APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 6 No. 1 (2025) 560-575
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

Image Splicing Detection using Support Vector Machine (SVM)

Wan Yasmin Badrina Wan Omar Fathil[1], Nordiana Rahim[1]*

[1] *Fakulti Sains Komputer dan Teknologi Maklumat,*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: *nordiana@uthm.edu.my*
DOI: https://doi.org/10.30880/aitcs.2025.06.01.031

Article Info

Received: 9 May 2025
Accepted: 19 June 2025
Available online: 30 June 2025

Keywords

Discrete Wavelet Transform, Error Level Analysis, Histogram of Oriented Gradients, Local Binary Pattern, Machine Learning, Support Vector Machine.

Abstract

The official establishment's fake images pose a real problem across the internet and mostly involve social media to represent cybercrime and misinformation. At present, a lot of reactions are based on the images. The objective of this research is to evaluate the accuracy of image splicing detection based on the true positive and false positive rates by implementing techniques using machine learning and leveraging the features such as HOG, ELA, DWT and LBP and classified with SVM. This research used images as their datasets that consists of CASIA v1.0 and CASIA v2.0 and it highlights the potential of integrating the feature extraction techniques for robust, accuracy and reliability. However, this research found that with combination feature extraction ELA and HOG have achieved the highest accuracy with 94.00% for CASIA v1.0 followed by CASIA v2.0 95.50%, it shown that they performed well balancing accuracy. Overall, these findings demonstrate the ELA and HOG combination feature extraction had reliable outcome for image splicing detection

1. Introduction

Nowadays, people experience various images that have been manipulated by image applications [1]. The authenticity of the image is one of the most concerning because it can be used at the court as evidence, while if the image fails the image was never being approved by court law [1]. Knowing the identified sources of the image exactly is the basic issue that Digital Forensic can solve to find how it was made such as using what application, camera, or other digital application to know the integrity of the image that has been modified [1].

In general, image forgery conceals digital images to hide or alter information within images to trick or mislead editing software is frequently used for this purpose. Meanwhile, can be detected using a variety of methods including copy-move and splicing, detecting image forgery is crucial for ensuring the authenticity and reliability of digital images, especially in contexts like social media and legal proceedings [2]. To detect image forgery, several approaches were employed to identify image forgery such as format-based, pixel-based, camera-based, and geometry-based detection. These techniques look at spectral information, format, camera-specific acquisition noise, and geometric modifications to determine whether an image has been altered or changed [3].

The study conducted in examining paper about the use of Support Vector Machines (SVM) for the analysis and detection of picture forgeries is the main topic of this chapter. SVM-based approaches, examining how they're implemented and how effective they are. The analysis will look at three main aspects: technical evaluation (feature extraction, classification algorithms, accuracy), user considerations (ease of use, result interpretation), and the wider impact on the digital ecosystem (deterring manipulation, fostering trust, potential challenges). By comparing different features extraction with SVM classification, it aims to identify their

advantages and disadvantages and ultimately suggest improvements to their design and functionality for better real-world applications.

The problem statement discussed in the paper is the detection of fake images shared across the internet, with a special focus on the disclosure of image-based cybercrimes. Three objectives of the study have been discussed:

- To implement the techniques of forgery detection on existing techniques.
- To detect image splicing detection using SVM based on true positive rate and false positive rate for better accuracy.
- To evaluate the performance of forgery detection based on the accuracy, reliability of the dataset.

2. Literature Review

A manipulated image often has different statistical characteristics from that of the original image due to the insertion, removal, or modification of certain objects in the image. Characteristics that can be observed by the human eye can also be found using image processing algorithms. This would suggest a higher level of automation to analyze and detect manipulation in digital images [4]. The current technological advancement has simplified the manipulation of digital content, making it easier to blend and generate realistic fabrications as opposed to the original image.

This literature review looks to provide some suggestions to help to prove whether they have been altered or not. If an image is a result of manipulation, there is a good chance that it will have been through numerous savings and therefore have lost some of its original quality. This can make it hard to distinguish between an altered image and an original image, so a quality assessment tool would compare the characteristics of a test image to a database of characteristics of original images to estimate whether the test image is an altered image or an original [5].

This will be the key theme to compare when reviewing methods that aim to determine the integrity of an image. Although the area of picture forensics has been around for a while, its significance has grown along with the rise in the use of digital images. The two primary areas of image forensics are determining the imaging process or source and determining if a picture has been altered from its source. Any picture that is presented as evidence in court should ideally have some kind of identification on it to prove that it hasn't been altered after the occurrence it depicts.

2.1 Image Forgery Detection Using SVM classification

The introduction mentions the problem of image forgery detection followed by a brief literature review. As the name indicates, SVMs classify input features based on hyper-planes, which are optimal decision boundaries that separate two classes, while maximizing the margin, which represents the distance from the hyper-plane to the closest examples (called support vectors). This method yields a larger margin and hence better separates different data classes [6].

For classification, SVMs will select the extremal points (vectors) that can help create this separation plane. These are called the support vectors of the decision boundary. SVMs can potentially be applied to many different tasks, such as the detection of faces, the identification of images in a database, or the categorization of text [7].
The SVM algorithm captures what is distinctive about the images using very high-dimensional data, and the algorithm's capacity to sift through the signal from the noise accounts for its ability to achieve a clean separation between images that can appear to be borderline cases.

2.2 Type of Forgeries

Forgery detection in digital images refers to the manipulation or alteration of an image to deceive viewers by presenting a false representation of reality. Furthermore, it can involve types of techniques that modify, enhance, or combine images to create a misleading impression. Common types of image forgeries include copy-move forgery, where a part of the image is duplicated and pasted elsewhere within the same image as image splicing, which contributes to combining elements from multiple images to form a composite image and image retouching, which enhances or diminishes certain features for aesthetic purposes. Moreover, the general common image forgery may be referred to as any alteration that had been made and it would include also the resizing cropping or made an adjustment brightness and contrast of the image, it would made falsify information.

2.2.1 Copy Move

Copy-Move Forgery Detection is a part of a critical aspect of image forensics, where it focuses on how to identify the piece image that has been copied and pasted from another source to alter the original content. In addition, the technique was often generated to misrepresent information or create false images while it is most

typically popular for blind image forensics [8]. For example, Fig 1 shows the difference between original and forged image.

