CHAPTER 4
SPLICED IMAGE FORGERY DETECTION USING INTRINSIC FOOTPRINTS OF AN IMAGE

Spliced image is another type of digital image forgery in which a region of an image is copied and pasted to another image. Here a set of two or more images are used for digital image forgery. State-of-the-art techniques are already reported by researchers in various literature for spliced image forgery detection in the past decade. These techniques are classified into two categories (discussed in chapter 2) in this thesis. These are data-driven approaches and statistical approaches which are associated with many limitations like the selection of suitable features for machine learning classification techniques and calculation of qualified noise patterns as an independent footprint of the image. Considering these limitations some techniques are developed for spliced image forgery detection and presented in this chapter. The chapter is divided into four sections- overview of spliced image and research findings are given in first and second sections respectively. The third section explains the proposed approaches which are divided into two subsections dedicated to the proposed data-driven approach and statistical approaches. Evaluation of the proposed approaches and their comparison with state-of-the-art are also given in this section. The fourth section summarizes the proposed approaches.

4.1 Background

Image splicing is a technique of image forgery where the content of the image is copied from one image and pasted to a different image in such a way that one cannot detect the tampered area [89], the objective may differ, sometimes it's just for fun but

sometimes it can be a political agenda. If the cropped region is resized in the spliced image, this process is called resampling. The forged region can be pasted with or without any post-processing operation. Hence, spliced image is a combination of two different images. In this case, different parts of the spliced image will have different properties i.e. texture or statistical patterns of the region. Examples of spliced image forgery can be seen in Figure 4.1. Since the image is also known as information currency, the authenticity of an image is important.

(a) (b) (c)

Figure 4.1: Examples of Spliced Image Forgery (a) First Original Image (b) Second Original Image (c) Spliced Image (Combination of both)

Filtering of the spliced image can be done in two ways, either by region localization or by image classification. Localization of morphed segment in an image ensures the exact location of altering which verifies whether the image has gone through any manipulation or not. Researchers descended various spliced image forgery detection techniques based on classification [42]–[48] using features of the image. Image forgery detection is a two-class problem- one is authentic or non-forged class and another is forged or tampered class. Hence, the image can be classified by using machine learning classification techniques. A large number of localization techniques [20], [23], [30], [90]–[93] are also derived from various research where traces from the image acquisition pipeline are extracted and using these traces tampering locations can be identified. These traces can be defined as physical (lighting), lens and shutter, noise and CFA traces etc.

Among these techniques, the commonly used fingerprint is noise estimation of the image which is based on sensor traces. These techniques of forgery detection assume that the pattern of noise in an authentic image is homogeneous (uniform and white Gaussian noise) across the entire image. Editing of an image may add some different noise patterns which may cause inconsistent noise locally. Literature research has been reported in recent years for image forgery detection with this concept. These techniques have some limitations reported in section 4.2. These limitations are identified and tried to overcome in the proposed approaches.

4.2 Research Gaps

From the literature survey of spliced image forgery detection techniques (given in Chapter 2), some research issues are identified. These issues are summarized as:

- Orientation features are not used for the classification of forged and non-forged images. Information regarding translation and rotation is not extracted in existing methods. Features for smooth edges from images are not identified. Hence, leading to loss of information.
- To classify the image into forged or non-forged classes these approaches use SVM or ANN-based machine learning techniques. Though SVM can handle large feature space, it is not efficient with a large number of observations.
- Blind noise estimation based on spliced image detection techniques suffers from limitations of post-processing technique, confusion between edges and noise and need for prior knowledge about the image.

Considering the above-mentioned limitations of state-of-the-art techniques, some spliced image detection and localization techniques are developed. One method is for the detection of the image whether the image belongs to a forged class or authentic class. Another method is for localization of the forged region. These techniques are given in the next section.

4.3 Proposed Method

The identified research issues motivated for the development of spliced image forgery detection and localization techniques. In these methods, limitations of state-of-the-art techniques are tried to overcome. This section focuses on these two proposed methods- one is the detection of spliced image and another is the localization of spliced region. These techniques are discussed in the given subsections-

4.3.1 A Technique for Image Splicing Detection using Hybrid Feature Seet

The proposed method works on combined texture and shape features. In this method, the image is first converted into the grayscale image from which 59 features of LBP, 36 features of DWT, 15 LTE features and 32 HoG features are extracted and combined to make a feature vector. A total 142-dimension feature vector is given to a logistic regression classifier to classify the image into a forged or non-forged image. To avoid overfitting, a 10-fold cross-validation test is used to examine the proposed method. The following points are of major concern in this work:

- Proposed a relevant feature set for image splicing detection based on the theoretical and experimental analysis.
- Rigorous comparison of the projected system with state-of-the-art techniques for image splicing detection with performance analysis of the anticipated work.

4.3.1.1 Method and Model

Image forgery detection is a binary class classification problem based on the decision of whether the image is forged or not. This work proposes a machine learning-based automatic decision tool to identify the forged image. In this method, the first input image is preprocessed by converting the color space of the image from RGB to Gray level, and then from this grayscale image, image features (HoG, LTE and DWT and LBP) are extracted. These relevant sets of features are combined to make a feature vector. A machine learning classifier logistic regression is used to train and classify forged and

authentic images. The proposed method improves the correctness of detection using a combination of texture features i.e. spatial and frequency-based. Also, the method reduces the overall cost of image splicing type forgery detection. Here, Figure 4.2 shows the flow diagram of the proposed method. The method has been divided into three sections, the first is image preprocessing, the second is feature extraction and the third is classification.

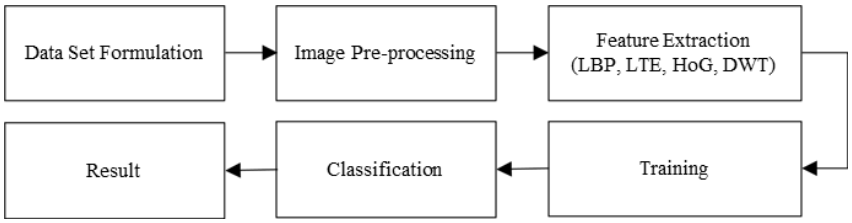


Figure 4.2: Flow Diagram of the Proposed method

4.3.1.1.1 Pre-processing

Initially, the input image is pre-processed, as to capture color features there will be a need for the original true-color image but for the LBP texture features a grayscale image of the original image will be required. In this situation, the original RGB color image will be converted into a grayscale image. Figure 4.3 shows a sample output of the same. Conversion formula according to [94] is:

$$GrayScale = 0.289 \times R + 0.587 \times G + 0.114 \times B \quad (4.1)$$

Color Image Grayscale Image

Figure 4.3: Color Conversion of the input image

4.3.1.1.2 Feature Extraction

According to the application and type of data, features can be extracted. To determine the features for image forgery detection in this method, various problems faced by researchers were analyzed. Operations used in image splicing are being found, which can help in determining the features. These operations are- translation, rotation and

scaling. Some-time smoothness of edges also creates a problem in identifying spliced objects edge detection. In the proposed approach these problems are identified and based on these, features are extracted which may be used to counter these problems. The following Figure 4.4 shows four types of features are extracted in this method which helps in identifying spliced objects in an image. Their description of why these features are used for splicing detection are given below-

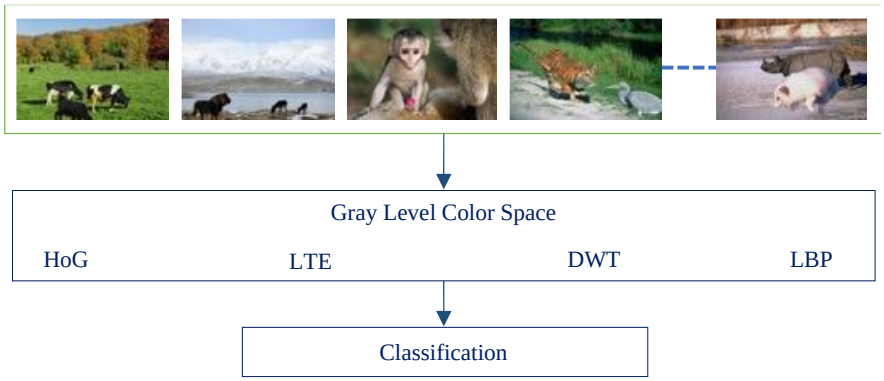


Figure 4.4: Multiple features from the input image Gray-level color space

A) HoG Features(Histogram of Oriented Gradients): The HoG features descriptor [95] is used for perceiving and extracting the orientation of the image. The descriptor has the advantage that it operates on local cells and is invariant to photometric and geometric transformation, thus it is used to describe the appearance and shape of the local object in some manner by describing the distribution of intensity gradient or edge direction. First, an input picture is divided into a small connected component known as cells and then for every pixel in these cells, a histogram of gradient direction is computed. In this method, HoG features around image corner points are extracted. In the proposed technique 36 features (F1-F36) of HoG have been taken. The method is also explained in Figure 4.5.

First, Corner points from the pre-processed image (*I*) are detected using Feature from Accelerated Segment Test Algorithm [96].

