future internet

MDPI

Article

Detection of Forged Images Using a Combination of Passive Methods Based on Neural Networks

Ancilon Leuch Alencar [1], Marcelo Dornbusch Lopes [1], Anita Maria da Rocha Fernandes [1,*], Julio Cesar Santos dos Anjos [2], Juan Francisco De Paz Santana [3] and Valderi Reis Quietinho Leithardt [4,5,6]

1 Master Program in Applied Computer Science (MCA), Escola Politécnica, University of Vale do Itajaí (UNIVALI), Campus Itajaí, Itajaí 88302-901, Santa Catarina, Brazil; ancilon@edu.univali.com (A.L.A.); marcelo@univali.br (M.D.L.)
2 Graduate Program in Teleinformatics Engineering (PPGETI/UFC), Federal University of Ceará, Campus of Itapaje, Fortaleza 60455-970, Ceará, Brazil; jcsanjos@ufc.br
3 Expert Systems and Applications Laboartory, Escuela Técnica Superior de Ingeniería Industrial de Béjar, University of Salamanca, 37700 Salamanca, Spain; fcofds@usal.es
4 Lisbon School of Engineering (ISEL), Polytechnic University of Lisbon (IPL), 1549-020 Lisbon, Portugal; valderi.leithardt@isel.pt
5 FIT-ISEL, 1959-007 Lisboa, Portugal
6 Center of Technology and Systems (UNINOVA-CTS) and Associated Lab of Intelligent Systems (LASI), 2829-516 Caparica, Portugal
* Correspondence: anita.fernandes@univali.br

Abstract: In the current era of social media, the proliferation of images sourced from unreliable origins underscores the pressing need for robust methods to detect forged content, particularly amidst the rapid evolution of image manipulation technologies. Existing literature delineates two primary approaches to image manipulation detection: active and passive. Active techniques intervene preemptively, embedding structures into images to facilitate subsequent authenticity verification, whereas passive methods analyze image content for traces of manipulation. This study presents a novel solution to image manipulation detection by leveraging a multi-stream neural network architecture. Our approach harnesses three convolutional neural networks (CNNs) operating on distinct data streams extracted from the original image. We have developed a solution based on two passive detection methodologies. The system utilizes two separate streams to extract specific data subsets, while a third stream processes the unaltered image. Each net independently processes its respective data stream, capturing diverse facets of the image. The outputs from these nets are then fused through concatenation to ascertain whether the image has undergone manipulation, yielding a comprehensive detection framework surpassing the efficacy of its constituent methods. Our work introduces a unique dataset derived from the fusion of four publicly available datasets, featuring organically manipulated images that closely resemble real-world scenarios. This dataset offers a more authentic representation than other state-of-the-art methods that use algorithmically generated datasets based on image patches. By encompassing genuine manipulation scenarios, our dataset enhances the model's ability to generalize across varied manipulation techniques, thereby improving its performance in real-world settings. After training, the merged approach obtained an accuracy of 89.59% in the set of validation images, significantly higher than the model trained with only unaltered images, which obtained 78.64%, and the two other models trained using images with a feature selection method applied to enhance inconsistencies that obtained 68.02% for Error-Level Analysis images and 50.70% for the method using Discrete Wavelet Transform. Moreover, our proposed approach exhibits reduced accuracy variance compared to alternative models, underscoring its stability and robustness across diverse datasets. The approach outlined in this work needs to provide information about the specific location or type of tempering, which limits its practical applications.

Keywords: digital image forensics; convolutional neural network; deep learning

Citation: Alencar, A.L.; Lopes, M.D.; Fernandes, A.M.d.R.; Anjos, J.C.S.d.; De Paz Santana, J.F.; Leithardt, V.R.Q. Detection of Forged Images Using a Combination of Passive Methods Based on Neural Networks. *Future Internet* **2024**, *16*, 97. https://doi.org/10.3390/fi16030097

Academic Editor: Gianluigi Ferrari

Received: 2 February 2024
Revised: 28 February 2024
Accepted: 11 March 2024
Published: 14 March 2024

1. Introduction

Fake news, the dissemination of false information via social networks, has become a pervasive issue in today's digital landscape. Such misinformation has the potential to deceive and garner more attention than factual news due to its deliberate crafting to evoke strong reactions [1]. Notably, events like the Higgs Boson exploration in 2012 generated a flurry of false information, with as many as 600 tweets per minute [2]. Of particular concern is the use of manipulated images, which can implicitly lend credibility to false narratives, thereby eroding public trust [3,4].

