electronics

MDPI

Article

A Robust Forgery Detection Method for Copy–Move and Splicing Attacks in Images

Mohammad Manzurul Islam *, Gour Karmakar, Joarder Kamruzzaman and Manzur Murshed

School of Engineering, IT and Physical Sciences, Federation University Australia, Churchill, Victoria 3842, Australia; gour.karmakar@federation.edu.au (G.K.); joarder.kamruzzaman@federation.edu.au (J.K.); manzur.murshed@federation.edu.au (M.M.)

* Correspondence: mm.islam@federation.edu.au

Received: 21 August 2020; Accepted: 9 September 2020; Published: 12 September 2020

Abstract: Internet of Things (IoT) image sensors, social media, and smartphones generate huge volumes of digital images every day. Easy availability and usability of photo editing tools have made forgery attacks, primarily splicing and copy–move attacks, effortless, causing cybercrimes to be on the rise. While several models have been proposed in the literature for detecting these attacks, the robustness of those models has not been investigated when (i) a low number of tampered images are available for model building or (ii) images from IoT sensors are distorted due to image rotation or scaling caused by unwanted or unexpected changes in sensors' physical set-up. Moreover, further improvement in detection accuracy is needed for real-word security management systems. To address these limitations, in this paper, an innovative image forgery detection method has been proposed based on Discrete Cosine Transformation (DCT) and Local Binary Pattern (LBP) and a new feature extraction method using the mean operator. First, images are divided into non-overlapping fixed size blocks and 2D block DCT is applied to capture changes due to image forgery. Then LBP is applied to the magnitude of the DCT array to enhance forgery artifacts. Finally, the mean value of a particular cell across all LBP blocks is computed, which yields a fixed number of features and presents a more computationally efficient method. Using Support Vector Machine (SVM), the proposed method has been extensively tested on four well known publicly available gray scale and color image forgery datasets, and additionally on an IoT based image forgery dataset that we built. Experimental results reveal the superiority of our proposed method over recent state-of-the-art methods in terms of widely used performance metrics and computational time and demonstrate robustness against low availability of forged training samples.

Keywords: Internet of Things; security; image forensics; machine learning models; feature extraction

1. Introduction

Today's digital lives have been evolving around the concept of the Internet of Things (IoT) smart objects. A recent report (June 2018) by Ericsson [1] projected that there will be 31.4 billion IoT devices connected to the Internet by 2023, which means each person living on earth will have approximately four IoT devices on average. Everyday objects such as home appliances, security devices, vehicles, computing devices, visual sensors, wearable sensors are becoming smarter and connected to the Internet, and they enable data sharing with each other to perform decision fusion for improved accuracy of decision. Among them visual sensors like camera play a vital role in the physical as well as cyberspace security and surveillance. Modern social media platforms such as Snapchat, Twitter, Facebook, Instagram and online digital news media are producing millions of digital images everyday. People tend to trust and rely on digital image contents more than written articles. However, easy to use

photo editing software and tools have been available in recent days to alter image contents, and anyone can produce fake images which appears to be quite natural and authentic. Our biological visual system cannot trace these forgeries as the fake images leave no identifiable footprint of image tampering to naked eyes. As a result, disrupting operational and decision making process, spreading false propaganda and hiding facts have become a common practice for the digital criminals. Among all image tampering techniques, copy–move and splicing are the most common and notorious digital image forgery operation [2]. In the copy–move forgery, one or more portion of the host image is copied and then pasted on different locations of the same image. In contrast, splicing forgery copies one or more portion from one image and then pasts on a completely different image. In this research article, we use 'image forgery' or 'image tampering' to denote copy–move and splicing attacks.

A single image can express a complex idea very easily as 'a picture is worth a thousand words'. Image tampering may convey a false impression which may lead to disastrous consequences [3–5]. An AP photo journalist Markus Schreiber captured the photo in Figure 1a on the very first day of G-20 summit in Germany [6] in 2017. Later, the image was altered and uploaded to Facebook by a Russian journalist [7]. By the time he deleted the post from Facebook, the image was already shared thousands of times in different social media and news portals which caused confusion and generated debate around the world. Similarly, a digitally forged image can convey wrong message to the world political leaders which may end up in forcing them to make wrong decision, political steps or even start a nuclear war.

(**a**) (**b**)

Figure 1. Image splicing example: (**a**) Original and (**b**) fake.

Though image tampering leaves no visible traces to human eyes, it can be identified through the state-of-the-art image forgery identification techniques. Based on the type of image domain used to extract the feature set(s), existing forgery detection methods available in literature can be divided into three main categories: (i) Spatial domain based techniques [8,9]; (ii) frequency domain based techniques [10,11]; and (ii) hybrid techniques [12,13]. Statistical artifacts introduced by image forgery can be identified more explicitly by exploiting statistical measures of spatial domain information. However, pixel value and boundary changes can be more accurately captured by leveraging the frequency domain information as shown in recent works introduced by [11,14]. However, the performance of these methods has not been analyzed on Internet of Things (IoT) image sensor data. For utilizing the benefits of spatial, temporal and frequency domains, there exist many hybrid techniques in the literature. The literature shows researchers developed the forgery detection techniques for color and grayscale images separately, however, most of these techniques do not necessarily perform equally well for both types of images. Most techniques require a huge number of features and hence, feature selection becomes essential to reduce the computational complexity [15]. Moreover, to the best of our knowledge, none of the existing methods is evaluated for an IoT image dataset.

To address these issues, in this paper, we introduce a new image forgery detection technique and evaluate it with extensive experimentation. The main contributions of the paper are summarized as follows:

- Our proposed method is highly robust as it is capable of detecting image forgery even when a low number of forgery samples (accuracy of 77.01% vs. 61.17% for 3% forgery samples in training set for our proposed method vs. method in [10]) are available in training set for FBDDF (Federation–Berkeley Deep Drive Forgery) dataset. This level of performance vindicates our proposed detection system is more suitable for IoT image forgery detection.
- We introduce an innovative feature extraction method that reduces the impact of translation, rotation, and scale on feature extraction, and inherently yields a fixed low dimensional feature set presenting a more computationally efficient system. Another important aspect of this technique is that the extracted feature set is equally applicable for both types of images which is justified by the reported results using the benchmark datasets for color and gray level images.
- We develop a new dataset named FBDDF (Federation–Berkeley Deep Drive Forgery) using Berkeley's Deep Drive dataset [16], one of the widely used dataset for IoT based self-driving vehicular vision research. FBDDF fills-in the gap for a benchmark dataset for IoT around image forgery research.
- Our proposed method outperforms other image forgery detection techniques when tested on five publicly available and widely used datasets. The use of DCT, LBP, and other processing steps to extract features in our method which, when trained with SVM, leads to better detection performance (up to 13.20% improvement across the datasets).

The rest of the paper is organized as follows. Section 2 reports the related works. Section 3 explains our proposed method in detail. Section 4 discusses the newly developed FBDDF dataset and experimental results. It also compares our method with existing methods. Finally, Section 5 concludes the article.

2. Related Works

All the methods proposed in the literature to identify image tampering mainly differ in identifying and modelling the structural and statistical changes that occur due to image forgery. Considering the image domain information from which features are extracted, we can broadly classify the image forgery detection techniques into three categories: (A) Spatial domain based techniques that compute the features using the value of a pixel and its location information [8,9]; (B) frequency domain based techniques where features are calculated exploiting the frequency of image pixels [10,11]; and (C) hybrid techniques that use the combination of features comprising spatial, frequency, wavelet (time-frequency) and other image domain information [12,13]. Note, since Discrete Wavelet Transformation (DWT) uses both spatial (location) and frequency domain information and provides temporal resolution, all techniques that utilize DWT based features have been classified into the hybrid group. In the following sections, we discuss the existing and contemporary image forgery detection methods belonging to these categories.

2.1. Spatial Domain Based Techniques

Since a human can easily perceive image using pixel values, earlier image processing techniques use spatial domain information only. Thus, it led to the development of many forgery detection techniques exploiting only spatial domain information. Hsu and Chang [17] proposed a method that uses camera response function and its geometry invariants in a semi-automatic approach. This method requires manual labelling of suspicious segments of the tampered image and hence, not workable for real time applications. Later, they improved their work by integrating automatic segmentation [18]. However, automatic segmentation is also limited to its efficacy as there are huge number of natural images having vast variations among them. In [8], Johnson and Farid introduced a technique that exploits the lighting inconsistencies between forged and unaffected areas of the image. However, this approach does not provide satisfactory result if the host image and the spliced section have similar lighting condition. Wang et al. [19] presented a method for image tampering detection by extracting

features from chroma space of an image using thresholded edge information as a finite-state Markov chain and characterized it using one-step transition probability matrix. Their detection accuracy was higher (95.6%) than that of [12,13] for CASIA 2 [20]. However, to achieve better output from SVM classifier, they randomly selected 5123 authentic images (out of 7491) from CASIA 2 to make a balance with given 5123 forged images, which in real life application is not practical. Besides, the effectiveness of this technique also depends on intuitively selected value for thresholding edges [21] explored pixel correlation anomalies and coherence introduced by splicing operation in a host image. They proposed a mechanism adopting the run-length and edge statistics of an image. Since splicing adds extra edges to an image, instead of run-length image pixels, this model was later improved by He et al. [9] using an approximate run-length (the run-length of edges). First, the authors derived run-lengths on edge pixels to reduce computational cost of calculating run-lengths of the entire image. Then, unlike [21] (who calculated run-lengths along 0, 45, 90 and 135 degree orientation), they only considered run-length computation along the respective gradient direction of individual edge pixel. Finally, a threshold was selected to obtain approximate run-lengths and hence, they improved the method of [21] in terms of feature dimensionality, computational cost and detection accuracy (80.58%). Zhao et al. [22] proposed a method by extracting features from gray level run-length number (RLRN) vectors having different directions and de-correlated chrominance channels. Their investigation revealed that RLRN performed better in chrominance channel than in luminance channel or RGB space for image tampering detection. They attained 85% and 94.7% detection accuracy on two benchmark image forgery datasets, namely Columbia Color [17] and CASIA 1 [20] respectively. Some researchers adopted different texture descriptors like LBP and WLD (Weber Local Descriptor) to detect image tampering. Since WLD appears to be an robust texture descriptor, Hussain et al. [23] adopted multi-resolution WLD on chroma component to extract features and reported detection accuracy of 91.52% and 93.33% for copy–move and splicing attack respectively on CASIA 1. Another work [24] by the same authors compared multiscale LBP with multiscale WLD and attained superior result using WLD (90.97% for copy–move, 94.29% for splicing) than LBP (85.83% for copy–move, 90.48% for splicing) on CASIA 1. However, they gained slight improvement of detection accuracy for splicing attack and decrement for copy–move attack on the same dataset. The main problem of multiscale WLD is that the feature dimension is very high as dimensionality increases proportionally to the number of scales considered. As a consequence, multiscale WLD becomes computationally expensive and may be trapped by the curse of feature dimensionality [25]. Since image forgery not only changes the pixel values but also the edges of an image, the frequency domain can represent these changes in a more quantitative way than the spatial domain. For this reason, over time, many techniques have been evolved that are presented in the following section.