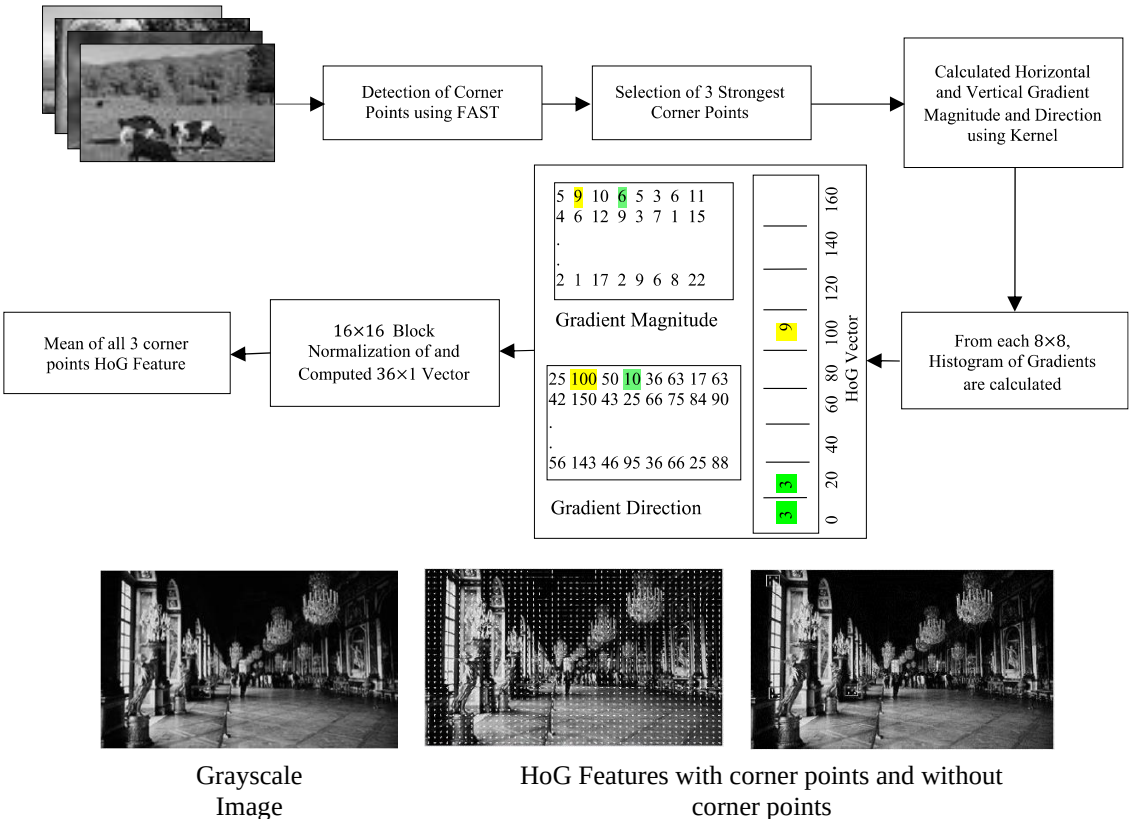


Figure 4.5: Extraction of HoG Based Features from Pre-processed Image

$$CornerPoints = DetectFAST\ [I(i,j)] \tag{4.2}$$

Among these detected corner points three strongest corner points are selected-

$$StrongestCorners = selectStrongest\ [CornerPoints, 3] \tag{4.3}$$

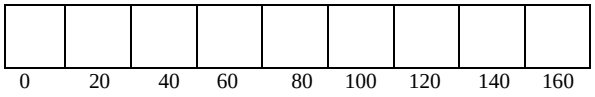
HoG features around these strongest corners are calculated as in [95]. To calculate the HoG feature first, vertical and horizontal gradients are calculated by filtering the image using $[-1\ 0\ 1]^T$ and $[-1\ 0\ 1]$ kernels. Then the gradient direction and magnitude are found using:

$$Magnitude = \sqrt{{G_x}^2 + {G_y}^2}, \qquad (Direction)\theta = \tan^{-1}\frac{G_y}{G_x} \tag{4.4}$$

Histogram of gradients are computed using 8×8 cells, from each cell 9 bins are computed using both magnitude and direction matrix. Where bins represent the direction

angle from 0 to 160 (0,20,40 … 160) and magnitude of gradients are inside these bins respected to its direction.

HoG Vector =



Then block normalization using L2 norm is done using 4 cells and this process is repeated. The mean of these HoG Features for all three corners is calculated for the feature vector.

$$Mean = \frac{1}{3}\sum_{i=1}^{3} HoG_i \tag{4.5}$$

B) LTE Feature (Laws Texture Energy): Texture features are very important and used to detect the intrinsic anatomical and textural changes in the image. Thus, the texture features are key features to distinguish images into two classes. LTE measures the volume of fluctuation in energy inside the window with a fixed size for an image [97]. This technique uses the mask for discriminating the energy of different textures. In this approach, the energy of the texture transformation is used to evaluate the energy presented in the pass filter region. Using four 1×5 masks, a combination of fifteen 5×5 masks is generated to calculate texture energy. Figure 4.6 explains the extraction of LTE features. The four masks are defined below where the name suggests meaning itself. Here 15 mean features (F37-F51) are used using the below masks-

L (Level)	=	[1, 4, 6, 4, 1]
E (Edge)	=	[-1, -2, 0, 2, 1]
S (Spot)	=	[-1, 0, 2, 0, -1]
R (Ripple)	=	[1, -4, 6, -4, 1]

Combinations of the above masks were taken (L'E, E'L, L'R, R'L, E'S, S'E, S'S, R'R, L'S, S'L, E'E, E'R, R'E, S'R and R'S) to generate fifteen 5×5 Mask.

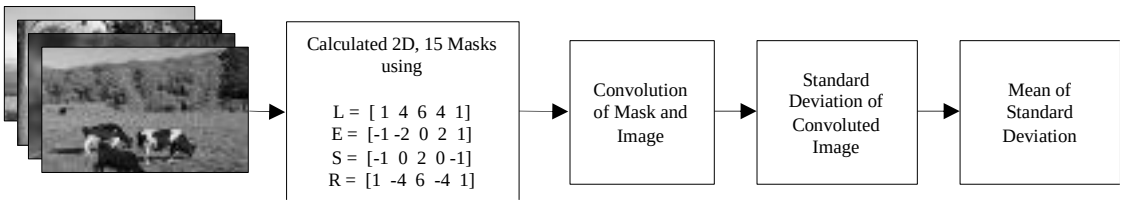


Figure 4.6: Extraction of LTE Based Features from Pre-processed Image

$$Mask = Transpose[L|E|S|R] \times [L|E|S|R] \quad (4.6)$$

Then these masks are applied to the pre-processed image (Convolution).

$$F = I * Mask = \sum_{i=1}^{n}\sum_{j=1}^{m} Mask(i,j) \times I(x-i, y-j) \quad (4.7)$$

From these filtered images mean from the standard deviation matrix row is calculated for the feature vector.

$$Mean = mean[stdDev(F)], \qquad stdDev: \text{Standard Deviation} \quad (4.8)$$

C) Wavelet Features (DWT): It is a frequency-based model for texture feature classification. In this, specific wavelet coefficients hold the energy of a signal. In DWT, a digital filtering technique is used for the timescale depiction of the digital signal. The images are decomposed into the small wavelets into their sub-frequency bands that are Low-Low (Approximation coefficients 'LL') and Low-High (LH), High-High (HH), High-Low (Detailed Coefficients 'HL') [98] as shown in Figure 4.7. In this work, 32 features (F52-F83) of the DWT are taken in the following manner.

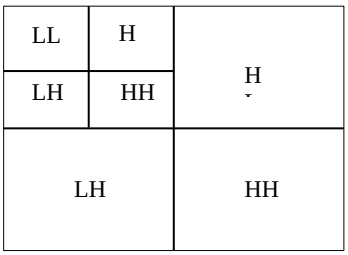


Figure 4.7: Frequency Representation of DWT

First, the pre-processed image was resized into 256×256 size, then applied 2D wavelet transform on this image and its approximation coefficient was further divided

into four sub-bands till its fourth level approximation coefficient. From this 16×16 size, coefficient means, and standard deviations of each row are calculated for 32 features. Here, Figure 4.8 explains its flow diagram.

$$Mean = \frac{1}{n}\sum_{j=1}^{n}|LL^4(i,j)| \tag{4.9}$$

$$SD = \sqrt{\frac{1}{(n-1)}\sum_{j=1}^{n}|LL^4(i,j) - Mean(LL^4)|^2} \tag{4.10}$$

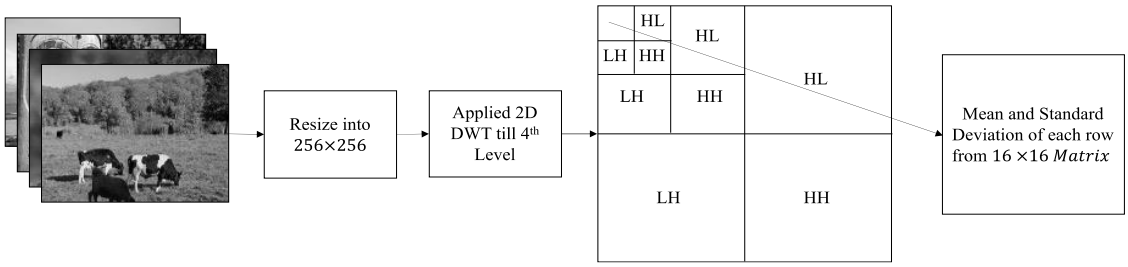


Figure 4.8: Extraction of DWT Based Features from Pre-processed Image

D) LBP Features: Local Binary Pattern is an efficient local descriptor and one of the unvarying features for rotation and grayscale conversion. LBP descriptor is used to achieve a feature vector that may be partially invariant to rotation, scaling and translation. Ojala et.al [99] have introduced the local pattern for the characterization of the spatial structure in local image texture. The LBP code is computed by comparing the pixel to its neighboring pixel cell. In the proposed approach, a uniform local binary pattern family is used where fifty-nine LBP features (F84-F142) are extracted. Figure 4.9 represents the uniform LBP extraction method. In this method, the preprocessed image is divided into a cell size of 3×3 with 8 neighbors and a radius of one. Then for each pixel in a cell LBP is computed using -

$$LBP^{u}_{(p,r)} = \sum_{p=0}^{p-1} s(g_p - g_c)2^p, \qquad s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{4.11}$$

g_c: center pixel

g_p: neighbors on a circle with radius r

(p, r): the subscript represents p-neighbor with r-radius

u: uniform local pattern

Histogram, over each cell, is then computed.

$$H_k = \sum_{i,j} I[f(i,j) = k], \qquad i = 0, 1, 2 \ldots n-1 \tag{4.12}$$

Now histogram is normalized to get a coherent description:

$$N_k = \frac{H_k}{\sum_{i=0}^{n-1} H_i} \tag{4.13}$$

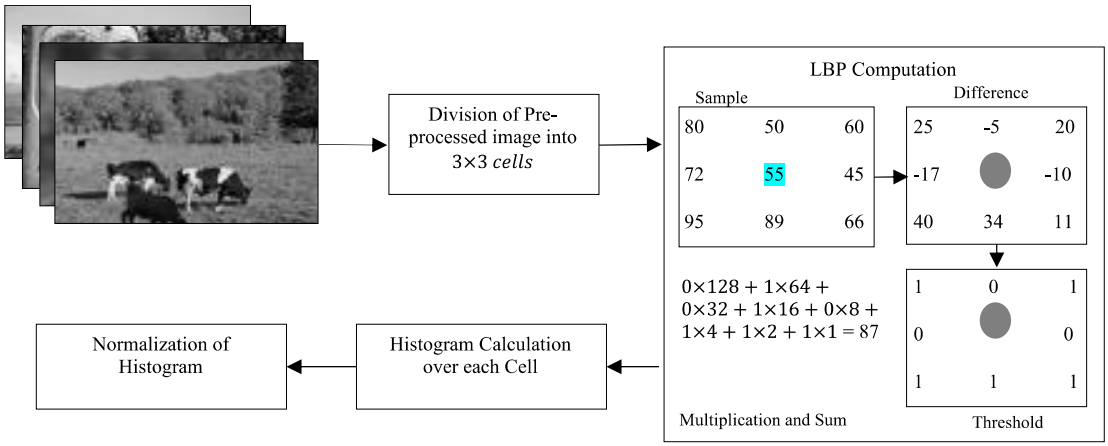


Figure 4.9: Extraction of LBP Features from Pre-processed Image

Here, Table 4.1 defines feature numbers with their name and feature group used in the proposed approach.