(a) **(b)**

Fig. 1 *showed that the left image(a) is the original image; (b) is a copy-move forgery cat containing a large, moderately textured copy region [8]*

Fig. 1 shows the detection of such forgery requires sophisticated algorithms to be capable of recognizing manipulated segments in the image, which may have undergone slight modifications such as retouching or localized image processing has been show in Fig.1.For example, a copy-move forgery could have the detection process several steps that could be through such as feature extraction, matching features, localization, and refining and detection results[9]. To effectively address this challenge, researchers have developed methods that leverage superpixel segmentation and the Helmert transformation. Moreover, superpixel segmentation divides an image into segments, or superpixels, that share similar characteristics, making it easier to identify and compare these segments across different images.

2.2.2 Image Retouching

Image retouching is a forgery detection in which the image is altered by enhancing the visual content of an image by using any editing software. Mostly it performed that by all the out their photographer uses this retouching to improve the quality of images by using their beautification filter to be more attractive [10]. For example, Fig. 2 shows the difference between original and image that has been retouch.

(a) **(b)**

Fig. 2 *showed Image Retouching on the left is the Original Image, and on the right is the Tampered Image [10]*

In Fig. 2, method is not considered a forgery because it involves tampering with the original image such as erasing freckles and wrinkles to make it more attractive. On the other hand, the forgery can be carried out using ink, paint, lightroom, photo beauty, or double exposure to make high-quality resolutions [11].

2.2.3 Image Splicing or Photomontage

Image splicing is a type of tampering that makes composite fake images that can be combined by the different regions of the source, for this splicing the image cannot be possible to detected easily by the naked eye, it needed to use a technique to identify the image whether authenticity or not [12].For example, Fig. 3 has been show the image spliced and authentic within the same image.

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 560-575

(a) **(b)**

Fig. 3 *showed Image Retouching on (a) is the Original Image, and on (b) is the Tampered Image [12]*

Fig. 3 shows that image splicing also can be defined as simply cutting and pasting the image region from one image to another image. The main use of image splicing is to make artistic creation, advertising, filmmaking, and news reports where it is not easily noticeable by any people without identifying and locating the actual image that is caused by splicing.

2.3 Image Splicing Detection Scheme

Image splicing detection schemes are designed to identify instances where parts of one or more images have been cut and pasted into another image to create a composite that may mislead viewers. Additionally, detecting image splicing is a critical task in digital forensics, media verification, and other fields where image authenticity is crucial. The process typically involves several steps and employs various techniques to identify discrepancies and inconsistencies that indicate manipulation.

2.3.1 Preprocessing Digital Image

Block-based image processing is a widely used method for extracting features from images. In this approach, the image is divided into smaller chunks—typically square in shape—which may be either overlapping or non-overlapping. This method is effective because each block is processed independently using only the data contained within it, allowing for localized feature extraction [13]. Two critical factors influencing the performance of block-based processing are block size and the degree of overlap. Overlapping blocks can capture richer contextual information across regions. This technique has been shown to be powerful in various applications such as image compression, pattern recognition, and object detection, due to its ability to reduce computational complexity and enhance local analysis.

2.3.2 Feature Extraction and Feature Selection

Feature Selection plays a role in enhancing the performance of SVM classifiers in image processing and pattern recognition applications. Furthermore, many types of processes can be used by breaking down signals into their constituent frequency components, the Discrete Wavelet Transform (DWT) can extract high and low-frequency information that enhances classification discriminative ability. Moreover, by comparing pixel values with their neighbours, Local Binary Pattern (LBP) characterizes local image structures and produces histograms that are resistant to texture alterations and grayscale variations, which makes them useful for SVM-based classification [14]. However, when using the SVM classifiers method, Error Level Obtaining (ELA) is specifically designed to identify digital image forgeries by highlighting differences in error levels caused during image compression, which enables the successful separation of authentic and spliced sections when applied with SVM classifiers these four techniques DWT, LBP, ELA, and HOG contribute significantly to the overall performance and accuracy of SVM classifiers by obtain essential aspects of the data and enabling the effective pattern recognition and classification.

2.3.3 SVM classifiers

Recently, using image editing tools to create a forged image has been growing without having the manner in some people, and very challenging to detect the image although in recent years many researchers have focused on how to solve the problem .Besides that, a Support Vector Machine (SVM) is a machine learning model used in image forgery detection to classify whether images as genuine or forged based on features extracted from the images. SVM is particularly effective in handling complex, high-dimensional data and creating clear class separations. It also can be trained on a dataset of genuine and forged images to learn to distinguish between the two classes based on their features on the other hand, The SVM algorithm creates a decision boundary that effectively separates genuine images from forged ones, with support vectors currently playing a

crucial role in observing this boundary. Afterwards, with a new image the SVM technique uses an algorithm to know the decision boundary to classify the image as either genuine or forged, leveraging its ability to handle complex data and create clear class separations.

2.4 Related Work

From this Perspective, the researchers developed an early method for detecting copy-move forgery by dividing the image into overlapping blocks and applying Principal Component Analysis (PCA) to reduce dimensionality. The method for feature vectors was then classified using SVM to identify duplicated regions, demonstrating the efficiency of detecting the copied and pasted within the image.

Work by [15] introduced a technique that can be utilized by the Discrete Cosine Transform (DCT) to make process overlapping blocks of the image. Furthermore, by transforming the blocks into the frequency domain, the method could more easily detect similarities, and SVM was used for classification. This approach has shown the robustness in identifying regions that had been copied and moved within an image. Investigated the application of analysis based on the Discrete Wavelet Transform (DWT) to identify image splicing. Furthermore, the image utilized Support Vector Machines (SVM) to differentiate between authentic images by examining the properties of wavelet coefficients. Also, the approach successfully detected inconsistencies resulting from merging sections of images.

In a study, by the combination of features from both DWT and DCT domains was employed to enhance splicing detection capabilities. The hybrid feature set underwent processing using an SVM classifier achieving accuracy, in discerning between unaltered images. The method proposed underscored the benefits of utilizing feature domains for robust detection purposes to meet all the processing achieved.

Researcher has proposed techniques that can detect the hidden message because it is more considerably difficult to detect without any method, moreover by using the SVM feature extracted from the pattern, their technique could be identified the retouched image even if the image has been a modification. Work by [16] focused on detecting retouching by analysing images by analysing the digital image forensic that detects the local contrast by identifying to achieve the detection with less than a false positive rate. Furthermore, the technique that is used to detect noise on JPEG compressed images effectively.