2.2. Frequency Domain Based Techniques

Alahmadi et al. [10] adopted both LBP and DCT to detect image forgery and attained promising result. The authors first applied LBP on blocks of image chroma channels followed by 2D-DCT on those individual blocks. For each DCT coefficient, they estimated a feature as the standard deviations of that specifically positioned DCT coefficients containing in all blocks. Many studies [10,22,26–28] show the improved performance of splicing and copy–move attacks detection using the chrominance channels than the luminance channel and gray scale image. The main drawback of the approach introduced in [10] is the application of LBP operator before applying DCT transformation as LBP eliminates some of the subtle artifacts introduced by an image forgery. To address this issue and enhance the performance, Islam et al. [11,14] proposed a new forgery detection system based on applying DCT first, and then, LBP. Islam et al.'s method performs better than existing methods in respect of detection accuracy, however, the study lacks extensive evaluation and comparison with other competing methods and particularly experimentation with IoT data. In this method, the computational complexity analysis has also not been performed. A recent work introduced by Kanwal et al. [29] presents three different models for image forgery detection based on LBP, local ternary pattern (LTP), enhanced LTP (ELTP)

and fast fourier transform (FFT). They applied these techniques on 3 by 3 overlapped blocks of different chrominance channels and compared the results among their suggested three models. They concluded that the model where they integrated FFT with ELTP operator yields best detection accuracy (88.62% on CASIA 1) and thus, more suitable for forgery detection. Furthermore, up to our knowledge, no existing forgery detection techniques available in the literature including [10,11,14] report how well their method performs when only a low number of forgery samples are available to train the classifier.

Each image domain has its ability to capture specific information. For example, pixel intensity changes (e.g., edges) created by splicing can be captured in a better way in the frequency domain than the spatial domain. On the other hand, statistical artifacts introduced by splicing can be identified more explicitly by exploiting statistical measures (e.g., second order statistics) of spatial domain information than that of frequency domain. For leveraging the merits of other domain information, a few approaches have been developed considering features extracted from multiple domains. Those works are discussed in the following section.

2.3. Hybrid Techniques

In [12], Shi et al. proposed a natural image model where intra-block DCT Markov random process based features, and statistical moment features in both frequency and spatial domains are extracted. The main strength of the proposed approach is the utilization of Markov features along with the statistical moment features. In general, forgery increases high-frequency components. Therefore, for calculating the transition probability matrix, the use of the same intuitive threshold for all DCT coefficients limits its effectiveness. Besides, some of the subtle changes occurred by image forgery can heavily influence the coefficients of certain frequency components. The threshold used in the Markov transition probability calculation cannot capture these coefficient changes well. For capturing the residual correlation (the correlation cannot the captured by [12]) of image pixels and exploiting the benefits of multi-resolution analysis of DWT, He et al. [13] presented a splicing detection model by using both intra-block and inter-block Markov features in both DWT and DCT domains. The main differences of this model compared with [12] are: (i) Besides, intra-block, inter-block Markov features are used, (ii) along with DCT, DWT is also used, and (iii) moment-based features are not utilized. Its main drawbacks are: (i) Markov features between DCT and DWT are correlated, (ii) the number of subbands needed appears to be high to obtain the wavelet coefficients in different positions, orientations, and scales, yielding very large feature dimension, and (iii) the requirement of feature selection also bounds its performance. Using only wavelet transformation information, a color image forgery technique has been articulated in [30]. Since Steerable Pyramid Transform (SPT) is a superior wavelet decomposition technique, SPT is used to derive a number of wavelet subbands of different scales having different orientations for both Cb and Cr color spaces. LBP operator is applied to the coefficients of each subband and then, LBP histogram features are used for the SVM classification. However, how many scales and how many orientations of each scale are appropriate for detecting all types of forgeries in all images still remain unanswered. Using LBP and DWT with Haar wavelet, another color image splicing detection technique has been introduced [15]. In contrast to [30], LBP is applied to the image before DWT, which suppresses many important pixel values required for detecting subtle changes for splicing. For effective feature selection at reduced computational complexity, they employed Principle Component Analysis (PCA).

Current literature indicates that color and grayscale image forgery techniques have evolved independently over time. This is because, for the effective forgery detection, the feature set has to be more sensitive to subtle changes for Cb and Cr color spaces than that for grayscale images. This demands an approach that is equally effective for both color and grayscale images. So far, none of the existing techniques is evaluated using IoT or Industrial IoT (IIoT) sensor data. It is essential to protect IoT/IIoT based systems driving industrial automation from cyber attacks. As expected, the performance of a forgery detection technique heavily depends on domain(s) information and image manipulation operator(s), and their usage sequence order. Finally, since, feature selection is a

challenging research issue, an image forgery technique should adopt an approach that inherently limits the number of features and thus avoids it. For addressing all of these issues, in this paper, we introduce an innovative method that is presented in the following section.

3. Proposed Method

The image forgery detection system is usually a binary decision problem i.e., identifying whether an image is authentic or forged. The system requires a pre-processing step to extract features that are followed by a classification step. The statistical and structural changes because of tampering are reflected by the features of an image. Therefore, an effective feature extraction technique is highly important, which is the main focus of our proposed system. The extracted features are fed into a binary classifier (e.g., machine learning model) to identify authentic and forged images. Figure 2 portrays the different key components and their operational sequence of our proposed method.

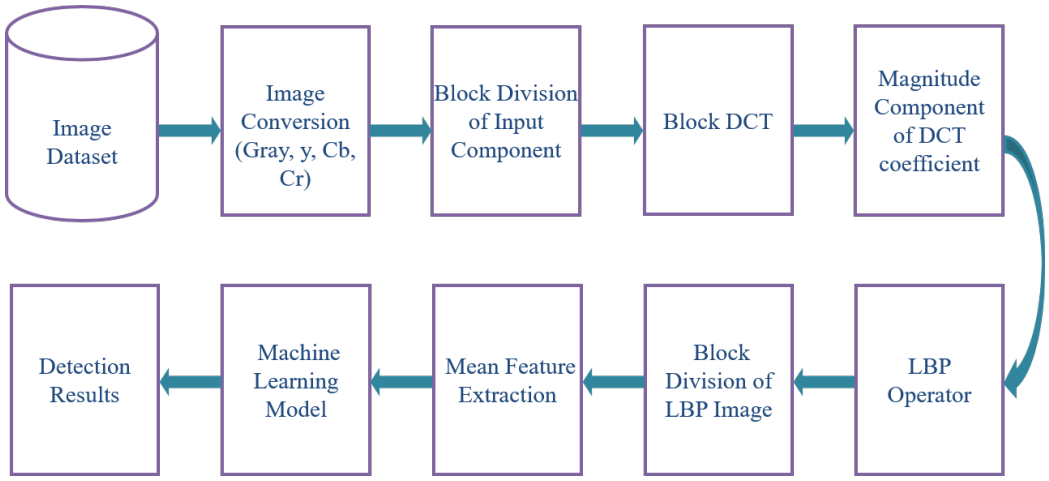


Figure 2. Proposed system for detecting image splicing and copy–move attacks.

As shown in Figure 3, we apply DCT before LBP (in short DCT-LBP). The reason behind selecting DCT-LBP is explained here. LBP considers only neighbouring pixels whose values are greater than or equal to the candidate pixel value and thus suppresses other pixels. As a consequence, LBP reduces the pixel intensity variations to a certain extent. If we use standard deviation-based features introduced in [10], applying LBP before DCT (in short LBP-DCT) outperforms DCT-LBP (refer to 88.50% vs. 79.50% detection accuracy in Figure 3). This can be explained by the fact that removing some pixel values by LBP and its magnification makes standard deviation-based features more discriminating than those for DCT-LBP. In contrast, since LBP removes some of the forgery artifacts that could be captured by DCT, using our mean-based features for FBDDF dataset (see details in Section 4.1.2), applying LBP-DCT in forgery detection leads to lower detection accuracy compared with utilizing DCT-LBP. For this reason, using our mean-based features, Figure 3 shows the image forgery detection accuracy of DCT-LBP (95.84%) is higher than that of LBP-DCT (92.18%). This is because subtle changes are captured by DCT and later some of them are enhanced by LBP. Consequently, this enhancement considerably impacts the mean-based features and make them more distinctive. This fact motivates us to apply DCT-LBP in our proposed method. Figure 3 also shows our mean-based features are more effective than the standard deviation-based features. Our proposed method is presented in the following section.