Table 4.1: List of features used in the Proposed Approach

Feature Group	Feature Number	Feature Description
HoG Features	F1-F36	Mean of HoG Features around strongest corner points
LTE Features	F37-F51	Mean of the standard deviation of a convoluted matrix of Image and above-mentioned masks
Wavelet Features	F52-F67	Row-wise Mean of Fourth Level Approximation Coefficient
	F68-F83	Row-wise Standard Deviation Fourth Level Approximation Coefficient
LBP Features	F83-F142	Extracted Uniform Local Binary Patterns from Pre-processed Image

4.3.1.1.3 Classification

Detection of the forged image is a two-class classification problem where predicted output y has two classes, one is a positive class (authentic image) and another is a negative class (forged image) -

$$y \in \{0,1\}, \quad \begin{matrix} 0: Forged\ Image \\ 1: Authenticated\ Image \end{matrix} \tag{4.14}$$

To classify training image dataset into two classes forged and authentic, one thing that can be done is to apply linear regression algorithm to the dataset and just try to fit the straight line to the dataset and hypothesis may get as-

$$h_{\emptyset}(x) = \emptyset^{T}(x), \tag{4.15}$$

where $'x'$ is the number of features and $\emptyset$ is the cost function.

In this case, the prediction may be done using a threshold at 0.5 as:

$$if\ h_{\emptyset}(x) \geq 0.5,\ predict\ y = 1$$
$$if\ h_{\emptyset}(x) < 0.5,\ predict\ y = 0$$

For a two-class classification problem, it is known $y = 0\ or\ y = 1$, but in the case of linear regression, $h_{\emptyset}(x)$ can be $> 1\ or < 0$, which is irrespective of the given problem. Thus, there will be a need for a classification algorithm whose hypothesis must between 0 and 1 ($0 \leq h_{\emptyset}(x) \leq 1$). For this, logistic regression classification technique has been used in the proposed method, which hypothesis function derived as:

$$h_{\emptyset}(x) = \psi(\emptyset^{T}x), \qquad where\ \psi(i) = \frac{1}{1+e^{-i}} \tag{4.16}$$

Here, $\psi(i)$ is called sigmoid or logistic function, so the hypothesis of logistic regression can be formulated as:

$$h_{\emptyset}(x) = \frac{1}{1+e^{-\phi^{T}x}} \tag{4.17}$$

Now, for fitting parameters there is an interpretation of hypothesis output:

$$h_{\emptyset}(x) = estimated\ probability\ that\ y = 1\ on\ input\ vector\ x$$

For example $if\ x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ LBP \\ HoG \end{bmatrix}, and\ h_{\emptyset}(x) = 0.7$ then the chance of an authentic image is 70%.

$$h_{\emptyset}(x) = P(y = 1|x; \phi) : probabilty\ that\ y = 1\ given\ x, parameterized\ by\ \phi$$

hence, $P(y = 0|x; \phi) = 1 - P(y = 1|x; \phi)$.

Pseudo Code: Image Splicing Detection

```
        Input:  Collected Dataset
        Output: Result(Confusion Matrix)
        Label: Authentic/Forged
1:   FOR each image in dataset
2:     Read image
3:     IF size(image)>3   //color image
4:        Conversion of color image into grayscale
5:     END IF
6:     F1_F36   = Calculate_HoG_Feature(grayscale_image);
7:     F37_F37  = Calculate_LTE_Feature(grayscale_image);
8:     F52_F83  = Calculate_Wavelet_Feature(grayscale_image);
9:     F84_F142 = Calculate_LBP_Feature(grayscale_image);
10:    Feature_vector = [image,label,F1-F142];
11:   END FOR
12:  Result = classification(Feature_vector);
```

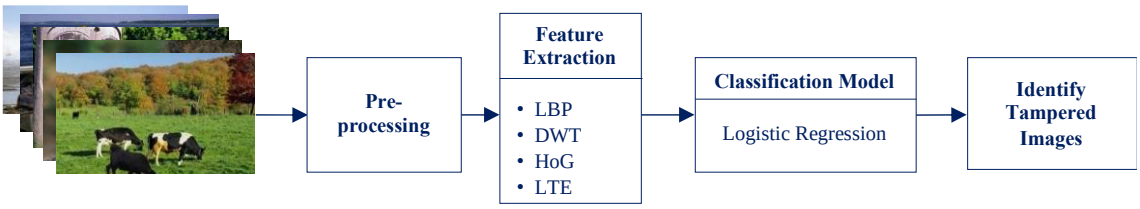


Figure 4.10: Overall Framework for Image Forgery Detection

4.3.1.2 Result Analysis and Discussion

The proposed method is evaluated with several experiments for the detection of the tampered image. The method is evaluated over different types of image datasets (CASIA v1.0 and CASIA v2.0) [65], [100] having several categories like the scene, natural, texture, animal and a COLUMBIA dataset contains different types of texture

image some of them are without corners and edges (discussed in Chapter 2). Images of these datasets are used for the training and validation of the model. The performance of the proposed method is also compared with state-of-the-art approaches for image splicing detection on these datasets presented in the following sections. A 10-fold cross-validation test is done on the given datasets to train and test the proposed model.

4.3.1.2.1 Case Study 1: Results and Analysis for CASIA v1.0 dataset

In the first case, the experiment is performed on the first dataset which is CASIA v1.0. Table 4.2 explains the result of an experiment on the CASIA v1.0 dataset with the dimension of the feature vector. This table also compares the result of the proposed method in [46], [47] and [48] on the same dataset. The result of accuracy with specificity and sensitivity rate of the proposed method given in table shows the proposed method is better comparatively where proposed method having 142-dimension vector with accuracy 98.3% and specificity and sensitivity rates are 98.25% and 97.60% respectively.

Table 4.2: Experimental Result of the Proposed method on CASIA v1.0 dataset

Method	Dimension	Accuracy (%)	Sensitivity (%)	Specificity (%)
Muhammad *et. al* [46]	360-490	94.89	96.38	94.94
Agarwal *et. al.* [47]	1024-2048	95.41	97.65	93.16
Abrahim *et. al.* [48]	-	97.4	99.03	96.07
Proposed	142	98.3	97.6	98.25

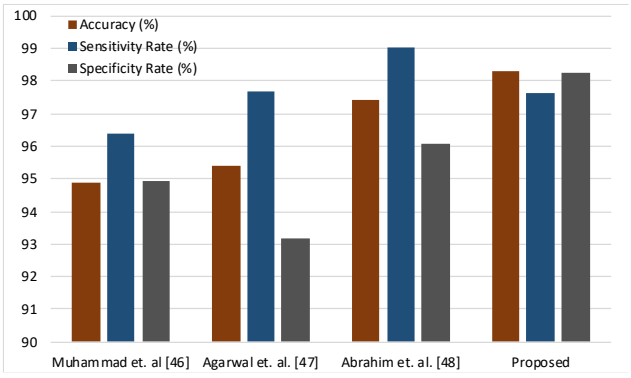


Figure 4.11: Result Analysis of the Proposed method on CASIA v1.0 dataset

The result can be seen in the bar graph which also shows the comparative analysis of other state-of-the-art methods. The method proposed by Muhammad *et. al.* [46] gives

an accuracy rate of less than 95% on the CASIA v1.0 dataset, where accuracy is increased to more than 98% in the proposed approach. In the same way specificity and sensitivity rates also increased from 94.9% to 98.2% and from 96.3% to 96.6%. Similarly, methods proposed by Abrahim *et. al.* [48] and Agarwal *et. al.* [47] have lesser accuracy and specificity rate on the CASIA v1.0 dataset. Though these methods have a higher sensitivity rate comparative to the proposed method, the overall accuracy rate is better in the proposed approach. In the proposed method extracted features are based on HoG, LBP, LTE and DWT which are used for texture and gradient analysis. The dataset is a collection of authentic images which are combinations of different types of textures like the sky, wood, water etc and forged images spliced from these authentic images. Experiments are performed on such massive and balanced images. The same dataset is also used in other state-of-the-art methods.

4.3.1.2.2 Case Study 2: Results and Analysis for CASIA v2.0 dataset

Table 4.3 shows the experimental result and analysis of the second case experiment on the CASIA v2.0 dataset. The table is having a feature vector dimension with accuracy, specificity rate and sensitivity rate. This table also compares the experimental result on the CASIA v2.0 dataset with another above-mentioned method on the same dataset.

Table 4.3: Experimental Result of the Proposed method on CASIA v2.0 dataset

Method	Dimension	Accuracy (%)	Sensitivity (%)	Specificity (%)
ResNet [101]	-	70.26	63.39	74.97
Zhongwei *et. al.* [45]	100	89.76	-	-
Muhammad *et. al* [46]	360-490	97.33	98.47	97.3
Agarwal *et. al.* [47]	1024-2048	98.33	99.22	97.73
Abrahim *et. al.* [48]	-	98.6	99.03	96.7
Proposed	142	99.5	99.63	99.51

Figure 4.12 explains the comparative analysis of the proposed method with other state-of-art methods on the CASIA v2.0 dataset. The overall performance of the proposed method is better than other methods. CASIA v2.0 is a balanced dataset of authentic and forged images. The proposed method is associated with more than 99% of each of the

performance metrics namely accuracy, sensitivity and specificity for the CASIA v2.0 dataset. This dataset has a large amount of authentic and forged images, the dataset is designed with different types of texture as explained in the above section. The combination of extracted feature (LBP, HoG, DWT and LTE) and logistic regression classifier gives higher accuracy, sensitivity and specificity rate comparatively.

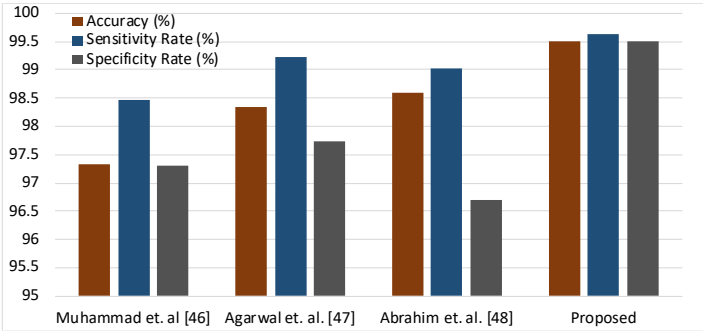


Figure 4.12: Result Analysis of Proposed method on CASIA v2.0 dataset

4.3.1.2.3 Case Study 3: Results and Analysis for COLUMBIA dataset

The third dataset COLUMBIA is also tested on the proposed classifier model. Its accuracy result with specificity and sensitivity rate is compared with the previously presented method. Table 4.4 shows the experimental result and its analysis with others result where the proposed method gives an accuracy of 98.8%, specificity and sensitivity rate of 99.53% and 99.63% respectively on the COLUMBIA dataset.

Table 4.4: Experimental Result of the Proposed method on COLUMBIA dataset

Method	Dimension	Accuracy	Sensitivity	Specificity
Zhongwei *et. al.* [45]	100	93.55	93.38	93.55
Muhammad *et. al.* [46]	360-490	96.39	96.19	94.58
Agarwal *et. al.* [47]	1024-2048	91.14	93.51	88.63
Abrahim *et. al.* [48]	-	99.43	99.03	96.07
Proposed	142	98.8	99.03	98.58

The above-mentioned bar graph in Figure 4.13 represents the comparative result of the proposed method with other state-of-the-art methods on the COLUMBIA dataset. In this case, the overall result is better than the others. The proposed method has a better

accuracy rate compared to other methods except the method proposed by Abrahim *et.al.* [48] on this dataset, but the proposed method gives better specificity and sensitivity rate.