Work by [17] proposed the method that focuses the original image from the altered image by using Binary Similarity Measure. Other than that, this method can classify effectively to know the original image in contrast enhancement and brightness adjustment attacks and can detect the different tampered images and introduced to the method to enhance tampering localization performance, this analysis sophisticated methods for digital picture forensics, with a particular emphasis on multi-scale analysis and decision fusion techniques. Moreover, the research involves developing algorithms for tampering detection using Conditional Random Fields (CRF) and correlation predictors, as well as adaptive parameter selection and data term computations. The benefits of the study method methods for forgery detection are emphasized, with reliability in improving the detection of the multi-scale analysis.

Table 1 *Comparative analysis of various image forensics using different techniques*

Title	Dataset	Features	Result Accuracy
DWT and LBP hybrid feature based deep learning technique for image [18]	IFC – TC, DVMM, Columbia, Casia	Discreate Wavelet Transaction (DWT) Local Binary Pattern (LBP)	• **IFC – TC** DWT = 92.45 %, LBP = 96.86% DWT + LBP = 98.31% • **DVMM** DWT = 94.27%, LBP = 95.13%, DWT + LBP = 98.59% • **Columbia** DWT = 93.23%, LBP = 97.84%, DWT + LBP = 98.87% • **Casia** DWT = 92.70%, LBP = 96.58%, DWT + LBP = 99.16%
Detecting image manipulation with ELACNN integration: a powerful framework for authenticity verification [19]	CASIA 2.0	ELA, CNN Statistical Features	• 99.05%

Applied Information Technology and Computer Science Vol. 6 No. 1 (2025) p. 560-575

Table 1*(Cont.)*

Title	Dataset	Features	Result Accuracy
Image Splicing Detection Scheme Based on Error Level Analysis and Local Binary Pattern [20]	Columbia, MICC F220, MICC 2000,	ELA + LBP	• COLUMBIA = 91.46% • MICC F220 = 94.09% • MICC 2000 = 97.30%
Histograms of Oriented Gradients for Human Detection [21]	MIT Pedestrian, INRIA Dataset	HOG	• Achieved essentially perfect results • Performance by 5% at a false positive rate of 10^{-4} FPPW, with overall accuracy reaching 89% at this rate
HOG Feature Extraction for Image Forgery Detection [22]	CASIA v1.0, CASIA v2.0	HOG	• CASIA v1.0 = 70.23%
Image Manipulation Detection Using Error Level Analysis [23]	CASIA	ELA, PCA	• 99%

Table 1 shows that the Comparative analysis of various image forensics using different techniques, table highlights various techniques and datasets used for image forgery detection. Among the methods, hybrid approaches combining features like Discrete Wavelet Transform (DWT) and Local Binary Patterns (LBP) consistently achieve high accuracy, with CASIA datasets reaching up to 99.16% when these features are combined. Similarly, techniques leveraging Error Level Analysis (ELA) integrated with CNN or PCA yield near-perfect results, such as 99.05% on CASIA 2.0 and 99% on CASIA datasets, respectively. For splicing detection, ELA combined with LBP performs well, achieving accuracies up to 97.30% on MICC 2000. Histograms of Oriented Gradients (HOG) are also effective, especially for human detection, with overall accuracy reaching 89% on pedestrian datasets, though their performance for forgery detection is moderate at 70.23% on CASIA v1.0. Overall, combining multiple features enhances detection accuracy across various datasets.

3. Research Methodology

Performance of the analysis provides an evaluation approach to conducting research and the process includes the guidance from process inquiry, analysis and interpretation. The research proposes splicing forgeries detection using methodologies like Local Binary Patterns (LBP), Histogram of Oriented Gradients (HOG), Error Level Analysis (ELA) and Support Vector Machines (SVM) for the outlined as follows. The Fig. 3 shows the framework of this research.

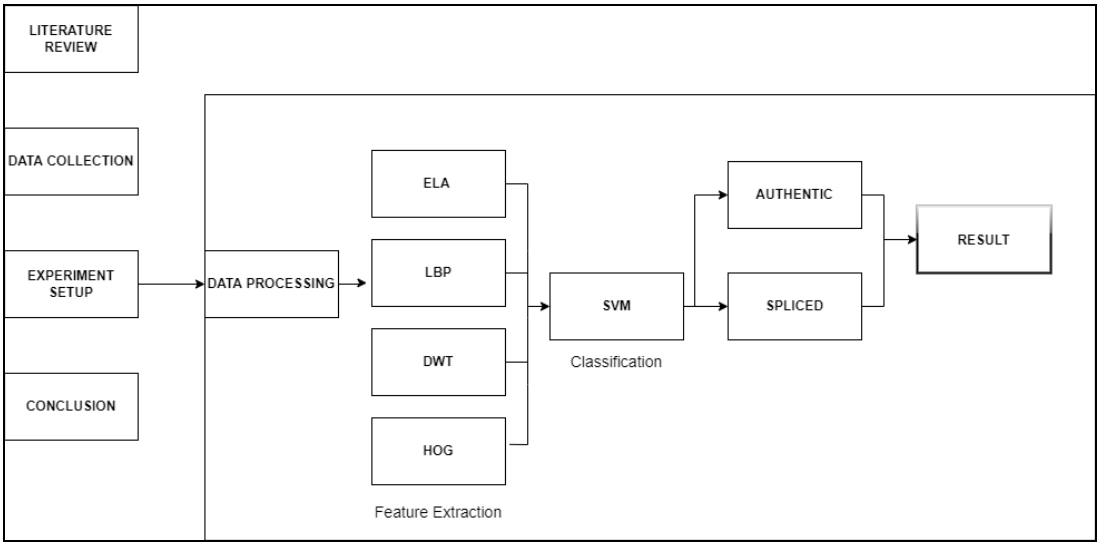

Fig. 3 *Research Framework*

In Fig. 3, it shows that the Research Framework illustrates the beginning of the analysis until it has the result with effective method and features extraction. Following this, multiple techniques that applied to ensure their robust and accuracy of different techniques.

3.1 Data Collection

The performance of image splicing was evaluated using two datasets which are the critical step to find out what suitable datasets and hardware tool are used in the research analysis. Firstly, the hardware used for this research also plays a crucial execution of this experimental, the primary device is Laptop Acer Aspire 5 (AS15-56) and the laptop operated in Windows 11 that equipped with Intel Core i5 (11th Gen) processors and additional the 8 GB RAM to ensure their smooth performance multitasking. For this research benchmark datasets CASIA v1.0 (Bhavya Shah,2023) and CASIA v2 (Dong, Wang & Tan, 2013) are utilized to recognize and provide images that contain authentic are image is unaltered, and it not being modified and for the spliced images is image that has been modified. Fig. 4 illustrates a prototype framework, highlighting the development process from initial analysis to the results. It incorporates effective methods and feature extraction techniques, supported by multiple approaches to ensure robustness and accuracy.