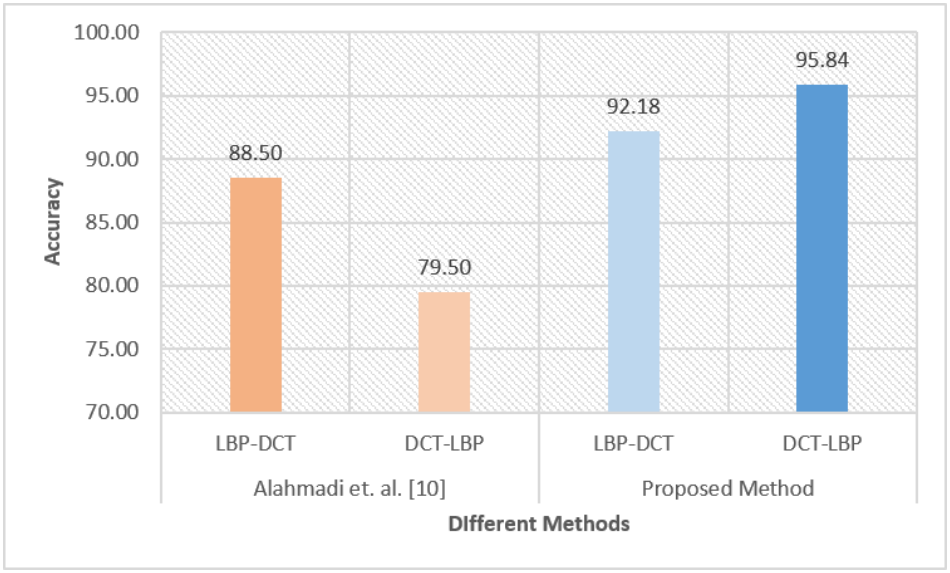


Figure 3. Comparison between application order of Local Binary Pattern (LBP) and Discrete Cosine Transformation (DCT).

3.1. Converting Color Images into Grayscale and YCbCr Color Space Images

Image forgery detection methods have been evolved separately for color and grayscale images. Unlike existing methods available in the current literature, our proposed image forgery detection approach is equally applicable to both color and grayscale images. However, for evaluating image forgery detection, so far, only one benchmark dataset is available for the grayscale images. On the other hand, there are a few datasets for color images. For this reason, for assessing the efficacy of our proposed approach for both color and grayscale images, we convert the color images into grayscale images and YCbCr color space images as required. In this paper, we specifically choose YCbCr color space because existing forgery detection techniques produce superior results for this over other color models.

3.2. Block Division of Grayscale Image and YCbCr Color Space Components

Dividing an image into some fixed size square blocks appears to be the most effective way adopted in image forgery detection methods. Because this allows us to propagate the impact of changes, even a small change made by copy–move and splicing operations, into all features, which makes features being more discriminating and thus effective [10]. The smaller the block, the higher the chance to detect the subtle changes and less the number of features are. Besides, this block division also enables us to identify a fixed number (size of a block) of features. Less number of features cannot represent all different changes in a large number of images. In contrast, a very high dimensional feature space suffers from the curse of dimensionality, losing the discriminating capability for a very large dimensional feature set [25]. For capturing changes effectively and a suitable feature dimension selection, in the first phase, we divide the grayscale images and YCbCr components into different blocks (4×4, 8×8, 16×16 and combining all these blocks). The second phase of block division is applied to the resultant image after applying DCT and LBP operations, which is discussed in Section 3.5. The block division of an image is performed using the following procedure.

Let $I^{wb \times hb}$ be an image of size $wb \times hb$ pixels. We divide $I^{wb \times hb}$ into $w \times h$ non-overlapping blocks of size $b \times b$ pixels. The resultant two dimensional block array is given by,

$$I^{wb \times hb} = \begin{bmatrix} I_{0,0}^{b \times b} & \cdots & I_{0,w-1}^{b \times b} \\ \vdots & \ddots & \vdots \\ I_{h-1,0}^{b \times b} & \cdots & I_{h-1,w-1}^{b \times b} \end{bmatrix} \quad (1)$$

3.3. Block Discrete Cosine Transformation (BDCT)

Splicing and copy–move operations introduce unexpected changes as well as micro patterns along the fine boundary of the tampered region. As a result, the local frequency distribution of the tampered area changes. The natural correlation among image pixels are also disrupted as these operations modify the smoothness, regularity and continuity of host image pixels, specially around the edges of the tampered region [12]. To aid forgery identification, the diversity of image contents needs to be reduced and the tampering artifacts enhanced, before deriving discriminating feature set. An image is converted from the pixel domain to the frequency domain using block DCT (BDCT) to capture the level of image content change. BDCT is well known for its excellent capability of energy compaction and image pixel decorrelation which, in turn, captures pixel domain changes in local frequency distribution [31].

We apply 2D-DCT on the blocks of $I^{wb\times hb}$ to generate DCT coefficients. Let $Y^{wb\times hb}$ be the resultant transform domain coefficient after applying 2D-DCT on each block and it is given by,

$$Y^{wb\times hb} = \begin{bmatrix} Y_{0,0}^{b\times b} & \cdots & Y_{0,w-1}^{b\times b} \\ \vdots & \ddots & \vdots \\ Y_{h-1,0}^{b\times b} & \cdots & Y_{h-1,w-1}^{b\times b} \end{bmatrix} \tag{2}$$

where, $Y_{i,j}^{b\times b} = 2D\text{-}DCT(I_{i,j}^{b\times b}), 0 \le i \le w-1, 0 \le j \le h-1$. The 2D-DCT of an input block $I_{i,j}^{b\times b}$ and output block $Y_{i,j}^{b\times b}$ is given by,

$$Y_{i,j}^{b\times b} = \alpha_p \alpha_q \sum_{m=0}^{b-1}\sum_{n=0}^{b-1} I_{i,j}^{b\times b}(m,n) \times \cos\frac{\pi(2m+1)p}{2b} \times \cos\frac{\pi(2n+1)q}{2b}, \tag{3}$$

where, $0 \le p \le b-1, 0 \le q \le b-1$ and

$$\alpha_p = \begin{cases} \sqrt{\frac{1}{b}}, & \text{if } p = 0 \\ \sqrt{\frac{2}{b}}, & \text{otherwise} \end{cases}, \tag{4}$$

$$\alpha_q = \begin{cases} \sqrt{\frac{1}{b}}, & \text{if } q = 0 \\ \sqrt{\frac{2}{b}}, & \text{otherwise} \end{cases}. \tag{5}$$

3.4. Local Binary Pattern (LBP) Operator

To enhance and identify forgery artifacts on images, we apply LBP, a computationally inexpensive and robust texture descriptor, on the magnitude components of the BDCT array derived in (2). Note, the consideration of DCT reduces the impact of translation, while magnitude of its coefficients makes it less sensitive to rotation. The main reason behind adopting LBP is to identify the occurrences of new micro-patterns arising from tampering. LBP highlights these forgery artifacts and augment them in the host image. LBP operation compares every DCT value with its eight neighbouring DCT values and generates LBP code for that value. The LBP codes are derived as below.

For the central DCT coefficient p_c, let $L_r(p_c)$ be an LBP operator applied to all DCT coefficients within a neighbourhood having radius r. $L_r(p_c)$ is defined as,

$$L_r(p_c) = \sum_{n=0}^{N-1} g(p_n - p_c)2^n. \tag{6}$$

where for a rectangular neighbourhood, $N = (r+1)^2$ for $r = 1$ and $N = (r+1)^2 - 1$ otherwise [32]. The neighbouring DCT coefficient is defined as p_n where $n = 0, 1, \cdots, N-1$. The function $g(p_n - p_c)$ is defined as,

$$g(p_n - p_c) = \begin{cases} 1, & p_n - p_c \geq 0 \\ 0, & p_n - p_c < 0 \end{cases}. \tag{7}$$

A rectangular window for calculating LBP has been adopted in our method where we choose the number of neighbours $N = 8$ with radius $r = 2$. If the central value p_c is greater than its neighbouring value, then 0 is recorded; otherwise 1. Thus, the central DCT value p_c receives an eight bit binary code which is converted into the corresponding decimal value, and this value is stored as the LBP code of the considered central DCT value. Figure 4 depicts the process with an example. The eight bit binary code is assembled from the Least Significant Bit (LSB) to the Most Significant Bit (MSB). The decimal value then replaces the central DCT coefficient p_c. The process continues for all the DCT values and respective LBP codes are being generated for the entire image.

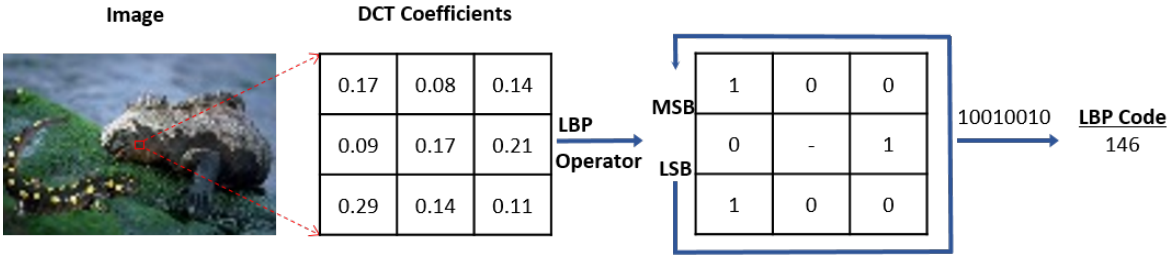


Figure 4. LBP code generation procedure.

Let $\mathscr{L}^{wb \times hb}$ be the resultant LBP array generated by applying LBP operator, $L_r()$ defined in (6), on magnitude components ($|Y^{wb \times hb}|$) of $Y^{wb \times hb}$. It is given by,

$$\mathscr{L}^{wb \times hb}(x, y) = L_r(|Y^{wb \times hb}(x, y)|), \tag{8}$$

where, $0 \leq x \leq wb - 1$ and $0 \leq y \leq hb - 1$.