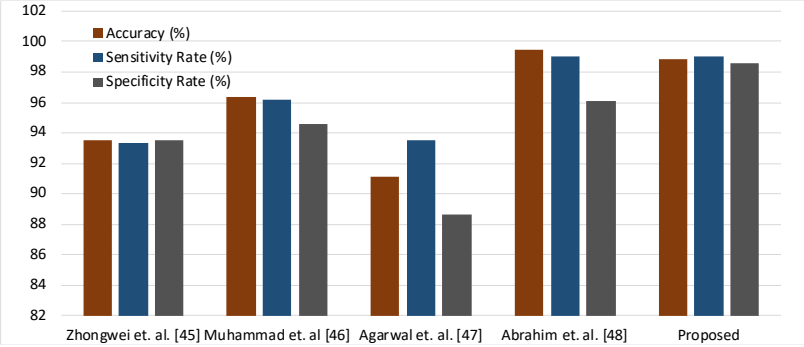


Figure 4.13: Result Analysis of Proposed method on COLUMBIA dataset

One can observe from above Figures 4.11, 4.12 and 4.13 when the experiment is performed on CASIA v1.0 and CASIA v2.0 dataset, it gives a better result compared to other state-of-the-art approaches, but in the case of COLUMBIA dataset where the dataset has computer-generated images, the proposed approach gives a slightly lesser accuracy rate compared to Abrahim *et. al.* [48]. Its explanation is as follows. A natural image consists of texture and sharp edges, smooth edges or texture can be generated with the help of advanced image editing tools in an image. In this proposed method, features from the pre-processed image are extracted and then these are combined to generate the feature vector. These features are based on LBP, LTE, DWT and HoG. The HoG is based on gradients of pixels in an image and LBP is based on the intensity of center and neighbor pixels. In the case of smooth edges, gradients and LBP may not be as good as in natural images. Though images having COLUMBIA dataset are balanced with authentic and spliced images but some of them (computer-generated images) are having smooth edges with arbitrary object boundary which creates false information after extraction of HoG and LBP-based features. Dataset having natural images (CASIA v1.0 and CASIA v2.0) have better information with the comparison to COLUMBIA. Also, the CASIA dataset is having large enough images for the training of the machine learning classifier. In this

proposed method logistic regression classifier is used to classify the images using the above features. The feature vector is also given to other classifiers where the result was poor than logistic regression and either time is taken by classifiers was greater or accuracy was lesser. In the case of forgery detection, detection is important than the time taken by the method. Although the time taken by the proposed technique is not much greater and the time taken by logistic regression was 21.87 seconds for CASIA1 while by ensemble it was 25.87 seconds on the same dataset. The accuracy, running time and prediction speed (obs/sec) are also compared with other classifiers of the proposed system on all three datasets in Table 4.5. Except for accuracy, sensitivity and specificity, other metrics are also evaluated on the same datasets. Table 4.6 is for other evaluation metrics precision and F1-Score:

Table 4.5: Running Time, Prediction Speed of different Classifiers on chosen Datasets

Classifier	CASIA1 (dataset = 1721)			CASIA2 (dataset = 12614)			COLUMBIA (dataset = 1845)		
	Time (sec)	Prediction Speed (obs/sec)	Accuracy (%)	Time (sec)	Prediction Speed (obs/sec)	Accuracy (%)	Time (sec)	Prediction Speed (obs/sec)	Accuracy (%)
Logistic Regression	21.87	8500	98.3	384.81	30000	99.5	18.763	10000	98.6
SVM	6.914	8400	97.4	78.078	10000	98.7	9.47	96000	96.5
Ensemble	38.609	310	92.3	520.32	7000	97.1	98.266	72	79.1
KNN	13.058	2900	68	454.11	470	76.2	49.38	290	96.2

Table 4.6: Experimental Result of the Proposed method on datasets

	Precision (%)	F1-Score (%)
CASIA v1.0	97.97	97.78
CASIA v2.0	99.66	99.64
COLUMBIA	98.60	98.81

From the obtained results and analysis for the chosen datasets, it has been observed that the proposed method works better than the state-of-the-art techniques, but the system gets confused in case of frequent changes of texture/sharpness in the down-sampled image. The below-mentioned Figure 4.14 shows a failure case of the proposed framework where the designed system classifies an image as authentic but originally it is spliced

image (forged). This is the case where an image may be compressed multiple times which results in the size of an image being downsampled. Also, the sharpness and texture of the image are being changed frequently in the down-sampled image can't be detected. This is the case where the classifier fails to detect the forged image and misclassifies it as an authentic image. Though the model misclassifies the forged image as an authentic image, and it is justified that the security concern is the model should not classify a forged image as authentic.

Classifier's Result	Authentic
Dataset Label	Forged

Figure 4.14: A Failure Case of the Proposed System

4.3.2 Spliced image forgery detection and localization using inconsistent noise pattern

In this work, the fourth-order central moment is used for noise estimation in the wavelet transformed image and thus for the localization of the forged region. The image is transformed into the wavelet domain first, then the block-wise central moment is calculated of 2x2 block size. A threshold is defined using noise distributions of blocks. Based on this threshold value, the pixel of the spliced region is detected. These pixels are grouped using a sequence of post-processing morphological operations. Considering the challenges and gaps mentioned in the above section 4.2 (research gaps) which are still in state-of-the-art techniques, the proposed work tries to overcome those challenges. The following points are the major contributions to this work:

- This work introduces a blind noise estimation technique locally, which aims at the detection of spliced regions in forged images efficiently.
- A sequence of suitable morphological operations for post-processing is important for the localization of spliced regions.
- Performance analysis and comparison of the projected system with state-of-the-art techniques for image splicing detection.

4.3.2.1 Proposed Inconsistent Noise Pattern Estimation Technique

This subsection presents the proposed noise estimation technique which is to be utilized in the image splicing detection scheme given in the next sub-section (Method and Model). First, how the method is different from other noise statistics estimation techniques and then how the proposed estimation technique is better than other techniques in the case of spliced forgery detection. Original Images are captured by portable cameras (imaging devices such as mobile phones, digital cameras etc.) containing random noises in them. These random noises are generated due to quantum effects, thermal fluctuations and dark current leakage in the image acquisition process. Though manufacturers are trying hard to reduce such hardware issues, noises are inevitable. Estimation of noise in a single image is very useful in the case of the application of image splicing detection. If a captured image from the imaging device is $I_C \in \mathcal{R}^2$ and $I_I \in \mathcal{R}^2$ is the ideal image (i.e. image without noise), a model of the noisy image can be represented as:

$$I_c(i,j) = I_I(i,j) + \eta(i,j) \tag{4.18}$$

where, (i,j) is the pixel location in the image and $\eta \in \mathcal{R}^2$ is noise present in the captured image. The noise model of an image can be derived as –

$$I_c = I_I + \eta \tag{4.19}$$

Generally, it is assumed that η in the image is AWGN with zero mean and σ_η^2 variance. While real-life images contain both Gaussian and non-gaussian noise in them.

In this way, to estimate noise- the goal should be the calculation of η in the image. There are multiple papers on noise estimation using variance where σ_η^2 is calculated. The problem with variance (second-order measure) is that lower order measures are better only if Gaussian noise is present in an image, while real-life images contain non-gaussian noise [102]. This is the case where higher-order measures work better. Siwei Lyu [14] utilized the statistical property of natural image in the bandpass domain where kurtosis value has been used for the blind noise estimation. The relationship between the fourth and second-order central moment can be seen as below:

Consider for 1D random variable I and $f(I)$ is the probability density function of that variable, the expected $'\mathcal{E}'$ or mean value of I to be:

$$\mathcal{E}(I) = \int_{-\infty}^{+\infty} I\, f(I) dI \tag{4.20}$$

This is also denoted by μ_1 i.e. first-order moment. For discrete random variables, the expected value is being replaced by:

$$\mathcal{E}(I) = \mu_1 = \sum_i p_i I_i \ , \quad p_i = P\{I = I_i\} \tag{4.21}$$

The variance (var) value is defined as μ_2 i.e. second-order central moment:

$$var(I) = \sigma^2 = \mathcal{E}[\{I - \mathcal{E}(I)\}]^2$$
$$\sigma^2 = \mu_2 = \mathcal{E}\{I^2\} - [\mathcal{E}\{I\}]^2 \tag{4.22}$$

With regards to this, the kurtosis (κ) value [103] is defined as:

$$\kappa(I) = \frac{\mu_4}{{\mu_2}^2} - 3 \tag{4.23}$$

Where μ_4 is the fourth-order central moment and μ_2 is the second-order central moment (variance) of I as defined in Eq. (4.22).

Now for 2D random variables, I_c (captured image) is the sum of random variables I_I(ideal image) and η (noise); see equation 4.19. So, the additivity of moments of independent variables implies: $\mu_4(I_c) = \mu_4(I_I)$ or by replacing equation (4.23):

$$(\mu_2(I_c))^2[\kappa(I_c) + 3] = (\mu_2(I_I))^2[\kappa(I_I) + 3] \quad (4.24)$$

Because μ_2 is the second-order central moment i.e. variance $\mu_2(I_c) = \sigma^2(I_c)$, hence-

$$\sigma^2(I_c)[\kappa(I_c) + 3] = \sigma^2(I_I))^2[\kappa(I_I) + 3]$$

By replacing $\sigma^2(I_c) = \sigma^2(I_I) + \sigma^2(\eta)$ and rearranging equation:

$$\kappa(I_c) = \kappa(I_I)\left[\frac{\sigma^2(I_I)}{\sigma^2(I_c)}\right]^2 - cons$$

where $cons = 3\frac{\sigma^2(\eta)}{\sigma^2(I_c)}$

$$\kappa(I_c) = \kappa(I_I)\left[\frac{\sigma^2(I_c) - \sigma^2(\eta)}{\sigma^2(I_c)}\right]^2 - cons$$

This is the relationship between the variance of random variable sample $\sigma^2(I_c)$ and kurtosis value $\kappa(I_c)$. To calculate the variance of the noise $\sigma^2(\eta)$ in an image, according to the above equation, one should know the kurtosis value of the ideal image $\kappa(I_I)$ and variance of a captured image $\sigma^2(I_c)$. But here the only variance of a captured image is known while both variance and kurtosis of an ideal image are unknown. In such a case, there will be a need for a technique for blind noise estimation using only the captured image sample. Since higher-order statistics have an advantage over low-order statistics values (i.e. higher-order statistics are needed when signals are non-gaussian [102]), the proposed approach is based on the fourth-order central moment value of the input image sample. In this approach first wavelet transformation of the image is done as

wavelet coefficients have finer details about noise than in the spatial domain. Then from this transformed image diagonal component is divided into distinct blocks as Meer et. al [104] observed that diagonal orientation has fewer edges and contours. Then the central moment value of each block is calculated and compared with its neighborhood blocks. As abrupt changes in the image are visible in wavelet components, the proposed approach took advantage of wavelet components. An image $I(i,j)$ can be defined as following in wavelet domain [105]:

$$I(i,j) = \sum_{k} W_{\varphi}(j_0,k)\varphi_{j_0,k}(n) + \sum_{j=j_0}^{\infty}\sum_{k} W_{\psi}(j,k)\psi_{j,k}(n) \tag{4.25}$$