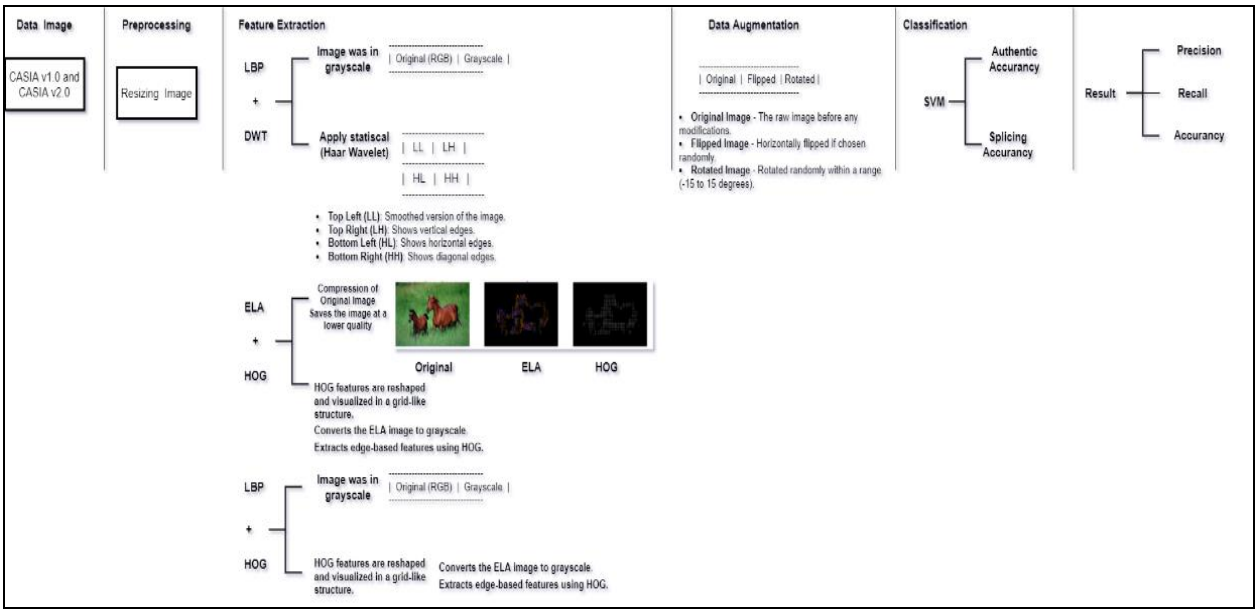

Fig. 4 *Prototype Framework*

3.2 Prototype Framework

The experimental setup includes data images that have datasets called CASIA v1.0 and CASIA v2.0 and data processing images like reduced size of image to make smoother when making a classification. Next, extracting the features like using ELA, LBP, DWT and HOG. These techniques will help to uncover the inconsistencies in image texture and compression of the image. Furthermore, once it has been done for the feature extraction it will classifier using SVM that can evaluate the result which it authentic or spliced. Data Processing needs to be done first to prepare a raw analysis and classification that needs to reduce the size of the image to ensure that the machine learning SVM classification process accurately detect the manipulated image.

3.2.1 Feature Extraction

The Feature Extraction method that has been proposed plays a vital role in this forgery detection that is applied in various processing fields. For this section, feature extraction that have been chosen during analysis are ELA, LBP, DWT and Gabor Filter that can reveal any potential that indicates to have tampered and detected the accuracy of the images. ELA would be highlighted on the images that have been altered and it to reveal the level of compression that used can introduce the certain area and artifact that have been saved a modified [18]. Moreover, it calculates the error level that significantly occurs in re-saving image and probably happens when it is in format JPEG that can be measured. Other than that ELA also has their limitation when it comes to image processing, complexity and compression setting and it can provide a true positive rate and negative positive rate. LBP would say that it is a textured descriptor to capture any object that appeared within the image using comparison underlying the pixel to make encoded in image [19]. After that it can come up with the binary pattern that produces local texture within a region. The LBP function is to make converting images to grayscale, comparing the pixel that occurs and generating histogram LBP code and that should be to make robust of the texture of the images. Additionally, it is more important to identify what are the actual and what are the subtle alterations in texture in image. HOG is the extracted texture of the feature set any filter that appoint in any

orientation and frequencies. Then it can capture the frequency of the information that can be used to know the authentic and spliced images and edges or gradient to provide robustness in the image transformation based on geometric and photometric are also statistical in the image that can be decomposed on different measures and frequency like mean and standard deviation. They have three types of dependencies of the wavelet, but it depends on the scales, orientation and position. Next, for the data augmentation it only does flip the original image for improve generalization.

3.2.2 Feature Classification

For the part of the model training, it would be said that the processing phase that takes as a crucial part in the development because it has the classification as all know that SVM are the supervised learning that particular make the effective of the classification during the task so it can know the authentic and spliced images occur. Furthermore, when the training phase started with the preparation of the feature extraction, it would present the vector feature to split for training and testing set to know the performance of model. Then, it used also to have applied hyperparameter in higher dimensional divided into two classes which it authentic and spliced with their maximum margin. Other than that, for better optimizing the performance of it explored like polynomial, linear and radial basis function kernels (RBF), all the functions to optimize can significantly increase the chances to have best configuration and side of for unseen data like do a cross- validation to generalize well. Additionally, when the model images have been trained it can be concluded to evaluate the accuracy, recall, F1score and precision and they all have been more effective and efficient to identify where the areas that had been normalized detection and to make them more enhancing for the side of reliability of image evidence.

3.2.3 Performance Metric

Finally, the results of the experiment and discusses the effectiveness of the algorithm was evaluated based on three metrics which are precision, recall, and F1-score. The calculation shown the performance metric for the detection as shown in Fig. 5.