3.5. Block Division of LBP Array

In the second phase of block division, we divide the LBP 2D array $\mathscr{L}^{wb \times hb}$ into same size of blocks, similar to the block division operation performed in Section 3.2. We divide $\mathscr{L}^{wb \times hb}$ into $w \times h$ non-overlapping blocks of size $b \times b$ LBP codes. The resultant block 2D array of LBP is represented as

$$\mathscr{L}^{wb \times hb} = \begin{bmatrix} \mathscr{L}^{b \times b}_{0,0} & \cdots & \mathscr{L}^{b \times b}_{0,w-1} \\ \vdots & \ddots & \vdots \\ \mathscr{L}^{b \times b}_{h-1,0} & \cdots & \mathscr{L}^{b \times b}_{h-1,w-1} \end{bmatrix} \tag{9}$$

3.6. Mean-Based Feature Extraction

The performance of a classifier is heavily dependent on the capability of a feature set to separate the classes. This fact motivates us to identify data points to approximate features that can apprehend an image forgery more accurately. For such identification, as mentioned before, we choose a similar

technique adopted by Alahmadi et al. [10] which calculates the standard deviation-based feature set. Since this technique propagates the change in a block to all features, it appears to be a promising approach for image forgery detection. However, instead of the standard deviation, we aim to use the mean-based features. The justification of selecting the mean-based features can be attributed to the fact that DCT transforms an image component from spatial domain to frequency domain to identify changes among different pixel values of that input. Because of alien micro patters introduced by forgery, higher and more non-zero values in high frequency DCT coefficients are observed in the tampered image based on the quality and quantity of image content modification. As justified before, to preserve these micro patterns, our method adopts first DCT and then LBP as DCT coefficients capture the unnatural changes in forged image blocks while LBP enhances those forgery artifacts. Since image tampering by splicing and copy move attacks introduces subtle local changes in host images, those changes can be identified as outliers. It is a well-known fact that mean is most affected by outlier more than any other statistical operator. Besides, for a fixed number of data points, the computational time for calculating mean is much less than that for standard deviation. This is why, we adopt mean operation for calculating our feature set. As mentioned before, a block contains b^2 number of cells that represent the number of features derived by our proposed method. For each cell, we calculate the mean of all LBP codes of all blocks containing in a 2D LBP array. The feature extraction process is detailed in the following equations.

The mean-based feature $F(x, y)$ for the cells indexed by xth row and yth column in all blocks is derived as,

$$F(x,y) = \frac{\sum_{u=0}^{w-1} \sum_{v=0}^{h-1} \mathscr{L}_{u,v}^{b\times b}(x,y)}{w \times h}. \tag{10}$$

where, $0 \leq x, y \leq b-1$ and $\mathscr{L}_{u,v}^{b\times b}()$ represents uth row and vth column block of the 2D LBP array ($\mathscr{L}^{wb\times hb}$). Note, as mentioned before, using (10), we calculate b^2 features over $w \times h$ blocks. Mean-based feature usage makes our method independent of image size.

For n data points, the computational complexity of both mean and sample standard deviation calculation is $O(n)$. Mean computation requires n additions and one division operations. Besides mean calculation, sample standard deviation requires extra n additions and squares, $n+1$ subtractions, and one square root and division operations. Therefore, extracting the mean-based feature is computationally faster than standard deviation-based feature calculation.

4. Experiments and Results

4.1. Description of Datasets

The following subsections discuss the datasets used to evaluate our proposed system:

4.1.1. Existing Datasets

We have evaluated our proposed splicing and copy–move attack detection system on four publicly available well recognized image forgery detection datasets: (i) Columbia Gray [33], (ii) Columbia Color [17], (iii) CASIA 1 and (iv) CASIA 2 [20]. We agree with [13,19] that among all of the publicly available datasets, CASIA 2 is the latest state-of-the-art splicing and copy–move attack dataset having a large number of color image samples in varying pixel dimensions which portray natural and realistic image forgery and thus, a robust forgery detection method should have higher detection accuracy on this dataset. Table 1 summarizes the key features of these datasets along with our newly created FBDDF dataset detailed below.

Table 1. Desription of datasets.

Dataset	Image Size	Image Type	No. of Images			Tampering
			Authentic	*Tampered*	*Total*	
Columbia Gray	128 × 128	BMP	933	912	1845	Simple crop-and-paste in small block of gray image , no post processing or color image
Columbia Color	757 × 568–1152 × 768	TIFF	183	180	363	Simple crop-and-paste using Photoshop, high resolution uncompressed images
CASIA 1	384 × 256, 256 × 384	JPEG	800	921	1721	Photoshop with pre-processing, no post-processing
CASIA 2	240 × 160–900 × 600	JPEG, TIFF, BMP	7491	5123	12,614	Photoshop with pre-processing and/or post-processing
FBDDF (New)	1280 × 720	JPEG, TIFF	200	200	400	Uncompressed Splicing and copy–move using Photoshop, suitable for self driving vision forensics

4.1.2. Our Developed Dataset—FBDDF

We have built a new splicing and copy–move detection dataset called FBDDF (Federation–Berkeley Deep Drive Forgery), using the BDD (Berkely Deep Drive) dataset [16]. BDD dataset is currently world's largest open and crowd sourced driving dataset of video and images (over 100 k videos and 100 k images extracted at 10th second of each video) covering different time of day, road conditions and weather conditions (e.g., day, night, sunny, foggy, rain, snow). This dataset is widely used for IoT based self-driving vehicular vision research. Advances in intelligent traffic system (ITS) and self-driving cars must implement strong security mechanism which should thwart image forgery attacks. Therefore, a splicing and copy–move dataset using BDD will fill-in the gap for a test dataset around IoT based ITS imagery. We have randomly selected 200 original images from BDD and performed deliberate splicing and copy–move attacks to generate 200 fake images using Photoshop CC 2018. All original images are in jpeg format whereas, the forged images are saved in uncompressed tiff format to reduce the effects of jpeg image compression issues in forgery detection. This is because if the spliced region on the host image is derived from a foreign image having different resolution, and image compression (e.g. jpeg compression) is performed to generate the forged image after splicing operation, then there could be a mismatch in image compression which would be easier to identify. Hence, to make FBDDF more challenging in image forgery detection, we have saved all forged images into uncompressed tiff format. FBDDF is publicly available for the research community at https://bit.ly/2ZrKe8Q. Figure 5 displays some samples from FBDDF and Table 1 summarizes the key features of the dataset.

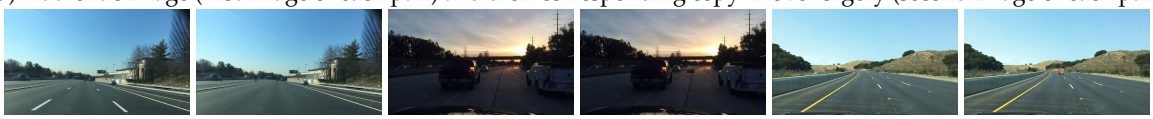
4.2. SVM Classifier and Model Validation

SVM, a machine learning algorithm, has been widely adopted in many well recognized spicing and copy–move attack detection methods as it shows encouraging performance in classifying authentic and forged image [9–11,13–15,17–24,26,28–30]. We selected LIBSVM [34] implementation as the classifier to evaluate overall accuracy and related performance metrics for our proposed architecture. We used radial basis function (RBF) as the SVM kernel. Since SVM's learning parameters influence classification performance, we utilized Bayesian Optimization [35] to find the best values for regularization parameter and the width of the RBF kernel. We used ten-fold cross-validation which is the most suitable and widely used method to divide a dataset into training and test sets [36]. In ten-fold cross validation, the whole dataset is divided into 10 nearly equal-sized mutually exclusive folds and the ratio of the real and forged images in each fold was kept roughly the same as in the total dataset. Each time nine folds were used for training and validation, and one fold was used as the test set. The process was repeated 10 times, each time with a different test set (test sets are disjoint). This constituted one trial and the average of ten fold's performance was recorded as the overall performance of the model in that trial. We conducted 20 such trials and in the following subsection

we report the average value of each of the four widely used performance metrics along with their standard deviations of these 20 trials. We identify the best block which yeilds best detection accuracy for a particular dataset and report in Section 4.4. We used MATLAB R2019a for data pre-processing, feature extraction and classification.

(**a**) Authentic image (first image of each pair) and their corresponding splicing forgery (second image of each pair)

(**b**) Authentic image (first image of each pair) and their corresponding copy–move forgery (second image of each pair)



(**c**) Authentic image (first image of each pair) and their corresponding splicing & copy–move forgery (second image of each pair)

Figure 5. Samples from the FBDDF (Federation–Berkeley Deep Drive Forgery) image dataset: (**a**) Splicing, (**b**) copy–move and (**c**) splicing and copy–move combined.

4.3. Performance Metrics

We evaluate our proposed method using the following performance measures:

4.3.1. Accuracy

Accuracy is defined as,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%, \tag{11}$$

where TP (true positive) is the number of tampered images that are correctly detected as tampered, TN (true negative) is the number of authentic images that are correctly classified as authentic, FP (False positive) is the number of authentic images that are incorrectly classified as tampered and FN (false negative) is the number of tampered images that are incorrectly identified as authentic image.