Here W_{φ} is approximation coefficients. W_{ψ} is details coefficients. $\psi_n(i,j)$ is scaling function. j_0 is an arbitrary starting scale and n=0,1,2 .. m. Since abrupt changes are visible in the detail coefficient, only detail components are taken to compute the central moment value. Central moment values over distinct blocks are estimated as:

$$\begin{aligned} \mu_4 &= \mathcal{E}[(I-\mathcal{E}(I))^4] \\ \mu_4 &= \mathcal{E}(I^4) - 4\mathcal{E}(I)\mathcal{E}(I^3) + 6\mathcal{E}(I)^2\mathcal{E}(I^2) - 3\mathcal{E}(I)^4 \end{aligned} \tag{4.26}$$

Here, $\mathcal{E}(I)$ is the expected or mean value of the respected distinct block. Normally block-size for the non-overlapping block may be any integer value. As pixel-level analysis is also important and if block-size is taken large value then some spliced pixels may not cover during localization, so here block-size has been taken as 2. As μ_4 sample values have been computed for all the blocks, its quartile values have been calculated. If the total number of samples is *n* then quartiles will be:

$$\mu_4{}^s = sort\ (\mu_4) \tag{4.27}$$

$\mu_4{}^s$ is sorted central moment values of *n* samples of image. Then the quartile values can be estimated as:

$$Q_2{}^1 = \begin{cases} \mu_4{}^s[n/2], & n\%2 \neq 0 \\ \dfrac{\mu_4{}^s\left[\frac{n-1}{2}\right] + \mu_4{}^s\left[\frac{n}{2}\right]}{2}, & n\%2 = 0 \end{cases} \quad (4.28)$$

$$Q_1{}^1 = median\,(\mu_4{}^s < Q_2) \quad (4.29)$$

$$Q_3{}^1 = median\,(\mu_4{}^s > Q_2) \quad (4.30)$$

Here, $Q_k{}^l$ represent the k[th] quartile value at the l[th] level. It is assumed that the spliced region in an image may be in the range of ~10 to ~15%, while quartiles will divide samples into quarters. So, its fourth quarter has been again divided into quarters from which the second quartile has been taken as a threshold (*th*) value of noise. Sample values above the threshold are considered to be spliced blocks and values below the threshold are considered to be non-spliced blocks.

$$th = Q_2{}^2 \quad (4.31)$$

$$\nu = \begin{cases} 0, & \mu_4 < th \\ 1, & \mu_4 \geq th \end{cases} \quad (4.32)$$

In equations 4.32, *th* represents the threshold value which is also the second level quartile value and ν represents binary inconsistent noise pattern.

4.3.2.2 Method and Model

Though there are still challenges in state-of-the-art techniques, image splicing detection using noise statistics has been used widely. Methods using noise variance estimation are based on second-order measures that work better in the case of the normal distribution of noise in an image. But the distribution of noise in a natural image may non-gaussian. In this section, a blind image splicing detection technique has been derived based on local noise statistics on higher-order statistics, which is best suitable for the description of the shape of the distribution (independent of location and scale). In this presented work, the fourth-order central moment value for the noise statistics is computed. The objective is to estimate noise statistics at distinct blocks of the image (see

Figure 4.15). Then these statistics are used to map the spliced region in an image. This methodology has been divided into the following four subsections:

- Preprocessing of Input Image
- Wavelet Transformation
- Noise statistics Estimation
- Post Processing and Result

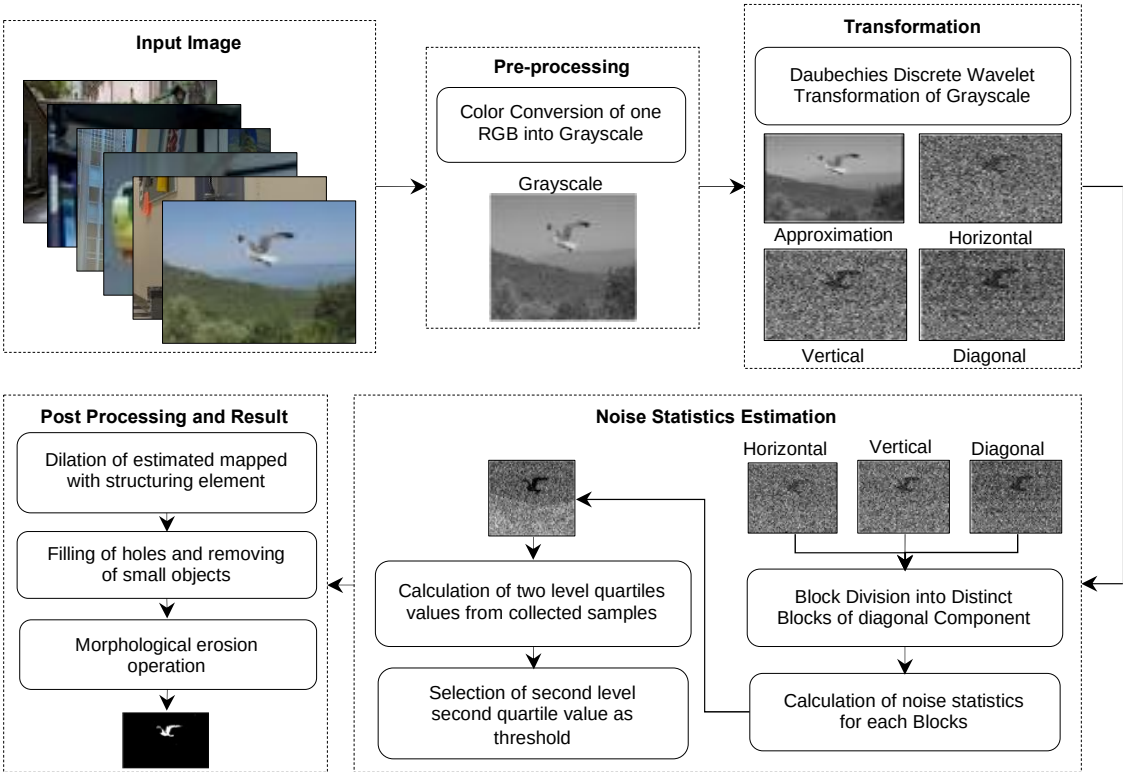


Figure 4.15: Graphical Abstract Representation of Proposed Approach (Overall method of spliced image detection and localization)

4.3.2.2.1 Pre-processing of Input Image

Pre-processing is a common term for operations with images at the lowest level of abstraction. This also involves the conversion of one-color space to another color space. In the presented work input image is taken from the dataset and given for the preprocessing i.e. color conversion. So, the given input image is in RGB color space and converted into Grayscale color space. The reason behind the color conversion is input in RGB color space has three color channels Red, Green and Blue individually and each has its intensity values. In this proposed work there is a requirement of noise statistics that do

not need all color channels. If all channels are taken, the whole process will take three times more to run. So, the given input image is first converted to grayscale and thus only one channel will be for further processing. As only luminance is significant for noise statistic estimation RGB color image is converted to a grayscale image. Weighted conversion of RGB to grayscale can be defined as:

$$G(i,j) = 0.2989 \times R(i,j) + 0.5870 \times G + 0.1140 \times B \quad (4.33)$$

Here, $G(i,j)$ is the grayscale intensity value of the image at the location *(i, j)* and R, G, B denotes red, green and blue channels of the color image. The following figure 4.16 shows the preprocessing step of the proposed approach.

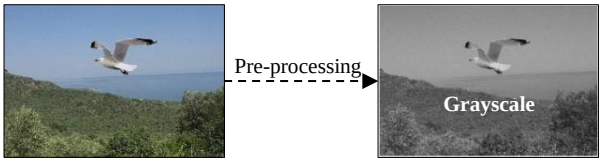


Figure 4.16: The Pre-processed result of Input Image (Conversion of a color image into Grayscale)

4.3.2.2.2 Wavelet Transformation

Wavelets are a general way to analyze and represent multiresolution images. Common applications of the wavelet transform of an image are image denoising [106] and edge detection [107]. The wavelet transform of an image has been widely used for the estimation and removal of noise. The presented approach is taken advantage of one of these approaches as wavelet coefficients have finer details about noise than in the spatial domain. The wavelet transform of an image is also used for edge detection. Noises are abrupt changes the same as an edge in the image.

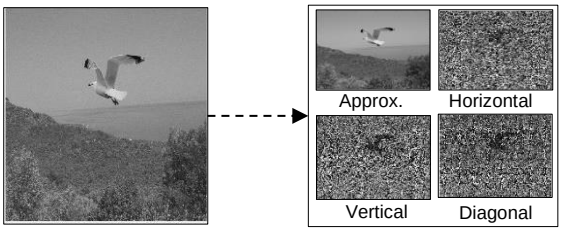


Figure 4.17: Result of Wavelet Transformed Image (Approximation and Detail Coefficients)

In the proposed technique discrete wavelet transformation (DWT) has been applied to the preprocessed grayscale image. DWT gives four components- one approximation and three detail components. Abrupt changes in neighborhood pixels of the image can be easily seen in detail components of the transformed image. Using coefficients of these detail components noise statistics can be calculated either by block-based methods or by region-based methods. In this work, the block-based technique has been used for noise estimation. Noise statistics should be based on neighborhood pixels, that's the reason why the block-based method has been used in the presented work. Assume a wavelet of the 1-D signal is $\psi(x)$ then its scaled and shifted version with scaling parameter $'s'$ and shifting parameter t can be represented by-

$$\psi_{s,t}(x) = \frac{1}{\sqrt{s}}\psi\left(\frac{x-t}{s}\right) \tag{4.34}$$

If $f(x)$ is a 1-D signal, then its wavelet transform can be defined as:

$$W_{s,t}(s,t) = \int_{-\infty}^{\infty} f(x)\,\psi_{s,t}(x)dx \tag{4.35}$$

So, discrete wavelet transforms an image which is 2-D signal can be defined as the weighted sum of wavelets and coarse coefficients [105]:

$$I(i,j) = \sum_{k} W_{\varphi}(j_0,k)\varphi_{j_0,k}(n) + \sum_{j=j_0}^{\infty}\sum_{k} W_{\psi}(j,k)\psi_{j,k}(n) \tag{4.36}$$

Here W_{φ} is approximation coefficients. W_{ψ} is details coefficients. $\psi_n(i,j)$ is scaling function. j_0 is an arbitrary starting scale and n=0,1,2 .. m. If f(x) is a spatial image then, approximation and details coefficients can be defined as:

$$W_{\varphi}(j_0,k) = \sum_x f(x)\varphi_{j_0,k}(n) \text{ and } W_{\psi}(j,k) = \sum_x f(x)\psi_{j,k}(n) \tag{4.37}$$

4.3.2.2.3 Noise Estimation

Estimation of noise statistics can be done using detailed coefficients of wavelet transformed images. State-of-the-art techniques are there for noise estimation using the variance in blocks of the image. These approaches use first and second order of measure statistics which is suitable for only gaussian signals. Real-life images contain non-gaussian noise in them. And second-order measures can't handle signals other than Gaussian signals, while higher-order statistic measures can handle non-gaussian signals. Using the noise estimation technique given in the above section, binary mapped values for the corresponding image are calculated (see Figure 4.18).