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F - meaure : } F = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Fig. 5 *Performance metric*

The performance metric has feature are True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN). True Positives are the spliced pixels that have correctly been marked spliced which means that the algorithm performed well in detecting the spliced regions. True Negatives are the non-spliced pixels that are correctly marked as non-spliced, show that the effectiveness of the model on reducing the chances of generating false alarms in regions which have not been altered. Furthermore, False Positives arise when non-spliced pixels are marked as spliced meaning there were no changes done to those areas. False Negatives are defined as spliced pixels that misrecognized to un-spliced, which indicates that such algorithm failed to notice that these pixels have been modified.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Fig. 6 *Formula accurancy*

Taken together as shown in Fig. 6 are the calculation of accuracy. That had been apply when the metric is done evaluation of the capabilities and limitations of the algorithm relative to the image splicing detection task. The calculation shown the performance metric for the detection.

4. Experimental Setup

The detection of image tampering has become essential in the age of digital images. Therefore, this analysis focuses on adversarial image detection by combining feature extraction techniques and machine learning algorithms, SVM. The analysis results in using different feature extraction techniques to process the image to establish where the image has been tampered with and identify the region on the images.

4.1 LBP + DWT Feature Extraction

This section presents an explanation about the details and dataset of how the algorithm executed for feature and classification.

4.1.1 Preprocessing Resizing Image

Preprocessing is the first step to be taken to make analysis input into model data, in general they include tasks like resizing image to reduce complexity by working with smaller images. In Fig. 7, the function resizes images to enlarge the image to ensure the pixel had uniformity with dataset and it to standardize the image. This happens when reducing any computational cost during feature extraction and classification with SVM.

```
# Function to resize images to reduce computational cost
def resize_image(image, target_size=(64, 64)):  # Reduced size for faster processing
    return cv2.resize(image, target_size)
```

Fig. 7 *Preprocessing (resize image)*

4.1.2 Feature Extraction LBP

Extract LBP feature function to focus on low frequency (LL) sub band that obtain from DWT. For the input validation it is to ensure that image was in grayscale situation as it operate in single channel images, but this current algorithm happens when input image was in three channel like grayscale using the OpenCV libraries. Then, it uses DWT to decompose the frequency of image into four sub band to identify different frequency which its LL for low – low, LH.HL. HH, it contains high frequency above in Fig. 8.

```
# Function to extract LBP features
def extract_lbp_features(image, max_features=500):  # Reduced max features
    if len(image.shape) == 3:
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    coeffs = pywt.dwt2(image, 'haar')
    LL, (LH, HL, HH) = coeffs
    radius = 1
    n_points = 8 * radius
    lbp_LL = local_binary_pattern(LL, n_points, radius, method='uniform')
    hist_LL = np.histogram(lbp_LL, bins=59, range=(0, 59))[0]
    features = hist_LL
    return features[:max_features]
```

Fig. 8 *Extract LBP features*

In Fig. 8, next process is to extract the LBP pattern by encoding intensity into binary pattern to applied in LL sub band to determine the radius, n_point and method for the robustness improvement and for the histogram values of LBP it computes 59 bins to correspond for uniform LBP pattern for 8 point and the range to represent only 0 – 59 values.

4.1.3 Feature Extraction DWT

The function is processed when input images want to extract DWT features where it has statistical like mean and deviation on different sub band (LL.LH. HL, HH) this four-sub band can help with this operation. In Fig. 9, apply the Haar Wavelet to the grayscale image that has high and low frequency and extract the statistical features from four each sub band to represent the computing the mean of pixel average value and standard deviation to measure the values. This happens when each of high and low frequency has their function like LL has all overall the structure image and High for edges, fine pattern and edges.

```python
# Function to extract DWT features
def extract_dwt_features(image):
    # Convert to grayscale if the image is in color
    if len(image.shape) == 3:
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Apply DWT (Haar wavelet)
    coeffs = pywt.dwt2(image, 'haar')
    LL, (LH, HL, HH) = coeffs

    # Extract statistical features (mean, standard deviation) from the DWT coefficients
    features = []
    for band in [LL, LH, HL, HH]:
        features.append(np.mean(band))
        features.append(np.std(band))
    return features
```

Fig. 9 *Extract DWT features*

4.1.4 Data Augmentation

To apply the random transformation and creating augmented to increase of dataset to help improve machine learning task to generalize better during the training and it would simulate the real-world variations such as flip and rotation without any added data collection as shown in Fig. 10. It generates 0 and 1 for random floating between them to have the useful model invariant. Next, the image dimension retrieved the rows and cols to center the image. For the input images it may be randomly flipped horizontally are might be rotated slightly in some cases both have transformations occur.

```python
# Function for dynamic data augmentation
def augment_image_dynamically(image):
    if np.random.rand() > 0.5:
        image = cv2.flip(image, 1)
    if np.random.rand() > 0.5:
        rows, cols = image.shape[:2]
        M = cv2.getRotationMatrix2D((cols/2, rows/2), np.random.randint(-15, 15), 1)
        image = cv2.warpAffine(image, M, (cols, rows))
    return image
```

Fig. 10 *Data Augmentation*

4.1.5 Model Training (Hyperparameter Tuning)

SVM would choose the best hyperparameter tuning with RandomsizedSearchCV to get the best model to make predictions as shown in Fig. 11. This part would be the crucial part because it is important for machine learning to standardize the scale. The features mean 0 and standard deviation of 1 where found that could have sensitive scale on the feature. Next, the reduction of information like of features using PCA can speed up the training process and reduce overfitting.

```python
# Standardize the features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Apply PCA for dimensionality reduction
pca = PCA(n_components=50)
X_reduced = pca.fit_transform(X_scaled)

# SVM Hyperparameter tuning with RandomizedSearchCV
param_grid = {
    'C': [0.1, 1, 10],
    'gamma': [0.01, 0.1],
    'kernel': ['rbf', 'linear']
}
random_search = RandomizedSearchCV(SVC(probability=True), param_grid, cv=3, n_iter=10, n_jobs=-1)  # Reduced n_iter
random_search.fit(X_reduced, y)

best_model = random_search.best_estimator_
return best_model, pca, scaler
```

Fig. 11 *Hyperparameter tuning*

4.2 ELA + HOG Feature Extraction

This section presents an explanation about the details and dataset of how the algorithm executed for feature and classification.