4.3.2. False Negative Rate (FNR)

Any forensic analytic system should be capable enough to produce low False Negative Rate (FNR) to make the system secure and trustworthy. FNR is given by,

$$FNR = \frac{FN}{TP + FN} \times 100\%. \tag{12}$$

4.3.3. Sensitivity

Sensitivity is the measure of actual portion of forged image being correctly identified as forged image. It is given by,

$$Sensitivity = \frac{TP}{TP + FN} \times 100\%. \tag{13}$$

4.3.4. Specificity

Specificity is the percentage of authentic image being correctly identified as authentic. It is given by,

$$Specificity = \frac{TN}{TN + FP} \times 100\%. \quad (14)$$

4.4. Results and Discussion

The detection accuracy, FNR, specificity and sensitivity along with their standard deviations (SD) for features extracted from gray scale image as well as different color channels of an image (Y, Cb, Cr, Cb+Cr) are reported in Tables 2 and 3. All of them were tested for block size of 4×4, 8×8, 16×16 as well as combining all blocks ($4 \times 4 + 8 \times 8 + 16 \times 16$). As different datasets consist of images with different resolution and different sized forgery operations, a fixed block size is not sufficient for every dataset. Hence, we identify the best block which yields best detection accuracy for a particular dataset and report them in this paper. Note, all the reported results are in percentage. For gray scale image, our forgery detection approach achieves detection accuracy of 85.56 ± 0.24, 96.10 ± 0.43, 98.61 ± 0.11, 99.29 ± 0.03 and 95.84 ± 0.37 over Columbia Gray, Columbia Color, CASIA 1, CASIA 2 and FBDDF datasets respectively. For color images, the detection accuracies for corresponding datasets are 98.20 ± 0.28, 99.55 ± 0.08, 99.88 ± 0.01 and 100.00 ± 0.00.

Table 2. Overall detection accuracy and False Negative Rate (FNR) in our proposed method with varying block size and different image channels.

Datasets	Block Size	Accuracy					FNR				
		Gray	Color Components				Gray	Color Components			
			Y	Cb	Cr	CbCr		Y	Cb	Cr	CbCr
Columbia Gray	4×4	78.82 ± 0.23	-	-	-	-	23.02 ± 0.46	-	-	-	-
	8×8	$\mathbf{85.56 \pm 0.24}$	-	-	-	-	14.47 ± 0.48	-	-	-	-
	16×16	81.44 ± 0.36	-	-	-	-	18.11 ± 0.70	-	-	-	-
	Combined	85.54 ± 0.32	-	-	-	-	$\mathbf{13.24 \pm 0.84}$	-	-	-	-
Columbia Color	4×4	92.51 ± 0.34	93.46 ± 0.49	89.66 ± 0.90	90.67 ± 0.57	91.45 ± 0.42	6.03 ± 0.73	5.81 ± 0.66	12.56 ± 2.58	12.28 ± 1.42	8.14 ± 0.63
	8×8	96.10 ± 0.43	96.20 ± 0.43	97.73 ± 0.32	96.58 ± 0.33	$\mathbf{98.20 \pm 0.28}$	4.00 ± 0.64	4.47 ± 0.82	3.560 ± 0.58	4.580 ± 0.59	$\mathbf{2.56 \pm 0.58}$
	16×16	95.22 ± 0.41	95.41 ± 0.26	96.17 ± 0.32	94.77 ± 0.46	95.33 ± 0.32	5.03 ± 0.64	4.86 ± 0.57	5.171 ± 0.04	7.831 ± 0.44	6.92 ± 1.45
	Combined	95.99 ± 0.45	96.85 ± 0.38	97.41 ± 0.29	96.23 ± 0.25	96.39 ± 0.27	4.39 ± 0.70	3.33 ± 0.54	3.670 ± 0.71	4.720 ± 0.58	5.17 ± 0.70
CASIA 1	4×4	87.33 ± 0.29	87.51 ± 0.34	92.86 ± 0.25	94.76 ± 0.23	95.28 ± 0.27	12.79 ± 0.36	12.45 ± 0.37	5.93 ± 0.37	4.33 ± 0.29	3.97 ± 0.37
	8×8	98.24 ± 0.11	98.07 ± 0.13	98.77 ± 0.09	99.03 ± 0.10	99.00 ± 0.05	2.00 ± 0.38	2.37 ± 0.31	1.18 ± 0.15	0.80 ± 0.15	0.93 ± 0.16
	16×16	96.14 ± 0.16	96.14 ± 0.18	99.11 ± 0.09	99.51 ± 0.05	99.44 ± 0.08	4.64 ± 0.40	4.60 ± 0.34	1.19 ± 0.14	0.48 ± 0.08	0.66 ± 0.12
	Combined	98.61 ± 0.11	98.57 ± 0.08	99.28 ± 0.12	$\mathbf{99.55 \pm 0.08}$	99.33 ± 0.06	1.56 ± 0.16	1.63 ± 0.12	0.59 ± 0.21	$\mathbf{0.43 \pm 0.09}$	0.66 ± 0.12
CASIA 2	4×4	93.86 ± 0.08	93.64 ± 0.08	98.72 ± 0.03	98.50 ± 0.04	98.81 ± 0.04	6.15 ± 0.25	6.38 ± 0.24	1.64 ± 0.06	1.84 ± 0.06	1.38 ± 0.07
	8×8	97.70 ± 0.05	98.33 ± 0.05	99.65 ± 0.02	99.60 ± 0.02	99.65 ± 0.02	2.40 ± 0.08	1.79 ± 0.10	0.40 ± 0.04	0.40 ± 0.05	0.33 ± 0.04
	16×16	99.06 ± 0.05	99.47 ± 0.03	99.85 ± 0.02	99.79 ± 0.01	$\mathbf{99.88 \pm 0.01}$	0.87 ± 0.07	0.55 ± 0.05	0.16 ± 0.03	0.24 ± 0.03	$\mathbf{0.16 \pm 0.02}$
	Combined	99.29 ± 0.03	99.36 ± 0.03	99.74 ± 0.03	99.71 ± 0.03	99.74 ± 0.02	0.71 ± 0.05	0.57 ± 0.06	0.27 ± 0.04	0.33 ± 0.04	0.27 ± 0.03
FBDDF	4×4	89.58 ± 0.58	89.38 ± 0.65	99.94 ± 0.11	$\mathbf{100.00 \pm 0.00}$	99.95 ± 0.10	10.83 ± 0.86	11.13 ± 0.89	0.00 ± 0.00	0.00 ± 0.00	$\mathbf{0.00 \pm 0.00}$
	8×8	95.84 ± 0.37	95.86 ± 0.34	99.75 ± 0.00	99.58 ± 0.12	99.74 ± 0.06	4.85 ± 0.59	4.25 ± 0.68	0.00 ± 0.00	0.08 ± 0.18	0.00 ± 0.00
	16×16	85.60 ± 0.70	84.91 ± 0.73	99.95 ± 0.10	99.98 ± 0.08	99.98 ± 0.08	14.03 ± 1.19	14.05 ± 1.39	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
	Combined	94.45 ± 0.42	94.63 ± 0.42	99.75 ± 0.00	99.75 ± 0.00	99.75 ± 0.00	5.65 ± 0.59	5.30 ± 0.73	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00

Table 3. Sensitivity and specificity in our proposed method with varying block size and different image channels.

Datasets	Block Size	Sensitivity					Specificity				
		Gray	Color Components				Gray	Color Components			
			Y	Cb	Cr	CbCr		Y	Cb	Cr	CbCr
Columbia Gray	4 × 4	76.98 ± 0.46	-	-	-	-	80.61 ± 0.56	-	-	-	-
	8 × 8	85.53 ± 0.48	-	-	-	-	**85.59 ± 0.43**	-	-	-	-
	16 × 16	81.89 ± 0.70	-	-	-	-	81.01 ± 0.72	-	-	-	-
	Combined	**86.76 ± 0.84**	-	-	-	-	84.35 ± 0.66	-	-	-	-
Columbia Color	4 × 4	93.97 ± 0.73	94.19 ± 0.66	87.44 ± 2.58	87.72 ± 1.42	91.86 ± 0.63	91.07 ± 0.60	92.73 ± 0.75	91.83 ± 1.93	93.58 ± 1.55	91.04 ± 0.99
	8 × 8	96.00 ± 0.64	95.53 ± 0.82	96.44 ± 0.58	95.42 ± 0.59	**97.44 ± 0.58**	96.20 ± 0.57	96.86 ± 0.46	**98.99 ± 0.44**	97.73 ± 0.67	98.93 ± 0.38
	16 × 16	94.97 ± 0.64	95.14 ± 0.57	94.83 ± 1.04	92.17 ± 1.44	93.08 ± 1.45	95.46 ± 0.59	95.68 ± 0.47	97.49 ± 0.84	97.32 ± 1.37	97.54 ± 1.43
	Combined	95.61 ± 0.70	96.67 ± 0.54	96.33 ± 0.71	95.28 ± 0.58	94.83 ± 0.70	96.37 ± 0.60	97.02 ± 0.57	98.47 ± 0.58	97.16 ± 0.46	97.92 ± 0.70
CASIA 1	4 × 4	87.22 ± 0.36	87.55 ± 0.37	94.07 ± 0.37	95.67 ± 0.29	96.03 ± 0.37	87.47 ± 0.45	87.47 ± 0.58	91.48 ± 0.41	93.72 ± 0.39	94.43 ± 0.45
	8 × 8	98.00 ± 0.38	97.63 ± 0.31	98.82 ± 0.15	99.20 ± 0.15	99.07 ± 0.16	98.51 ± 0.35	98.59 ± 0.25	98.71 ± 0.14	98.84 ± 0.23	98.92 ± 0.15
	16 × 16	95.36 ± 0.40	95.40 ± 0.34	98.81 ± 0.14	99.52 ± 0.08	99.34 ± 0.12	97.04 ± 0.48	97.00 ± 0.31	99.46 ± 0.14	99.50 ± 0.07	**99.54 ± 0.11**
	Combined	98.44 ± 0.16	98.37 ± 0.12	99.41 ± 0.21	**99.57 ± 0.09**	99.34 ± 0.12	98.80 ± 0.20	98.81 ± 0.19	99.12 ± 0.15	99.53 ± 0.12	99.33 ± 0.13
CASIA 2	4 × 4	93.85 ± 0.25	93.62 ± 0.24	98.36 ± 0.06	98.16 ± 0.06	98.62 ± 0.07	93.87 ± 0.15	93.66 ± 0.15	98.96 ± 0.04	98.73 ± 0.06	98.94 ± 0.05
	8 × 8	97.60 ± 0.08	98.21 ± 0.10	99.60 ± 0.04	99.60 ± 0.05	99.67 ± 0.04	97.77 ± 0.08	98.41 ± 0.06	99.69 ± 0.03	99.59 ± 0.03	99.63 ± 0.03
	16 × 16	99.13 ± 0.07	99.45 ± 0.05	99.84 ± 0.03	99.76 ± 0.03	**99.84 ± 0.02**	99.01 ± 0.08	99.48 ± 0.05	99.85 ± 0.02	99.80 ± 0.02	**99.90 ± 0.02**
	Combined	99.29 ± 0.05	99.43 ± 0.06	99.73 ± 0.04	99.67 ± 0.04	99.73 ± 0.03	99.29 ± 0.05	99.31 ± 0.05	99.75 ± 0.03	99.74 ± 0.04	99.74 ± 0.03
FBDDF	4 × 4	89.18 ± 0.86	88.88 ± 0.89	100.00 ± 0.00	100.00 ± 0.00	**100.00 ± 0.00**	89.98 ± 0.60	89.88 ± 0.97	99.88 ± 0.22	**100.00 ± 0.00**	99.90 ± 0.21
	8 × 8	95.15 ± 0.59	95.75 ± 0.68	100.00 ± 0.00	99.93 ± 0.18	100.00 ± 0.00	96.53 ± 0.75	95.98 ± 0.60	99.50 ± 0.00	99.23 ± 0.26	99.48 ± 0.11
	16 × 16	85.98 ± 1.19	85.95 ± 1.39	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	85.23 ± 0.92	83.88 ± 1.09	99.90 ± 0.21	99.95 ± 0.15	99.95 ± 0.15
	Combined	94.35 ± 0.59	94.70 ± 0.73	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	94.55 ± 0.67	94.55 ± 0.54	99.50 ± 0.00	99.50 ± 0.00	99.50 ± 0.00