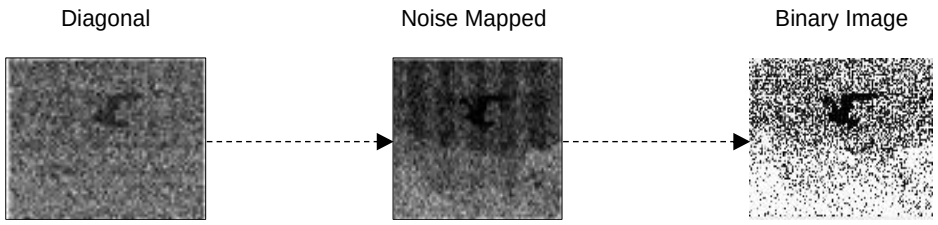


Figure 4.18: Noise Statistic Estimation of Diagonal Component of Discrete Wavelet Transformed Image

- Wavelet transformed diagonal component has been taken from all components.
- This diagonal component is got from the high pass filter of columns and rows. Then this diagonal component is divided into distinct blocks of block size 2x2. Here distinct means non-overlapping blocks. If the size of the image is $m \times n$ then the number of distinct blocks will be $\left(\frac{m}{2} \times \frac{n}{2}\right)$.
- Higher-order measure (using noise estimation given in the above section) calculation of each distinct block. The higher measure is defined as the fourth-order central moment (μ_4) of a 2x2 distinct block of the image. These values are stored in an array.
- Quartile statistics of the measured samples have been calculated. To calculate quartile values first measured samples array is sorted and then calculated first (Q_1), second (Q_2) and third (Q_3) quartile values of the sorted array.
- Resultant samples values are converted into binary values using the threshold (*th*) value chosen from the quartile value.

The pseudo-code of noise estimation using diagonal coefficients of the transformed image is given in Pseudo Code 1 (Noise Estimation).

Pseudo Code 1: Noise Estimation (noise_estimation)

input: *cD*: Diagonal Coefficient of Transformed Image, *BS*: 2 //Block Size
output: **N**: Resultant Noise Statistic Image
functions: DivideBlocks: Division of Input Image into BSxBS blocks
CalculateMoment: Calculate Central Moment
Count: To count the size of the input array
Sort: To Sort the given array
Search: Find the elements using a given logical operation
Median: Calculate the Median of the input sequence

```
1:  initialize cD,BS;
2:  Blocks = DivideBlocks (cD,BS);
3:  for each Block in Blocks
4:    FOCM = CalculateMoment (Block,4);
5:  end
6:  n = Count (FOCM);
7:  sFOCM = Sort (FOCM);
8:  if n%2! = 0
9:    q2 = sFOCM(n/2);
10: else
11:   q2 = [sFOCM((n-1)/2) + sFOCM(n/2)]/2;
12: end
13: q1 = Median (search (sFOCM < q2));
14: q3 = Median (search (sFCM > q2));
15: th = q3;
16: N = search (FOCM < th);
17: return N;
```

4.3.2.2.4 Post Processing

Post Processing is also referred to as image retouching. Since the forged region should be a combination of some successive blocks, an individual non-homogenous block may not be part of the spliced region. This is the reason why post-processing is more important in this proposed method. To refine the result contextual information may be additionally used. In this technique, the following steps are used to refine the result. The output of the post-processing is shown in Figure 4.19. Pixel value correction has been done by removing inaccurate pixels in the following manner-

- Dilation operation of the mapped image is performed using the following structuring element.

$$SE = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

- Hole filling is performed to fill the inaccurate pixels with the background pixel value.
- Removed small objects which are lesser than the assumed spliced region size.
- Erosion is performed to correct the border pixels of the spliced region.

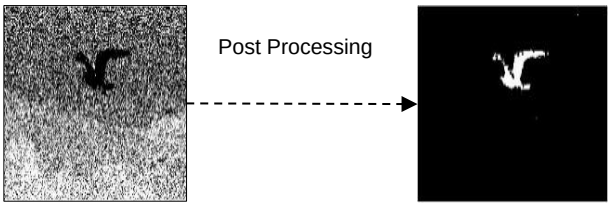


Figure 4.19: Result after post-processing (After Morphological Operations)

The pseudo-code of the presented method is also given below.

Pseudo Code 2: Spliced Image Localization

---

input: *I*: Input Image, *SE*= [1 0 1;0 1 0;1 0 1]

output: *R*: Resultant Localized Image

functions: rgb2grayscale: Convert Input RGB Image to Grayscale

wavelet_transform: Transform Input into Wavelet Domain

initialize *I*;

I_Gray = rgb2grayscale (*I*);
coeff[] = wavelet_transform (*I_Gray*);
noi_st = noise_estimation (coeff[3]);
dil_i = dilation (*noi_st*, *SE*);
rem_ob = remove_small_object(*dil_i*);
R = Erosion (*rem_ob*);

} Morphological Operations

return *R*;

4.3.2.3 Result Analysis and Discussion

This section provides the detail of the experiments performed for the proposed method. To evaluate the result for the proposed algorithm MATLAB R2017b tool has been used to program the algorithm. The tool was installed on the Linux Server desktop system with the system configuration of a Xeon processor and 16 GB RAM installed in it. The minimum memory requirement for the MATLAB R2017b is 2GB and any intel or AMD processor on which the experiment can be performed. The dataset should be diverse

in terms of its acquisition process, the format of the picture and the resolution of the image. Datasets that are used in experiments have a huge variation in all these terms and they are:

- Columbia Uncompressed Dataset [67]
- CASIA [65]
- IEEE IFS-TC Image Forensic Challenge Database [66]

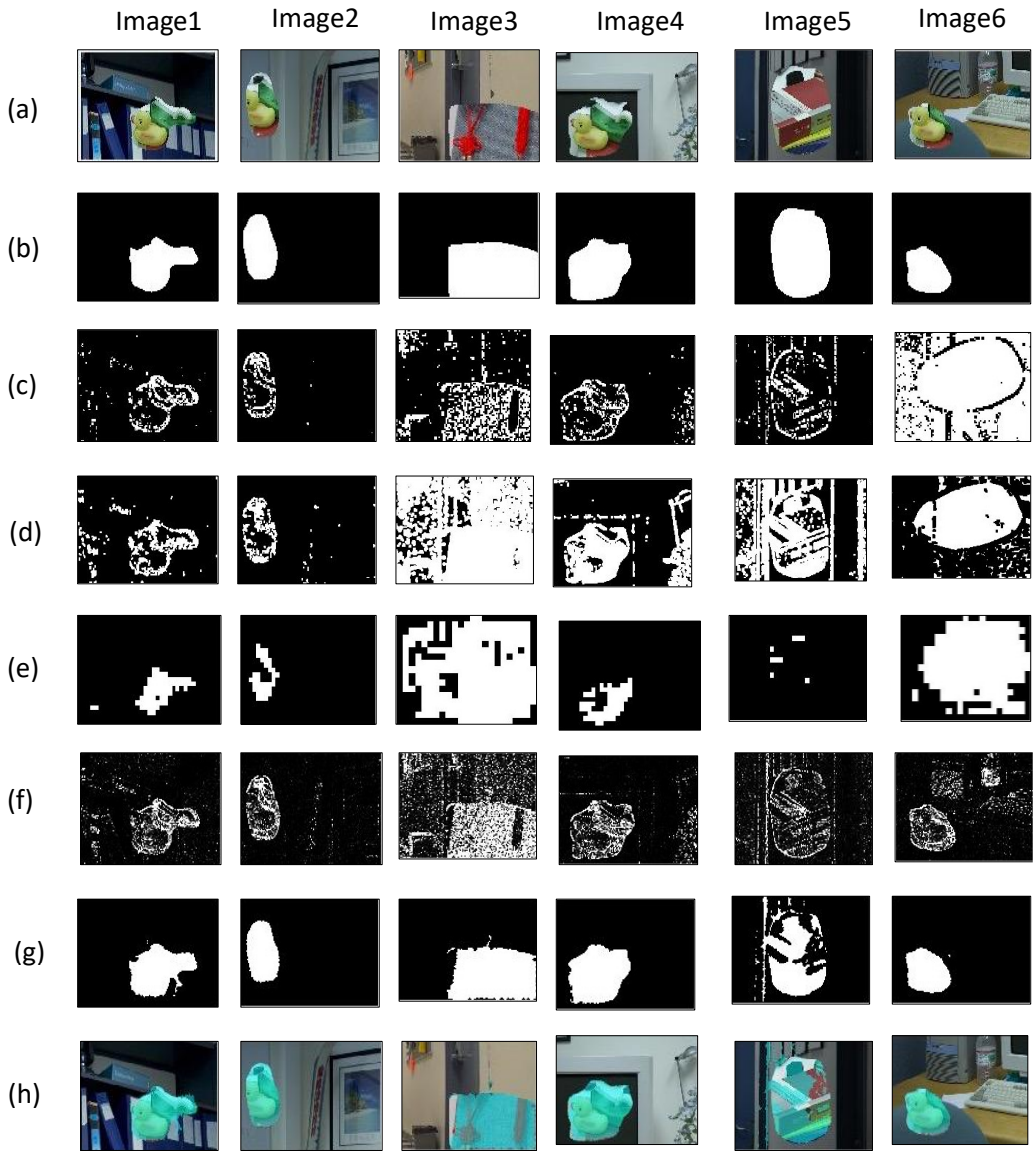


Figure 4.20: Result of Image Splicing detection and localization on Columbia uncompressed dataset (a) Test Image (b) Ground Truth Mask (c) Localized splice region result of BLNVS [13] *(d) Localized splice region result of PKNV* [14] *(e) Localized splice region result of NIBIF* [16] *(f) Noise Statistic Map of the given method (g) Localized Spliced Region from the Noise Map (h) Color Overlay of the spliced region on the RGB input Image*

The above-mentioned datasets are publicly available and already discussed in the theoretical background section of chapter 2. The proposed algorithm is compared with three similar state-of-the-art techniques [13], [14], [16]. For comparison, these state-of-the-art techniques are annotated as BLNVS [13], PKNV [14] and NIBIF [16] in short. For the demonstration purpose, some parameters were taken into consideration. As in the previous section, it is already mentioned that block size has been taken as 2 for noise estimation. The minimum spliced region is also assumed to be around 10 to 15% of the whole image size. The compared method PKNV [14] has not given any specific sequence of morphological operations for the localization of the image through the best suitable sequence of morphological operations that was used during the comparison demonstration. Parameters chosen for the state-of-the-art methods were based on their reported values in the methodology or experiment section of the published article. In the technique NIBIF [16], block size has been chosen as 32. Such parameter value doesn't give effective results in the case of the small size of images. This can be seen in the visual demonstration where the experiment was performed on the images of the Columbia Uncompressed Dataset. First, an experiment was performed on Columbia Uncompressed dataset [67].

