems (ICACCS)*. IEEE, 2020.
11. Ali, Syed Sadaf, et al. "Image forgery detection using deep learning by recompressing images." *Electronics* 11.3 (2022): 403.
12. https://github.com/SunnyHaze/CASIA2.0-Corrected-Groundtruth
13. Rajkumar, Rajeev. "Deep Learning Feature Extraction Using Attention-Based DenseNet 121 for Copy Move Forgery Detection." *International Journal of Image and Graphics* 23, no. 05 (2023): 2350042.
14. S. S. Ali, I. I. Ganapathi, N.-S. Vu, S. D. Ali, N. Saxena, and N. Werghi, ‘‘Image forgery detection using deep learning by recompressing images,’’ Electronics, vol. 11, no. 3, p. 403, Jan. 2022.
15. Ali, S.S., Ganapathi, I.I., Vu, N.S., Ali, S.D., Saxena, N. and Werghi, N., 2022. Image forgery detection using deep learning by recompressing images. *Electronics*, *11*(3), p.403.
16. Al_Azrak, F.M., Sedik, A., Dessowky, M.I., El Banby, G.M., Khalaf, A.A., Elkorany, A.S. and Abd. El-Samie, F.E., 2020. An efficient method for image forgery detection based on trigonometric transforms and deep learning. *Multimedia Tools and Applications*, *79*, pp.18221-18243.
17. Z. He, W. Lu, W. Sun, and J. Huang, “Digital image splicing detection based on Markov features in DCT and DWT domain,” Pattern Recognition, vol. 45, no. 12, pp. 4292-4299, 2012.
18. Kubal, P., Mane, V., & Pulgam, N. (2023). Image Manipulation Detection Using Error Level Analysis and Deep Learning. *International Journal of Intelligent Systems and Applications in Engineering*, *11*(4), 91-99.