In general, we found the best detection accuracy for Cr component and the combination of Cb and Cr (CbCr) components. In the same way, we found better FNR, sensitivity and specificity for color components. The variations of detection accuracy in different chroma channels are observed in different datasets because of the nature of images [10]. For example, images in Columbia Color dataset were mostly taken in indoor conditions with exactly four specific models of cameras, whereas CASIA 1 and CASIA 2 contain images that were taken in both indoor and outdoor environments using different sources (e.g., Corel image dataset, websites, own camera sources). The detection accuracy using Cb component, Cr component and their combination for each dataset varies between 1∼2% approximately which indicates that our proposed method is quite robust. Selecting either of the chroma channels can result in satisfactory outcome.

In any forgery detection technique, achieving a low FNR indicates a more secure system. FNR indicates how many positive samples (forged images) are identified erroneously as negative (authentic image). From the forensic and security viewpoint, missing a tampered image has more severe consequences and thus a reliable detection system must produce FNR as low as possible. If an original image is identified as forgery (i.e., false alarm), then further investigation can lead to a safer solution. In contrast, if a forged image is identified as original image, it may compromise the integrity of the entire system. Our proposed system produces FNR between 0∼2.56% for chroma channels when evaluated with color datasets. Similar trend is also observed for sensitivity and specificity. We found encouraging results for sensitivity (97.44 ± 0.58∼100.00 ± 0.00) and specificity (98.99 ± 0.44∼100.00 ± 0.00) in either of the chroma channels or their combinations when tested with color datasets.

4.4.1. Effects of Number of Positive Samples in Training Set

In machine learning, sufficient number of training samples helps to build reliable models. Moreover, classifiers like SVM, neural networks work well for balanced class distribution, i.e., when the number of training samples in the classes are balanced. However, recent trend in heterogeneous IoT network introduces applications where sufficient samples may not be available. For example, a self

driving car and its image training dataset may not have sufficient real life tampered image samples as these occurrences are not yet common. A robust forgery detection method should still perform better despite low number of forgery samples. We tested our method for low number of positive samples in the training set and compared its performance with method in [10]. Here, we used CASIA 1 to evaluate how our system performs when different amount of forgery samples were used to train the SVM classifier. To make the experiment simple, we converted the images of CASIA 1 into gray scale. We included all the authentic images (800) in the training sets and added a certain number of forgery images (out of 921 forgery images) in the training set, and test the classifier with all the remaining tampered images. Table 4 summarises the results. It is observed that our method performs consistently better for different number of positive samples in the training set than the method in [10]. The existing method performs good only when there are sufficient number of positive training samples (accuracy 78.99% with 30% forgery data) while they fail to perform when there are few samples (accuracy 61.17% with 3% forgery data). In contrast, our method consistently works better for both sufficient (30% forgery) and and low number of positive samples (3% forgery) with accuracy of 93.90% and 77.01%, respectively in the above cases. Experiemnts with other datasets with low number of forgery samples for training also revealed similar improvement by our method. This demonstrates the robustness of our method which is capable of attaining acceptable detection accuracy even when available forgery samples are very low.

Table 4. Low number of forgery in training set: Proposed method vs. method in [10] using CASIA 1 dataset for gray scale images.

Train Set	Block Size	CASIA 1			
		Proposed Method		Method in [10]	
		Accuracy	FNR	Accuracy	FNR
3% Forgery	4 × 4	19.83	80.17	00.50	99.50
	8 × 8	72.88	27.12	52.18	47.82
	16 × 16	77.01	22.99	61.17	38.83
5% Forgery	4 × 4	20.51	79.49	01.10	98.90
	8 × 8	85.54	14.46	53.20	46.80
	16 × 16	80.53	19.47	63.17	36.83
7% Forgery	4 × 4	46.35	53.65	03.94	96.06
	8 × 8	86.41	13.59	58.00	42.00
	16 × 16	79.92	20.08	65.63	34.37
9% Forgery	4 × 4	46.66	53.34	13.34	86.66
	8 × 8	89.12	10.88	62.17	37.83
	16 × 16	82.06	17.94	66.12	33.88
10% Forgery	4 × 4	54.35	45.65	05.49	94.51
	8 × 8	89.95	10.05	63.50	36.50
	16 × 16	82.43	17.57	62.30	37.70
20% Forgery	4 × 4	69.48	30.52	38.87	61.13
	8 × 8	91.79	08.21	70.55	29.45
	16 × 16	83.88	16.12	68.16	31.84
30% Forgery	4 × 4	69.60	30.40	48.19	51.81
	8 × 8	93.90	06.10	77.66	22.34
	16 × 16	85.14	14.86	78.99	21.01

4.4.2. Effects of IoT Image Rotation, Scaling and Compression

In different application domains such as IoT based smart vehicles, wildlife monitoring, and underwater monitoring, the captured image may be rotated or scaled depending on different external events and environments. For example, a strong wind, storm, or strong underwater current may rotate and change the location of the tiny IoT camera nodes which can change the orientation of capturing images (rotate) as well as their focal distances affecting the scale of images, respectively. While existing literature has reported the effect of rotation and scaling of the tampered region in a forged image, which is very common in image tampering operation and inherently applied in all publicly available datasets [10,23], to the best of our knowledge, no existing image forgery method has experimented the impact of rotation or scaling of a whole image. We evaluate the performance of our proposed method with different rotations and scalings of the images of all reported datasets. Again, there are many types of IoT networks such as LoRa, Sigfox and NBIoT having different transmission rate. For example, their respective transmission rates are 0.3–50 kbps, 100/600 bps and 106/158.5 kbps which show that Sigfox has the lowest and NBIoT has highest transmission rate. However, transmission power consumption (28 mA) of Lora is the lowest [37]. In general, remote IoT sensors (e.g., used in wildlife monitoring) are constrained by power, data transmission rate, and processing capabilities whereas, stationary IoT sensors (e.g., home surveillance system) connected with mainstream power lines have higher data transfer rate and processing power [37]. This leads to the options of storing an IoT acquired image in either compressed or uncompressed form depending on the application systems and requirements to be detailed later. Hence, compression of images captured by IoT devices should be carefully selected. To check the effect of image compression in forgery detection, we have added an extra set of forged images for our developed FBDDF using jpeg compression. The effects of rotation, scaling, and compression on images are discussed below:

Compressed and uncompressed image domains: In block based compression such as JPEG, deblocking filters are used to smooth the block edges to avoid blocking artefacts. This effectively increases difference of pre-distortion in the image with the newly introduced distortion by splicing. As a result, out technique performs better with compressed images. To verify this concept, we used FBDDF (JPEG) and FBDDF (TIFF) datasets with the same images having different compressions. Table 5 shows forgery detection accuracy is higher for FBDDF (compressed—JPEG) than FBDDF (uncompressed—TIFF) across all block sizes (94.25% vs. 89.58%, 99.44% vs. 95.84%, 99.63% vs. 85.60%, 99.58% vs. 94.45% for block sizes 4×4, 8×8, 16×16 and combined blocks respectively). Again, the average accuracy of image tampering on different datasets for compressed images is also higher (96.65% vs. 89.72%) than that of uncompressed image-based datasets when there are no rotation or scaling attacks. In the next section, we discuss the effect of rotation and scaling on tampered images, which introduces new artifacts on tampered images. For this effect, our proposed method exhibits different performances for compressed and uncompressed image domains.