Table 4.7: Comparison of the proposed algorithm with state-of-the-art techniques on evaluation metrics precision, recall, tnr and accuracy on Columbia Uncompressed dataset

Images	BLNVS [13]				PKNV [14]				NIBIF [16]				Proposed			
	p	*r*	*tnr*	*a*	*p*	*r*	*tnr*	*a*	*p*	*r*	*tnr*	*a*	*p*	*r*	*tnr*	*a*
Image1	0.970	0.559	0.997	0.932	0.589	0.476	0.943	0.876	0.677	0.331	0.973	0.879	0.911	0.965	0.984	0.981
Image2	1	0.439	1	0.936	0.811	0.504	0.985	0.931	0.862	0.348	0.993	0.921	0.990	0.967	0.998	0.995
Image3	0.410	0.882	0.384	0.547	0.368	0.971	0.209	0.454	0.675	0.476	0.891	0.758	0.980	0.907	0.991	0.964
Image4	1	0.411	1	0.888	0.554	0.834	0.839	0.838	0.654	0.287	0.963	0.834	0.979	0.954	0.995	0.987
Image5	1	0.050	1	0.724	0.393	0.717	0.547	0.596	0.623	0.234	0.941	0.736	0.878	0.634	0.964	0.868
Image6	0.700	0.987	0.6974	0.818	0.876	0.831	0.914	0.879	0.457	0.937	0.190	0.505	0.962	0.974	0.995	0.993

The performance of the proposed algorithm can be seen in Figure 4.20. This figure demonstrates the experimental result on different images of the Columbia Uncompressed dataset. The experimental results presented in the figure compare the result of the proposed work with other techniques. The first row of the figure represents the spliced images of the dataset, the second row represents the ground truth binary mask generated by the edge mask of the corresponding image. Experiment results of the method BLNVS [13], PKNV [14] and NIBIF [16] on these images are shown in the third, fourth and fifth rows. The estimated noise-mapped image of the proposed work is in the sixth row. The localized result and color overlay of the spliced region are presented in the seventh and eighth rows of the figure. From the visual results given in, it can be analyzed that the result of the proposed method gives a better localization result than other similar techniques. The reason for the worse results of NIBIF [16] is the parameter of block size for the estimation of noise. In this case, the block size is taken as 32 which is much larger for small size images and pixel-level comparison, in this case, gives the worse result. However, in the proposed method block size is taken much lesser than the compared method. Also, the sequence of post-processing operations is better than other similar techniques which give better result in the case of the proposed method.

Table 4.8: Comparison of the proposed algorithm with state-of-the-art techniques on evaluation metrics elapsed time (in seconds), csi, f1 and mcc on Columbia Uncompressed dataset

Images	BLNVS [13]				PKNV [14]				NIBIF [16]				Proposed			
	ET	*csi*	*f1*	*mcc*	*ET*	*csi*	*f1*	*mcc*	*ET*	*csi*	*f1*	*mcc*	*ET*	*csi*	*f1*	*mcc*
Image1	2.352	0.550	0.709	0.707	2.679	0.357	0.527	0.460	0.181	0.286	0.445	0.417	0.623	0.882	0.937	0.927
Image2	2.209	0.436	0.610	0.640	2.695	0.451	0.622	0.607	0.185	0.329	0.496	0.517	0.627	0.959	0.979	0.976
Image3	1.161	0.389	0.560	0.274	1.407	0.364	0.534	0.236	0.142	0.387	0.558	0.410	0.541	0.891	0.942	0.918
Image4	2.245	0.411	0.583	0.601	2.701	0.499	0.666	0.586	0.182	0.249	0.399	0.356	0.673	0.936	0.967	0.959
Image5	1.716	0.050	0.095	0.189	2.075	0.340	0.508	0.241	0.160	0.205	0.340	0.256	0.586	0.583	0.736	0.667
Image6	1.152	0.694	0.819	0.686	1.388	0.744	0.853	0.751	0.145	0.443	0.614	0.184	0.642	0.939	0.968	0.965

Table 4.7 compares the precision, recall, accuracy and true-negative rate(tnr) values with different techniques on different images of this dataset. This table shows the accuracy value of the proposed algorithm is higher than other techniques. Sometimes predicted pixels as positive in result image may more than positive pixels in the ground truth. This case also misleads the result and here the precision value of BLNVS [13] is more than the proposed work which follows that case. Similarly, if the predicted positive pixels in the resulting image are more than ground truth positive pixels this will give a higher recall value and mislead the result which can be seen in PKNV [14].

Though BLNVS [13] gives better *precision* and *tnr* values and with some images, PKNV [14] gives better recall values, the accuracy of the proposed work is better than state-of-the-art methods with all images. The result of NIBIF [16] is not better due to bigger block partitioning and lower resolution of the image in the case of images of Columbia Uncompressed Dataset. For the unbiased result Table 4.8 is given where evaluation metrics *csi, f1-score* and *mcc* values are shown of the experimental result. The elapsed time (in second) taken by the methods are also presented in the table which clearly shows that the given method can work in real-time also. The above-mentioned Table 4.8 shows unbiased metrics of the experimental result. The elapsed time for a single image is taken by the proposed algorithm is lesser than 1 second for images of Columbia Uncompressed Dataset (lower resolution). In such cases, NIBIF [16] takes lesser elapsed time than the proposed algorithm because of its larger block size during the noise estimation. This is the reason why this state-of-the-art doesn't give better accuracy or f1-score value for the lower resolution of the image. Table 4.9 compares the average elapsed time, precision, recall, tnr, accuracy, csi, f1-score and mcc value of the proposed method with the state-of-the-art on Columbia uncompressed dataset.

Table 4.9: Average Elapsed Time, precision, recall, tnr, accuracy, csi, f1-score and mcc on Columbia Uncompressed dataset

Method	*ET*	*p*	*r*	*tnr*	*a*	*csi*	*f1*	*mcc*
BLNVS [13]	1.8063	0.8469	0.5552	0.8464	0.8079	0.4225	0.5632	0.5167
PKNV [14]	2.1581	0.5990	0.7227	0.7400	0.7628	0.4597	0.6185	0.4804
NIBIF [16]	0.1663	0.6581	0.4360	0.8258	0.7725	0.3171	0.4758	0.3573
Proposed	0.6157	0.9508	0.9007	0.9883	0.9651	0.8654	0.9221	0.9024

The proposed algorithm has been also compared with the state-of-the-art techniques on publicly available dataset CASIA and IEEE IFS-TC Image forensics Challenge Dataset. Figure 4.21 demonstrates the experimental result on different images of CAISA and IEEE IFS-TC Image forensics Challenge datasets. The experimental results presented in the following figure compare the result of the proposed work with other techniques BLNVS [13], PKNV [14] and NIBIF [16]. The first row of the figure represents the spliced images of the dataset, the second row represents the ground truth binary mask generated by the edge mask of the corresponding image. Experiment results of the method BLNVS [13], PKNV [14] and NIBIF [16] on these images are shown in the third, fourth and fifth rows. The estimated noise-mapped image of the proposed work is in the sixth row. The localized result and color overlay of the spliced region are presented in the seventh and eighth rows of the figure. BLNVS [13] method partition the image into larger blocks and calculates variance and sharpness at that block level. This may give false detection in case of images having low resolution. This can be seen in Figure 4.21. The second image in the above figure is an image of the CASIA dataset. This image has a low resolution of 384×256 pixels and BLNVS [13] method takes a block size of 32. In such a case remaining blocks will be 12×8 which is not enough to estimate noise variance locally. Although other state-of-the-art techniques didn't report parameters and sequence of morphological operations for the post-processing operation, the best possible sequence was tried to program for the experimental purpose. The state-of-the-art

techniques give satisfactory results on the IEEE IFS dataset but for the CASIA dataset, they don't provide satisfactory results.

Figure 4.21: Result of Image Splicing detection and localization on CAISA and IEEE IFS-TC Image forensics Challenge datasets (a) Test Image (b) Ground Truth Mask (c) Localized splice region result of BLNVS [13] (d) Localized splice region result of PKNV [14] (e) Localized splice region result of NIBIF [16] (f) Noise Statistic Map of the given method (g) Localized Spliced Region from the Noise Map (h) Color Overlay of the spliced region on the RGB input Image

Table 4.10 compares the precision, recall, accuracy and *tnr* values with different techniques on different images of CASIA and IEEE IFS dataset. This table shows the precision, accuracy and f1-score value of the proposed algorithm is higher than other techniques. If the predicted positive pixels in the resulting image are more than ground truth positive pixels this will give a higher recall value and mislead the result.

Table 4.10: Comparison of the proposed algorithm with state-of-the-art techniques on evaluation metrics precision, recall, tnr and accuracy on IEEE IFS and CASIA dataset

Images	BLNVS [13]				PKNV [14]				NIBIF [16]				Proposed			
	p	*r*	*tnr*	*a*	*p*	*r*	*tnr*	*a*	*p*	*r*	*tnr*	*a*	*p*	*r*	*tnr*	*a*
Image1	0.1304	0.2500	0.6923	0.6234	0.1132	1	0.0070	0.1187	0.0070	0.0070	0.4354	0.3997	0.3496	0.8938	0.7899	0.8016
Image2	0.1495	1	0.8489	0.8528	0.1146	0.9319	0.8143	0.8173	0.8143	0.8143	0.6107	0.6202	0.8883	0.9684	0.9968	0.9961
Image3	0.1238	0.6369	0.7475	0.7416	0.4398	0.5470	0.9612	0.9393	0.9612	0.9612	0.9964	0.9541	0.9922	0.5827	0.9997	0.9778
Image4	0.0639	0.4405	0.8116	0.8011	0.2639	0.6678	0.9442	0.9362	0.9442	0.9442	0.9687	0.9554	0.9682	0.7133	0.9993	0.9910
Image5	0.7451	0.2879	0.9980	0.9841	0.0038	0.0712	0.6322	0.6213	0.6322	0.6322	0.9681	0.9574	0.3773	0.8493	0.9722	0.9698
Image6	0.8720	0.4449	0.9941	0.9487	0.8794	0.9695	0.9880	0.9865	0.9880	0.9880	0.3765	0.4272	0.9203	0.9890	0.9923	0.9920

It can be seen from Table 4.11, that the recall values of other techniques give better results than the proposed algorithm. As it is already discussed that if the predicted positive pixels in the resulting image are more than ground truth positive pixels this will give a higher recall value and mislead the result. The unbiased metrics csi, f1-score and mcc values are compared for the images of CASIA and IEEE IFS dataset in table 4.11. This table also gives elapsed time (time taken by the algorithm in second) of the proposed algorithm and other state-of-the-art techniques for the single image. The table compares the time taken by algorithms to localize the spliced regions.