4.2.1 Perform preprocessing ELA

To perform the function of ELA to determine the original image and image that has been compressed using JPEG compression, when it produces, they visually highlight the tampered which means different key identifying region manipulated appeared and all this kind method to make sure that between two images may indicate the splicing appear as shown in Fig. 12. The images were converted to RGB where it is more standard on color format then it does a process to quality specified compressed the image. Then the difference between compressed images has been calculated from the side of pixel – wise and they would be highlighting area discrepancies that may indicate splicing.

```
# Function to perform Error Level Analysis (ELA)
def perform_ela(image_path, quality=90):
    image = cv2.imread(image_path)
    if image is None:
        raise ValueError(f"Error: Unable to load image at {image_path}")
    image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

    temp_image_path = "temp_image_quality.jpg"
    cv2.imwrite(temp_image_path, cv2.cvtColor(image_rgb, cv2.COLOR_RGB2BGR),
                [int(cv2.IMWRITE_JPEG_QUALITY), quality])

    compressed_image = cv2.imread(temp_image_path)
    compressed_image_rgb = cv2.cvtColor(compressed_image, cv2.COLOR_BGR2RGB)

    ela_image = np.abs(image_rgb.astype(np.float32) - compressed_image_rgb.astype(np.float32))
    ela_image = np.uint8(np.clip(ela_image * 255.0 / np.max(ela_image), 0, 255))  # Normalize ELA
    return ela_image
```

Fig. 12 *Perform ELA*

4.2.2 Extraction HOG

The feature extraction explained HOG functions are dependent on color information which means that apply HOG features are to operate on their intensity gradient to grayscale and convert into one channel intensity. Then, the function encoded the gradient magnitude and texture or structural in the image as shown in Fig. 13. The hog captures edges and their orientation, texture of pattern in the images which can distinguish between object or detect anomalies and object detection like other images in the pictures

```
# Function to extract HOG features
def extract_hog_features(image):
    gray_image = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)  # Convert to grayscale
    features = hog(gray_image, block_norm='L2-Hys', pixels_per_cell=(8, 8), cells_per_block=(2, 2), orientations=9)
    return features
```

Fig. 13 *Perform Extraction HOG*

4.2.3 Model Training (K-Fold Cross Validation)

After combining the ELA and HOG, the code was implemented by using K – Fold Validation using Stratified to perform the performance and using Radial Basis Function (RBF). Furthermore, the function of the K - Fold Cross - Validation that preserves class on each fold is StratifiedFold. In Fig. 14, for the first step, it needs to be splitting the data for the set x and y into train and test, then using RBF to get the cross-validation accuracy score. The SVM model using RBF kernel like 5-Fold Stratified Cross Validation would provide a robust assessment to classify the authentic and spliced.

```python
# k-Fold Cross-Validation
kf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
cv_accuracies = []

for train_index, test_index in kf.split(X, y):
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Use SVM with RBF kernel
    clf = SVC(kernel='rbf', C=1.0, class_weight='balanced', probability=True, random_state=42)
    clf.fit(X_train, y_train)

    y_pred = clf.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    cv_accuracies.append(accuracy)

print(f"Cross-Validation Accuracy: {np.mean(cv_accuracies) * 100:.2f}%")
```

Fig. 14 *Model Training (K- Fold cross validation)*

4.3 HOG + LBP Feature Extraction

This section presents an explanation about the details and dataset of how the algorithm executed for feature and classification.

4.3.1 Feature Extraction LBP

For this part they have a list of LBP features that need to be extracted and initialize the empty list of features will store all the images and for this part must calculate the histogram of the LBP by normalizing the histogram as shown in Fig. 15. It goes through loops each of the paths and then reads a grayscale image, as LBP operates on a single channel. Other than that, the LBP features were calculated and computed on histogram.

```python
# Function to extract LBP features from an image
def extract_lbp_features(image_paths, radius=1, n_points=8):
    features = []
    for image_path in image_paths:
        image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
        image = cv2.resize(image, (128, 128))  # Resize image to standard size
        lbp = local_binary_pattern(image, n_points, radius, method='uniform')

        # Calculate the histogram of LBP
        lbp_hist, _ = np.histogram(lbp.ravel(), bins=np.arange(0, n_points + 3), range=(0, n_points + 2))
        lbp_hist = lbp_hist.astype(np.float32)
        lbp_hist /= (lbp_hist.sum() + 1e-6)  # Normalize the histogram
        features.append(lbp_hist)
    return np.array(features)
```

Fig. 15 *Feature Extraction LBP*

In Fig. 16, features extraction are steps like the LBP and for HOG are extracts the oriental gradient because it is like feature descriptor to detect image classification for capturing structure or texture of image. Furthermore, it can combine features such as concatenation horizontally to creating a single feature vector per image.

```python
# Function to extract HOG features from an image
def extract_hog_features(image_paths, pixels_per_cell=(8, 8), cells_per_block=(2, 2), visualize=False):
    hog_features = []
    for image_path in image_paths:
        image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)
        image = cv2.resize(image, (128, 128))  # Resize image to standard size

        # Extract HOG features
        if visualize:
            features, hog_image = hog(image, pixels_per_cell=pixels_per_cell, cells_per_block=cells_per_block, visualize=visualize)
        else:
            features = hog(image, pixels_per_cell=pixels_per_cell, cells_per_block=cells_per_block, visualize=visualize)
        hog_features.append(features)
    return np.array(hog_features)
```

Fig. 16 *Feature Extraction HOG*

4.3.2 Data Preprocessing (PCA)

For the next step after standardizing the data to ensure all in same scale, it needs to determine the number samples PCA and initialize the dimensions to reduce the space that it has in the feature as shown in Fig. 17. Then, fit the PCA to compute the linear combinations of the original feature. The use of PCA is when in small dataset in many features and to improve computational model.

```
# Perform PCA for dimensionality reduction (optional)
pca_components = min(X_train.shape[0], X_train.shape[1])  # Set to the smaller of number of samples or features
pca = PCA(n_components=pca_components)  # Reduce number of components for faster processing
X_train = pca.fit_transform(X_train)
X_test = pca.transform(X_test)
```