Rotation and compression: We conducted an experiment on images of different datasets used in this article. In Industrial IoT based monitoring system, image sensors may be deployed to monitor mechanical variables such as rotation, the number of cycles, position, the direction of travel for different components of machinery (e.g., motors, actuators) which result in rotated images of the same object of interest [38]. Again, natural and external environmental incidents may also lead to a rotated image due to image sensor disorientation. Even if the orientation of the deployed image sensor may change due to natural, accidental or operational incidents, for IoT applications, a good forgery detection system should still be able to identify forgery on these types of image data. Hence, for a given image dataset, we randomly rotated the images with the angles of 0 (no rotation), 90, 180, and 270 degrees and conducted an experiment with our proposed forgery detection system. While our proposed method is not perfectly rotation invariant, Table 5 shows that it is robust enough (for rotated images) to identify forgery with the average detection accuracy being 95.50%, 91.39%, 98.84% and differences of detection accuracy between original and rotated image sets having 1.63%, 3.27% and 1.37% for compressed, uncompressed and mixed image compression based datasets, respectively. For our

proposed method, these results vindicate that any natural, system inherent, external, or environmental events that may change the orientation of IoT image sensors or rotation of an object of interest occur during image capture will have very little effect in detecting image forgery even with different ratio of image compression.

Table 5. Effects of rotation, scaling and compression on images.

Dataset		Block	Proposed Method	Rotation	Scaling
			Accuracy	Accuracy	Accuracy
Compressed	CASIA 1 (JPEG)	4 × 4	87.33	87.94	77.68
		8 × 8	98.24	97.55	79.88
		16 × 16	96.14	97.48	84.85
		Combined	98.61	97.59	84.02
	FBDDF (JPEG)	4 × 4	94.25	86.79	65.48
		8 × 8	99.44	97.91	71.75
		16 × 16	99.63	99.31	82.35
		Combined	99.58	99.45	80.73
Average (compressed image datasets)			96.65	95.50	78.34
Uncompressed	FBDDF (TIFF)	4 × 4	89.58	96.74	98.64
		8 × 8	95.84	99.45	99.50
		16 × 16	85.60	98.95	99.50
		Combined	94.45	99.50	99.51
	Columbia Gray (BMP)	4 × 4	78.82	77.44	75.25
		8 × 8	85.56	84.66	79.23
		16 × 16	81.44	80.76	85.30
		Combined	85.54	85.18	85.46
	Columbia Color (TIFF)	4 × 4	92.51	90.14	95.10
		8 × 8	96.10	93.65	96.98
		16 × 16	95.22	95.70	97.15
		Combined	95.99	94.52	97.38
Average (uncompressed image datasets)			89.72	91.39	92.42
Mixed	CASIA 2 (JPEG, TIFF, BMP)	4 × 4	93.86	96.78	96.68
		8 × 8	97.70	99.26	99.69
		16 × 16	99.06	99.72	99.91
		Combined	99.29	99.61	99.89
Average (mixed compression image datasets)			97.48	98.84	99.04

Scaling and compression: As alluded before, because of image sensor movement or disorientation, the focal distance of a sensor can change which results in image scaling. Moreover, a malicious user may scale an image to achieve the intended forgery. To simulate image scaling operation, we used the imresize function of Matlab [39] with the default interpolation method (i.e., bicubic interpolation). This scaling operation introduces rounding error which modifies the sharp edges caused by image forgery [40]. To test the robustness of scaling operation in images, any of the following operations were randomly applied on individual images within a dataset: (a) Scale at 5%, 10% or 15% larger than the original image, (b) scale at 5%, 10% or 15% smaller than the original image, or (c) no change

in the original image. Therefore, the scaled dataset has a good mixture of upscaled, downscaled, and unchanged images. Table 5 shows that on average, the detection accuracy of scaled datasets drops to 18.31% than the original non-scaled datasets where the images in the datasets are in compressed (jpeg) form. However, if the images are in uncompressed form (tiff or bmp) or having a mix of compressed and uncompressed images (jpeg, tiff, bmp) in the dataset, our forgery detection method can identify image tampering having an average detection accuracy of 92.42% and 99.04% respectively for scaled datasets which is almost similar to the non-scaled original version of the datasets. Based on these experimental results, we suggest using the uncompressed form of images in mission-critical IoT based applications where the images are likely to suffer image scaling and tampering attacks.

4.4.3. Comparison with Recent Methods

There exist different methods for detecting splicing and copy–move attacks in current literature as described in Section 2. Among them, a recent work by Alahmadi et al. [10] adopted both LBP and DCT in their system and reported good detection accuracy. Though they used both LBP and DCT, but the application order and feature extraction method are different from ours. We implemented this method to compare thoroughly with our proposed system. Table 6 compares our proposed method with Alahmadi et al.'s method using newly proposed FBDDF dataset. For simplicity, we performed the test by converting the images into gray scale for FBDDF. It is observed that our method outperforms Alahmadi et al.'s method for all the performance metrics (accuracy 95.84% vs. 88.50%, FNR 4.85% vs. 11.50%, sensitivity 95.15% vs. 88.50% and specificity 96.53% vs. 88.50%) irrespective of block size. In addition to [10], we compared our method with other notable works on splicing and copy–move attacks and the results are presented in Table 7. Results show that our method increases the accuracy across all datasets by a margin of up to 13.20%.

Table 6. Comparison between proposed method and method in [10] using FBDDF dataset for gray scale images.

Evaluation	Proposed Method	Alahmadi's Method [10]
Accuracy	95.84	88.50
FNR	4.85	11.50
Sensitivity	95.15	88.50
Specificity	96.53	88.50

Table 7. Comparison of forgery detection accuracy of proposed method with existing methods.

Methods	Columbia Gray	Columbia Color	CASIA 1	CASIA 2
Proposed	**85.56**	**98.20**	**99.55**	**99.88**
Alahmadi et al. [10]	-	97.77	97.00	97.50
Muhammad et al. [30]	-	96.39	94.89	97.33
Hakimi et al. [15]	-	95.13	97.21	-
Hussain et al. [24]	-	94.29	-	-
Hsu and Chang [17]	-	87.00	-	-
Zhao et al. [22]	-	85.00	94.70	-
Wang et al. [19]	-	-	-	95.60
He et al. [13]	-	-	-	89.76
He et al. [9]	80.58	-	-	-
Dong et al. [21]	76.52	-	-	-
Kanwal et al. [29]	-	-	88.62	-

4.4.4. Comparison of CPU Processing Time

We recorded the CPU processing time to extract features between our method and Alahmadi's [10] method using the same hardware and software configurations. Note, as alluded before, data points used to calculate the mean and standard deviation-based features are directly proportional to the number of blocks. As discussed in Section 3.6, since the calculation of standard deviation involves more computational processes (operations), the higher the number of blocks, the higher the difference between the CPU processing time of mean and standard deviation is. Table 8 summarizes the CPU processing time of our proposed method and Alahmadi's method [10]. As expected, Table 8 shows our method is 14.54∼62.29% faster in feature extraction than Alahmadi's method over different block sizes and datasets. Our technique was able to extract features in as low as 0.004 seconds (which is equivalent to 4 ms) for a single image. This means that the technique can process 250 frames per second which is much higher than real-time video processing (e.g., 25 frames per second for PAL televisions). This vindicates our method is capable of handling real-time forgery detection in images. Additionally, if there is any edge device in an IoT environment, using GPU and parallel processing would increase the processing time.

Table 8. Comparison of mean CPU time (in second) for a single image.

Dataset (Gray)	Block Size	Mean CPU Time (in second) for a Single Image		
		Alahmadi's Method	Proposed Method	Percentage Decrease
FBDDF	4 × 4	1.157	0.862	34.23
	8 × 8	0.385	0.309	24.49
	16 × 16	0.183	0.159	15.25
Columbia Gray	4 × 4	0.027	0.017	62.29
	8 × 8	0.009	0.007	27.55
	16 × 16	0.005	0.004	21.48
Columbia Color	4 × 4	0.870	0.651	33.57
	8 × 8	0.288	0.229	25.72
	16 × 16	0.137	0.119	14.54
CAISA 1	4 × 4	0.130	0.092	40.82
	8 × 8	0.043	0.034	28.94
	16 × 16	0.022	0.018	19.41
CASIA 2	4 × 4	0.179	0.138	29.99
	8 × 8	0.062	0.048	28.33
	16 × 16	0.031	0.025	22.47

5. Conclusions

In this paper, we introduced an efficient and robust model for detecting splicing and copy–move attacks in both grayscale and color images using traditional machine learning technique (SVM) and hand-crafted features. First, we applied block DCT on image components to capture the changes in frequency domain due to tampering operation and then LBP of the magnitude component of resultant DCT coefficients are computed to enhance those tampering artifacts in pixel domain and also to identify the occurrences of micro pattern footprints left by image forgery. The LBP image was again divided into non-overlapping blocks and the mean of inter-cell LBP values are computed to extract unique features. SVM with RBF kernel was used for classification and the kernel parameters were tuned using Bayesian optimization. Experimental results in terms of detection accuracy and CPU time on different publicly available datasets and our newly developed IoT based forgery dataset confirm the superiority and robustness of our proposed method over existing methods found in current literature. Results also show that our proposed method is more effective and consistent in detecting splicing

and copy–move image forgery attacks even for a very low amount (3%) of forgery images in the training set. Furthermore, our method achieved encouraging performance when images were being rotated but struggled to cope with image scaling when the target images were in compressed (jpeg) format. However, when the images were in uncompressed format, the forgery detection accuracy of our method was similar to non-scaled images. Hence, we suggest to capture and store uncompressed images for any system where target images are more sensitive to scaling and tampering attacks. In the future, we will explore the performance of deep learning-based techniques for forgery detection. Since barrel distortion introduced by fisheye wide-angle lenses is one of the common issues of IoT vision, the forgery detection accuracy of our method may slightly vary for such distortion. We will also investigate the impact of barrel distortion on forgery detection in our further research.