Table 4.11: Comparison of the proposed algorithm with state-of-the-art techniques on evaluation metrics elapsed time (in seconds), csi, f1 and mcc on CASIA and IEEE IFS dataset

Images	BLNVS [13]				PKNV [14]				NIBIF [16]				Proposed			
	ET	*csi*	*f1*	*mcc*	*ET*	*csi*	*f1*	*mcc*	*ET*	*csi*	*f1*	*mcc*	*ET*	*csi*	*f1*	*mcc*
Image1	0.5589	0.0938	0.1714	-0.045	0.3671	0.1132	0.2033	0.0281	0.1264	0.0223	0.0436	-0.281	0.4917	0.3357	0.5026	0.4771
Image2	14.298	0.1495	0.2602	0.3563	11.273	0.1136	0.2041	0.2897	0.4864	0.0619	0.1165	0.1904	1.1138	0.8633	0.9266	0.9255
Image3	10.604	0.1156	0.2073	0.1934	9.2725	0.3224	0.4876	0.4588	0.4113	0.1807	0.3061	0.3652	1.0893	0.5800	0.7342	0.7515
Image4	7.6339	0.0591	0.1116	0.1055	9.2122	0.2333	0.3783	0.3939	0.4136	0.2500	0.4000	0.3880	1.0741	0.6969	0.8214	0.8269
Image5	15.534	0.2621	0.4153	0.4572	20.862	0.0036	0.0073	-0.085	0.8064	0.1597	0.2753	0.2730	1.8785	0.3536	0.5225	0.5545
Image6	7.4494	0.4176	0.5892	0.6015	9.1978	0.8558	0.9223	0.9162	0.4139	0.1250	0.2222	0.2113	1.3812	0.9110	0.9534	0.9498

Although the table shows that NIBIF [16] takes lesser time to localize the spliced region than the proposed algorithm, the proposed algorithm gives a much higher f1-score and mcc values than NIBIF [16]. The time taken by the proposed algorithm is not much higher than the NIBIF [16] algorithm and can work in real-time also. Table 4.12 compares the average elapsed time, precision, recall, tnr, accuracy, csi, f1-score and mcc value of the proposed method with state-of-the-art on CUD.

Table 4.12: Average Elapsed Time, precision, recall, tnr, accuracy, csi, f1-score and mcc on IEEE IFS and CASIA dataset

Method	*ET*	*p*	*r*	*tnr*	*a*	*csi*	*f1*	*mcc*
BLNVS [13]	9.3465	0.3474	0.5100	0.8487	0.8253	0.1829	0.2925	0.2780
PKNV [14]	10.0310	0.3024	0.6979	0.7244	0.7365	0.2736	0.3671	0.3336
NIBIF [16]	0.4430	0.2497	0.5351	0.7260	0.7190	0.1332	0.2273	0.1911
Proposed	1.1714	0.7493	0.8327	0.9584	0.9547	0.6234	0.7435	0.7476

The proposed method and state-of-the-art techniques are performed on various images of publicly available datasets. Above mentioned tables demonstrate the visible results. These datasets also comprise of authentic images. This may also be possible that the proposed algorithm can localize authentic regions in an authentic image. To prove that the proposed algorithm shows only the spliced regions in spliced images, the proposed algorithm is demonstrated on two authentic images of the CUD. Figure 4.22 presents the authentic image, estimated noise map and localized image.

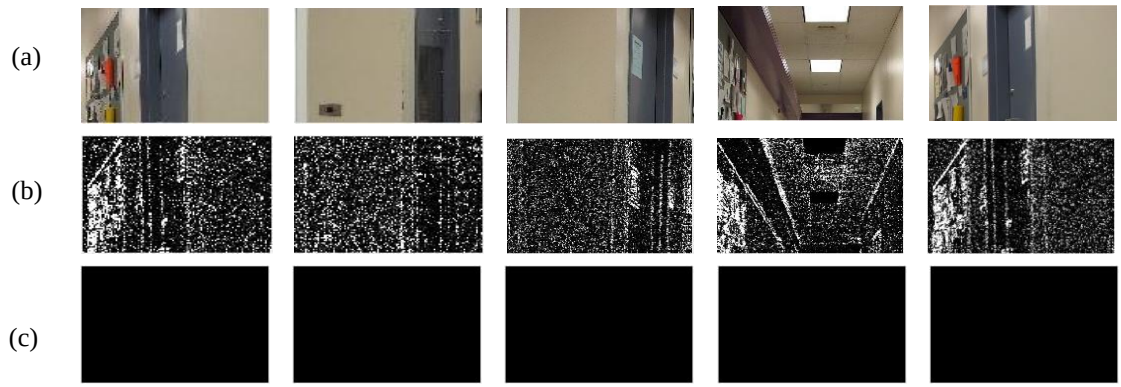


Figure 4.22: Proof of the proposed algorithm on authentic images of datasets (a) Natural Color Image (b) Noise Mapped Image (c) Localized Spliced Region

The first row of the above figure contains an authentic, natural RGB color image of the Columbia Uncompressed Dataset. The second row represents the estimated noise mapped image corresponding to the authentic image and the last row represents the resultant localized image after post-processing operation. In the presented last row, there is no such spliced region that can be seen which proves that the proposed algorithm works well with both authentic and spliced images. Except for the quantitative results compared in the tables, graphical comparison results are also shown with the help of the line graph plot.

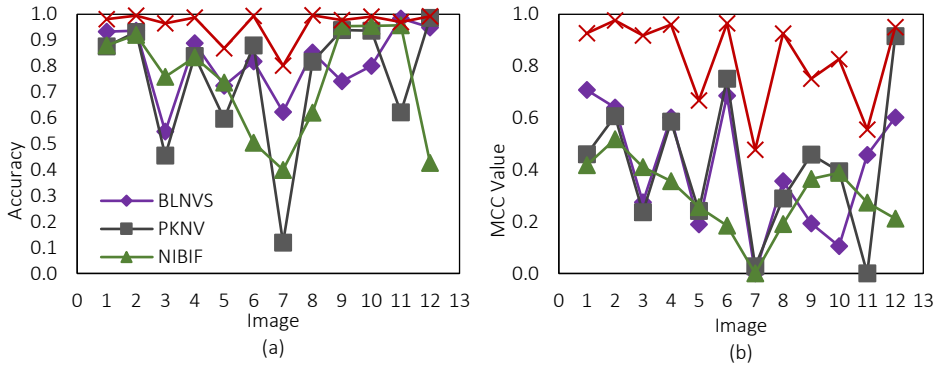
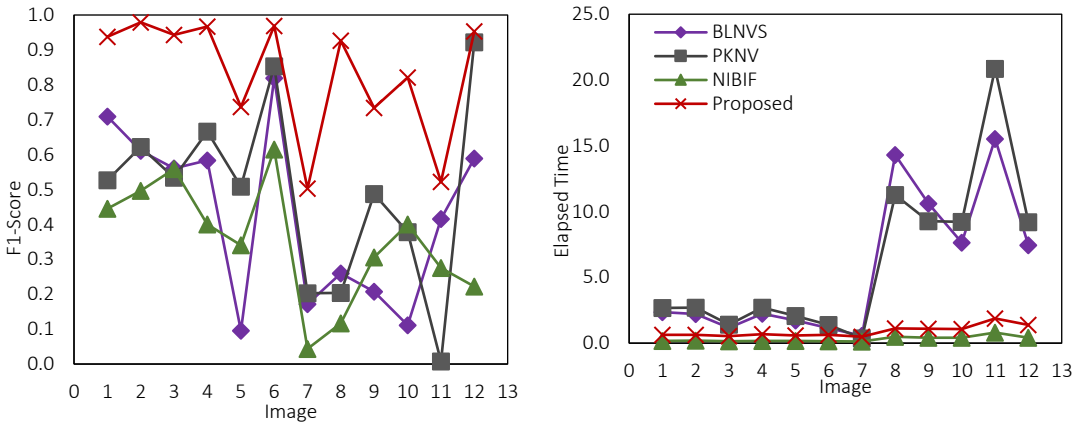


Figure 4.23: (a) Comparison of the Accuracy value of the proposed work with other techniques (b) Comparison of Matthews Correlation Coefficient value of the proposed work with other techniques

Figure 4.23 (a) compares the *accuracy* result on different images. Figure 4.23 (b) presents the comparison result of *mcc* values. F1-score values of different methods are compared in Figure 4.24(a) and Figure 4.24(b) shows a comparison of the elapsed time of the proposed algorithm with three state-of-the-art techniques.

A comparison of only precision and recall values is not enough. These performance matrices sometimes mislead the results. Accuracy may also mislead the result as sometimes accuracy gives biased results. CSI, F1-score and MCC are required in those cases, as the f1-score evaluation metric gives the harmonic mean of precision and recall values. This is a better performance measure to compare the result. The above figure

4.24(a) presents the comparison of the f1-score of the proposed work with other state-of-the-art techniques on different images. Figure 4.24(b) is the comparison of the time taken by the proposed work with state-of-the-art techniques on different images. From the above-presented experiments, it can be observed that the proposed method works better than other state-of-the-art techniques. The elapsed time by the proposed method for maximum image lesser than other algorithms. While NIBIF takes lesser time this algorithm also takes block size as 32 which leads to false results. The elapsed time taken by the proposed algorithms shows that the method is time-efficient.



(a) (b)

Figure 4.24: (a) Comparison of F1-Score value of proposed work with other techniques (b) Comparison of Elapsed Time of proposed work with other techniques

4.4 Summary

Spliced images are another type of digital image forgery in which regions of two or more images are merged. Images are combined in such a way that human eyes can't differentiate tampered regions. A lot of methods are already reported for spliced image detection in various literature. Some data-driven techniques are there to detect an image whether the image is forged or non-forged. These techniques are unable to localize the tampered region in the forged image. Other techniques are based on intrinsic footprint as noise. These suffer from limitations like automatic localization technique, estimation of

qualified noise patterns etc. This chapter provides two different spliced image detection and localization techniques based on data-driven and noise patterns.

A data-driven technique is proposed in the first method that can differentiate spliced and non-spliced images using a logistic regression method. For this, a combination of four features has been extracted from images for feature vector and classified using a logistic regression technique. The main contributions of the paper are the most relevant feature set that can be used for a more correct classification of the spliced image and rigorous comparison of the proposed system with state-of-the-art techniques. Evaluation metrics of the proposed model are measured on three different datasets: CASIA v1.0, CASIA v2.0, COLUMBIA. In these datasets, images are very different in texture, nature and scenes. In such different types of images detection of splicing forged images is a challenging task.

A forged region localization technique is used in the second method. In this work, the noise estimation is done using fourth-order central moment values locally which estimate the non-gaussian noise of the image. The quartile value used in the method is for the threshold value of the assumed spliced region. Now, these values are mapped for the localization of the spliced region using a sequence of best suitable morphological operations. An experimental analysis of the proposed technique has been done on the images of three publicly available datasets. The result of the proposed work is also compared with state-of-the-art techniques on the pixel level. Another measure in this work is taken as elapsed time which tells that how much time the proposed algorithms take. The comparison of these measures shows that the proposed algorithm is efficient.