Fig. 17 *Apply PCA*

4.3.3 Model Training (Hyperparameter Tuning)

The training is performed in hyperparameter tuning using GridSearchCV using cross-validation to train multiple SVM models with combination grom the grid training data. Next, evaluate the model to make predictions and accuracy when it had been labelling for the test data as shown in Fig 18. In Fig 18, select the best combination based on the cross – validation and report in accuracy, classification metrics and confusion matrix. Here above shown that the flowchart on how they operate to perform the analysis.

```
# Hyperparameter tuning for SVM
print("Performing hyperparameter tuning...")
param_grid = {
    'C': [0.1, 1, 10],
    'gamma': ['scale', 'auto'],
    'kernel': ['rbf', 'linear', 'poly'],
    'class_weight': [None, 'balanced']  # Added class weight to handle imbalance
}
grid_search = GridSearchCV(SVC(), param_grid, cv=3, n_jobs=-1)  # Use all CPU cores
grid_search.fit(X_train, y_train)

# Print best parameters
print(f"Best Parameters: {grid_search.best_params_}")

# Evaluate the model
y_pred = grid_search.predict(X_test)
test_accuracy = accuracy_score(y_test, y_pred)
print(f"Test Accuracy: {test_accuracy * 100:.2f}%")
print("Classification Report:")
print(classification_report(y_test, y_pred))

# Confusion matrix
cm = confusion_matrix(y_test, y_pred)
plot_confusion_matrix(cm, classes=['Authentic', 'Spliced'])
```

Fig. 18 *Apply SVM (Hyperparameter Tuning)*

4.4 Dataset (CASIA v1.0 and CASIA v2.0)

Dataset that evaluated to perform the technique are CASIA v1.0 and CASIA v2.0 are available publicly and can be downloaded as shown in Table 2. But in this case, the dataset had been limited for testing.

Table 2 *Dataset Casia v1.0 and Casia v2.0*

Algorithm	Dataset
LBP + DWT	CASIA v1.0 = 200
	CASIA v2.0 = 200
LBP + HOG	CASIA v1.0 = 150
	CASIA v2.0 = 150
ELA + HOG	CASIA v1.0 = 100
	CASIA v2.0 = 100

5. Results and Discussion

The proposed method for this different algorithm and with two different benchmarks dataset will discuss the result in detail as shown in the table above and the experimental would evaluates by comparing the best performance and each algorithm would base on the Precision, Recall, F1-score and accuracy.

First, ELA and HOG combine algorithms with the best performance because on the CASIA v1.0 it achieves 94.00% where it is resulting more on spliced image. Furthermore, for the CASIA v2.0 it improves slightly 95.50% which is more solid result in both precision and recall having 99% for class 0 for authentic image and for class 1 are spliced get also 99%. Based on the result achieved, it outperformed that both combinations got the highest best accuracy, and it may be on the analysis the train test, processing and extracting the features consistently performed smoothly and better.

Next, the combination of algorithm HOG and LBP was shown to drop their performance because it very least challenges extreme when it combines, and the result was on CASIA v1.0 it had low precision 30 % and 55% for class 0 and it better perform in class 1 74% recall and overall leading to accuracy are 51.33%. Moreover, for dataset CASIA v2.0 with class 0 had a good precision while in class 1 got had drop drastically to 10% and overall, for the accuracy resulting are 78.11%.

Besides that, for the next algorithms are LBP and DWT which combination that get a moderate performance. Dataset CASIA v1.0 it got achieve of the accuracy of 60.5% with leading to precision for class 1 got 61% thus it has moderate in recall for class 0 in recall and 57% for class. Next, the accuracy overall got 60.50% but it still got a low performance. In addition, on CASIA v2.0 the performance improves to 69.25% with higher precision but on class 0 81% and goof recall for 88%. However, the recall got drop-in class 0 51% this will indicate that have an imbalance detect for authentic. In a nutshell ELA and HOG outperform other than algorithms and across both datasets with delivering with high accuracy and the result have well balanced, while for LBP and HOG and LBP and DWT got a challenge during the analysis especially in recall and accuracy.

Table 3 *Performance with combinations of algorithms*

Algorithm	Dataset Used	Precision	Recall	F1-score	Accuracy
ELA + HOG	Casia v1.0	Class 0: 99%	Class 0: 89%	94.00%	94.00%
		Class 1: 90%	Class 1: 99%		
	Casia v2.0	Class 0: 99%	Class 0: 92%	95% -96%	95.50%
		Class 1: 93%	Class 1: 99%		
HOG+ LBP	Casia v1.0	Class 0: 55%	Class 0: 30%	Class 0: 39%	51.33%
		Class 1: 50%	Class 1: 74%	Class 1: 59%	
	Casia v2.0	Class 0: 78%	Class 0:100%	Class 0: 87%	78.11%
		Class 1:100%	Class 1: 10%	Class 1: 19%	
LBP+ DWT	Casia v1.0	Class 0: 60%	Class 0: 64%	Class 0: 62%	60.50%
		Class 1: 61%	Class 1: 57%	Class 1: 59%	
	Casia v2.0	Class 0: 81%	Class 0: 51%	Class 0: 62%	69.25%
		Class 1: 64%	Class 1: 88%	Class 1: 74%	

6. Conclusion

The corresponding details explanation about the objective that implementing various feature extraction technique and that has been analysis using three algorithms with two different benchmarks dataset to determine the effective combination for image splicing detection. The findings highlight the importance of efficient feature extraction and algorithm compatibility for successful image splicing detection.

6.1 Objective Achievement

The objective of this research was successfully met all by using the proposed method for image splicing using various techniques features extraction and make the classification with two types of SVM such as hyperparameter tuning and K-Fold Cross Validation. Then for the first objective it achieved by implementing them by integrating the extraction ELA, HOG, LBP and DWT features. Secondly, for the objective to improving the accuracy and well balancing, especially when it comes to algorithms ELA and HOG. Thirdly, the evaluating performance method was demonstrated that it becomes robust and achieves the high accuracy of into moderate and lower accuracy.

6.2 Research Limitation

While performing this research, they might be highlighted that faced certain limitations during analysis. In addition, the key to that is basically reliance on a small and imbalanced dataset hampers the ability to generalize effectively and some of the algorithms get a poor performance because the handcrafted feature has their complexities of splicing pattern. Then, inefficiencies and inadequate augmentation constrain the approach and for the interactive environment may not also have limitations from the google colab like computational power or make long processing toward to low GPU and TPU when it comes to highest dataset need to test.

6.3 Future Work

The future work in image splicing should be more prioritized on what are the limitations that have through the methodologies and broader dataset. In real world image splicing must expand the datasets with varying resolution and levels of compression need enhance the model generalization. Furthermore, the part of Convolutional Neural Network (CNN) can capture the complexity of patterns and can improve the accuracy of image. Besides that, optimization can use Bayesian hyperparameter tuning and ensemble learning to refine the performance, dimensionality reduction techniques like t-SNE or UMAP to make any challenging when faced with critical information during preprocessing. This enhancement will build a more scalable m reliable and effective system in real world image splicing detection. Finally, for the interactive environment, it may accessibly set up the dedicated GPUs like NVIDIA A100 or V100 to accelerate the model training and testing and it also provides to decrease the limitation of cloud services and its suite for using the deep learning tasks and large dataset.