Author Contributions: M.M.I. contributed to theoretical formulation, design methodology, dataset development, experiment design and implementation, result interpretation, original draft preparation and revision. The rest of the authors (G.K., J.K., and M.M.) contributed to supervising the project, theoretical formulation, result interpretation, and revising the initial draft. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Research Priority Area (RPA) scholarship of Federation University Australia.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this article:

IoT	Internet of Things
DCT	Discrete Cosine Transformation
LBP	Local Binary Pattern
SVM	Support Vector Machine
FBDDF	Federation–Berkeley Deep Drive Forgery
DWT	Discrete Wavelet Transformation
RLRN	Run-Length Run Number
WLD	Weber Local Descriptor
LTP	Local Ternary Pattern
ELTP	Enhanced Local Ternary Pattern
FFT	Fast Fourier Transform
SPT	Steerable Pyramid Transform
PCA	Principle Component Analysis
IIoT	Industrial Internet of Things
BDCT	Block Discrete Cosine Transformation
LSB	Least Significant Bit
MSB	Most Significant Bit
BDD	Berkely Deep Drive
ITS	Intelligent Traffic System
JPEG	Joint Photographic Experts Group
RBF	Radial Basis Function
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
FNR	False Negative Rate
SD	Standard Deviations
TIFF	Tagged Image File Format

References

1. Jonsson, P.; Carson, S. *Ericsson Mobility Report*; Ericsson: Stockholm, Sweden, 2018.
2. Redi, J.A.; Taktak, W.; Dugelay, J.L. Digital image forensics: A booklet for beginners. *Multimed. Tools Appl.* **2011**, *51*, 133–162. [CrossRef]

3. Mallonee, L. Infamously Altered Photos, Before and After Their Edits. Wired. 2015. Available online: https://bit.ly/2Ia8zqf (accessed on 15 March 2020).
4. Wikipedia contributors. List of Photo Manipulation Controversies. Wikipedia. 2019. Available online: https://bit.ly/2wcweBB (accessed on 15 March 2020).
5. Allbeson, T.; Allan, S. The War of Images in the Age of Trump. In *Trump's Media War*; Happer, C., Hoskins, A., Merrin, W., Eds.; Springer International Publishing: Berlin/Heidelberg, Germany, 2019; pp. 69–84.
6. Schreiber, M. *APTOPIX TRUMP GERMANY G20*; Associated Press: New York, NY, USA, 2017.
7. Novak, M. That Viral Photo of Putin and Trump Is Totally Fake. GIZMODO. 2017. Available online: https://bit.ly/3bL6PQo (accessed on 7 July 2020).
8. Johnson, M.K.; Farid, H. Exposing digital forgeries in complex lighting environments. *IEEE Trans. Inf. Forensics Secur.* **2007**, *2*, 450–461. [CrossRef]
9. He, Z.; Sun, W.; Lu, W.; Lu, H. Digital image splicing detection based on approximate run length. *Pattern Recognit. Lett.* **2011**, *32*, 1591–1597. [CrossRef]
10. Alahmadi, A.; Hussain, M.; Aboalsamh, H.; Muhammad, G.; Bebis, G.; Mathkour, H. Passive detection of image forgery using DCT and local binary pattern. *Signal Image Video Process.* **2017**, *11*, 81–88. [CrossRef]
11. Islam, M.M.; Kamruzzaman, J.; Karmakar, G.; Murshed, M.; Kahandawa, G. Passive Detection of Splicing and Copy–Move Attacks in Image Forgery. In Proceedings of the International Conference on Neural Information Processing, Siem Reap, Cambodia, 13–16 December 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 555–567.
12. Shi, Y.Q.; Chen, C.; Chen, W. A natural image model approach to splicing detection. In Proceedings of the 9th workshop on Multimedia & Security, Dallas, TX, USA, 20–21 September 2007; ACM: New York, NY, USA, 2007; pp. 51–62.
13. He, Z.; Lu, W.; Sun, W.; Huang, J. Digital image splicing detection based on Markov features in DCT and DWT domain. *Pattern Recognit.* **2012**, *45*, 4292–4299. [CrossRef]
14. Islam, M.M.; Karmakar, G.; Kamruzzaman, J.; Murshed, M.; Kahandawa, G.; Parvin, N. Detecting Splicing and Copy–Move Attacks in Color Images. In Proceedings of the 2018 Digital Image Computing: Techniques and Applications (DICTA), Canberra, Australia, 10–13 December 2018; pp. 1–7.
15. Hakimi, F.; Hariri, M.; GharehBaghi, F. Image splicing forgery detection using local binary pattern and discrete wavelet transform. In Proceedings of the 2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, Iran, 5–6 November 2015; pp. 1074–1077.
16. Yu, F.; Xian, W.; Chen, Y.; Liu, F.; Liao, M.; Madhavan, V.; Darrell, T. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv* **2018**, arXiv:1805.04687.
17. Hsu, Y.F.; Chang, S.F. Detecting image splicing using geometry invariants and camera characteristics consistency. In Proceedings of the 2006 IEEE International Conference on Multimedia and Expo, Toronto, ON, Canada, 9–12 July 2006; pp. 549–552.
18. Hsu, Y.-F.; Chang, S.-F. Image splicing detection using camera response function consistency and automatic segmentation. In Proceedings of the 2007 IEEE International Conference on Multimedia and Expo, Beijing, China, 2–5 July 2007; pp. 28–31.
19. Wang, W.; Dong, J.; Tan, T. Image tampering detection based on stationary distribution of Markov chain. In Proceedings of the 2010 IEEE International Conference on Image Processing, Hong Kong, China, 25 January 2010; pp. 2101–2104.
20. Dong, J.; Wang, W.; Tan, T. CASIA image tampering detection evaluation database. In Proceedings of the 2013 IEEE China Summit and International Conference on Signal and Information Processing, Beijing, China, 6–10 July 2013; pp. 422–426.
21. Dong, J.; Wang, W.; Tan, T.; Shi, Y.Q. Run-length and edge statistics based approach for image splicing detection. In Proceedings of the International Workshop on Digital Watermarking, Jeju Island, Korea, 8–10 November 2008; Springer: Berlin/Heidelberg, Germany; pp. 76–87.
22. Zhao, X.; Li, J.; Li, S.; Wang, S. Detecting digital image splicing in chroma spaces. In Proceedings of the International Workshop on Digital Watermarking, Atlantic City, NJ, USA, 23–26 October 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 12–22.
23. Hussain, M.; Muhammad, G.; Saleh, S.Q.; Mirza, A.M.; Bebis, G. Image forgery detection using multi-resolution Weber local descriptors. In Proceedings of the Eurocon 2013, Zagreb, Croatia, 1–4 July 2013; pp. 1570–1577.

24. Hussain, M.; Saleh, S.Q.; Aboalsamh, H.; Muhammad, G.; Bebis, G. Comparison between WLD and LBP descriptors for non-intrusive image forgery detection. In Proceedings of the 2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA), Alberobello, Italy, 23–25 June 2014.
25. Friedman, J.H. On bias, variance, 0/1—Loss, and the curse-of-dimensionality. *Data Min. Knowl. Discov.* **1997**, *1*, 55–77. [CrossRef]
26. Alahmadi, A.A.; Hussain, M.; Aboalsamh, H.; Muhammad, G.; Bebis, G. Splicing image forgery detection based on DCT and Local Binary Pattern. In Proceedings of the 2013 IEEE Global Conference on Signal and Information Processing, Austin, TX, USA, 3–5 December 2013; pp. 253–256.
27. Johnson, M.K.; Farid, H. Exposing digital forgeries through chromatic aberration. In Proceedings of the 8th Workshop on Multimedia and Security, Geneva, Switzerland, 26–27 September 2006; ACM: New York, NY, USA, 2006; pp. 48–55.
28. Wang, W.; Dong, J.; Tan, T. Effective image splicing detection based on image chroma. In Proceedings of the 2009 16th IEEE International Conference on Image Processing (ICIP), Cairo, Egypt, 7–10 November 2009; pp. 1257–1260.
29. Kanwal, N.; Girdhar, A.; Kaur, L.; Bhullar, J.S. Detection of Digital Image Forgery using Fast Fourier Transform and Local Features. In Proceedings of the 2019 International Conference on Automation, Computational and Technology Management (ICACTM), London, UK, 24–26 April 2019; pp. 262–267.
30. Muhammad, G.; Al-Hammadi, M.H.; Hussain, M.; Bebis, G. Image forgery detection using steerable pyramid transform and local binary pattern. *Mach. Vis. Appl.* **2014**, *25*, 985–995. [CrossRef]
31. Khayam, S.A. The discrete cosine transform (DCT): Theory and application. *Mich. State Univ.* **2003**, *114*, 1–31.
32. Karmakar, G.C. An Integrated Fuzzy Rule-Based Image Segmentation Framework. Ph.D. Thesis, Monash University, Melbourne, Australia, 2002.
33. Ng, T.T.; Chang, S.F. A model for image splicing. In Proceedings of the 2004 International Conference on Image Processing, ICIP'04, Singapore, 24–27 October 2004; Volume 2, pp. 1169–1172.
34. Chang, C.C.; Lin, C.J. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol. (TIST)* **2011**, *2*, 27. [CrossRef]
35. Martinez-Cantin, R. Bayesopt: A bayesian optimization library for nonlinear optimization, experimental design and bandits. *J. Mach. Learn. Res.* **2014**, *15*, 3735–3739.
36. Kohavi, R. *A Study of Cross-Validation and Bootstrap for Accuracy Estimation And Model Selection*; Ijcai: Montreal, QC, Canada, 1995; Volume 14, pp. 1137–1145.
37. Buurman, B.; Kamruzzaman, J.; Karmakar, G.; Islam, S. Low-Power Wide-Area Networks: Design Goals, Architecture, Suitability to Use Cases and Research Challenges. *IEEE Access* **2020**, *8*, 17179–17220. [CrossRef]
38. RYS, R. Smart Sensors Increasingly Important in the Industrial IoT Age. ARC Advisory Group. 2018. Available online: https://bit.ly/2W0et3f (accessed on 22 June 2020).
39. MATLAB. *Version 9.6.0.1072779 (R2019a)*; The MathWorks Inc.: Natick, MA, USA, 2019.
40. Wikipedia contributors. Image Scaling. Wikipedia. 2020. Available online: https://bit.ly/3fivvkO (accessed on 19 August 2020).