To summarize, research for the impact of used features extraction methods such as LBP , DWT, ELA and HOG that demonstrate through the experimentation and analysis by the significant of detection performance which may detect which are the less effective and high effective by capturing the complex splicing pattern and despite for the limitation occur that provide the study about the importance of selecting extraction for more robust dataset .The findings of the future study particularly on exploring deep learning approaches and optimizing to enhance image splicing detection accuracy. Overall, the research contributes to the reliability and trustworthiness of evidence in forensic investigations.

Acknowledgement

The authors would like to thank the Faculty of Computer Science and Information Technology, University Tun Hussein Onn Malaysia for its support.

Conflict of Interest

Authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: **study conception and design:** *W. Y. B. Wan Omar Fathil, N. Rahim*; **data collection:** *W. Y. B. Wan Omar Fathil, N. Rahim*; **analysis and interpretation of results:** *W. Y. B. Wan Omar Fathil, N. Rahim*; **draft manuscript preparation:** *W. Y. B. Wan Omar Fathil, N. Rahim*, all authors reviewed the results and approved the final version of the manuscript.

References

[1] A. D. Warbhe, R. V. Dharaskar, and V. M. Thakare, "Computationally Efficient Digital Image Forensic Method for Image Authentication," *Physics Procedia*, vol. 78, pp. 464–470, 2016, doi: 10.1016/j.procs.2016.02.089.

[2] A. Thapaliya, D. E. Atonge, M. Mazzara, S. Chakraborty, and M. Ahmad, "Digital Image Forgery," 2019.

[3] P. Sharma, M. Kumar, and H. Sharma, "Comprehensive analyses of image forgery detection methods from traditional to deep learning approaches: an evaluation," Multimedia Tools and Applications, vol. 82, no. 12, pp. 18117–18150, 2023, doi: 10.1007/s11042-022-13808-w.

[4] Wei, X., X. Wei, Y. Wu, F. Dong, J. Zhang, and S. Sun, "Developing an image manipulation detection algorithm based on edge detection and faster R-CNN," Symmetry, vol. 11, no. 10, 2019, doi: 10.3390/sym11101223.

[5] S. Singh and R. Kumar, "Fake Image Identification using Image Forensic Techniques," pp. 95–100, 2022, doi: 10.5220/0010563000003161.

[6] M. Kumar Garg, "Image Forgery Detection and Classification using Support Vector Machine," JETIR, vol. 9, 2022.

[7] A. H. Mohammed, D. H. Badr, and F. Ali, "Detection of Image Forgery Using Information Standard Method with SVM," J. Phys.: Conf. Ser., vol. 1818, no. 1, 2021, doi: 10.1088/1742-6596/1818/1/012212.

[8] V. Christlein, C. Riess, J. Jordan, C. Riess, and E. Angelopoulou, "An Evaluation of Popular Copy-Move Forgery Detection Approaches," IEEE Trans. Inf. Forensics Security, 2012, doi: 10.1109/TIFS.2012.2218597.
[9] M. F. Hashmi, A. Hambarde, V. Anand, and A. Keskar, "Passive Detection of Copy-Move Forgery using Wavelet Transforms and SIFT Features," J. Inf. Assur. Secur., vol. 9, 2014. www.mirlabs.net/jias/index.html
[10] T. Thakur, K. Singh, and A. Yadav, "Blind Approach for Digital Image Forgery Detection," Int. J. Comput. Appl., vol. 179, no. 10, pp. 34–42, 2018, doi: 10.5120/ijca2018916108.34–42.
[11] C. N. Bharti and P. Tandel, "A Survey of Image Forgery Detection Techniques," in *Proc. IEEE WiSPNET*, Chennai, India, Mar. 2016, pp. 877–881.
[12] B. Mahdian and S. Saic, "Using noise inconsistencies for blind image forensics," Image and Vision Computing, vol. 27, no. 10, pp. 1497–1503, 2009.
[13] D. P. Tian and J. P. Tian, "A Review on Image Feature Extraction and Representation Techniques," *Int. J. Multimedia Ubiquitous Eng.*, vol. 8, no. 4, pp. 385–396, Jul. 2013.
[14] W. Luo, J. Huang, and G. Qiu, "A Survey of Passive Technology for Digital Image Forensics," Front. Comput. Sci. China, vol. 1, no. 2, pp. 166–179, 2007.
[15] Y. Kumar, R. Kumar, and R. Patel, "A Review on Image Forgery Detection Techniques Using Machine Learning," EasyChair Preprint 10580, Jul. 17, 2023.
[16] O. Mayer and M. C. Stamm, "Improved Forgery Detection with Lateral Chromatic Aberration," in *Proc. IEEE ICASSP*, Shanghai, China, Mar. 2016.
[17] H. Mareen, D. V. Bussche, F. Guillaro, et al., "Comprint: Image Forgery Detection and Localization Using Compression Fingerprints," in *Workshop on MMForWILD @ ICPR*, Milan, Italy, Aug. 2022.
[18] M. K. Singh, "DWT and LBP Hybrid Feature Based Deep Learning Technique for Image Splicing Forgery Detection," *Soft Comput.*, 2024, doi:10.1007/s00500-024-09919-1.
[19] A. M. Nagm, M. M. Moussa, R. Shoitan, A. Ali, M. Mashhour, and A. S. Salama, "Detecting image manipulation with ELA-CNN integration: a powerful framework for authenticity verification," PeerJ Comput. Sci., vol. 10, art. e2205, 2024, doi: 10.7717/peerj-cs.2205.
[20] Y. J. Zhang, T. T. Shi, and Z. M. Lu, "Image splicing detection scheme based on error level analysis and local binary pattern," Journal of Network Intelligence, vol. 6, no. 2, pp. 303–314, 2021.
[21] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, Jun. 2005, pp. 886–893, doi: 10.1109/CVPR.2005.177.
[22] R. Jaiswal, A. Yadav, and S. Sharma, "HOG feature extraction for image forgery detection," International Journal of Signal and Imaging Systems Engineering, vol. 11, no. 4, pp. 5–10, 2023.
[23] B. Shah, D. Shah, S. Thakar, S. Shah, and S. Dhage, "Image Manipulation Detection Using Error Level Analysis, 2023.