While numerous studies delve into identifying manipulation in images, many focus on datasets containing only specific tampering types for example [5,6], or contain datasets algorithmically generated such as [7]. Other papers use numerous more specific types of manipulation; for example, Liu and Pun [8] mentioned a kind of manipulation to remove parts of an image and replace them with surroundings. Besides, Rocha et al. [9] mentioned healing as a manipulation capable of softening features.

With the intention of categorizing manipulations in as few types as possible while still addressing all forms of manipulation, this paper adopts the categorization of image manipulations proposed by [10,11]. Along with copy-move and splicing, this categorization recognizes retouching as a manipulation technique that involves slightly modifying an image content without completely hiding large portions. Those modifications include applying filters, highlights, resizing, inpainting, or rotating parts of an image.

There are two general approaches to identifying image manipulation: active and passive. Active techniques, also known as preemptive, aim to preemptively insert structures in an image that can be used to detect any future changes. These structures can be visible to the human eye, like in watermarking, or hidden using steganography techniques. Manipulation can be reliably detected by assessing the integrity of the inserted structure. However, this approach requires that the structure be inserted correctly in a trustworthy image capture before any manipulation, as these methods only enable the detection of modifications post-insertion [12].

Meanwhile, passive methods, also known as blind methods, use only the content already present in an image and do not require any prior action, making them better suited for use in social media. To detect image forgery, passive methods can either analyze artifacts intrinsic to a digital image or search for inconsistencies in the content of an image. According to Lubna and Chowdhury [12], the artifacts present in digital images can be divided into three types: acquisition, format, and manipulation.

Acquisition artifacts are introduced in the image either by imperfections created by the manufacture of camera sensors called fixed-pattern noise or by programs used by cameras to process sensor data before storage; the introduction of artifacts in this manner usually follows a predictable pattern, and any divergences to it is evidence of manipulation.

Format artifacts, introduced in the digital storage of images by algorithms like the jpeg compression, by removing information less relevant to human eyes, areas of the image with artifacts of this type different than expected indicate the image was manipulated. Manipulation artifacts, that get introduced during image manipulation by a program, like the application of a blur filter that alters areas of an image in predictable patterns, the presence of those known artifacts in areas of an image is evidence of manipulation.

The other possible approach involves analyzing the content of an image for inconsistencies. Some inconsistencies are duplicated portions of an image, unnaturally sharp edges, shadow inconsistencies, and perspective inconsistencies [13]. Unfortunately, all those methods of detection have their pros and cons. There is no absolute best method capable of detecting all types of manipulation. Therefore, to better understand detection methods and their limitations, it is essential to understand how images can be manipulated in the first place [14].

There has yet to be a consensus on classifying all image manipulations; however, most papers recognize the existence of at least two types, copy-move, and splicing. In copy-move forgery, an area of an image is duplicated and pasted over another area of the same image.

In splicing, however, an area of another image is pasted over the image. This manipulation can add new meaning to the image or conceal information; the resulting image is a fusion of two different images.

Following the discussion on various approaches to identifying image manipulation, it is essential to highlight how this paper contributes to advancing the field. While existing studies often focus on datasets limited to specific tampering types or utilize algorithmically generated datasets (e.g., [15–18]), this research employs a more comprehensive dataset. Notably, this dataset is manually curated and designed to encompass a broad spectrum of realistically manipulated images, covering all types of manipulation encountered in real-life scenarios.

This methodological improvement overcomes the limitations of previous studies, ensuring that the detection model is trained on diverse and representative data. By incorporating a wider range of manipulations, including subtle retouching techniques such as filters, highlights, resizing, and inpainting, the proposed approach enhances the model's ability to generalize and accurately detect manipulation across various contexts.

Furthermore, our approach capitalizes on established passive detection methods within a multi-stream neural network architecture, as outlined in the abstract. This integration enables the utilization of knowledge acquired from traditional detection methods to enhance the accuracy of neural network-based approaches. By leveraging this synergy, our method aims to significantly mitigate overfitting, a prevalent challenge in such methodologies.

Besides this introductory section, the following sections aim to explain better how the proposed approach works; in Section 2, we discuss several relevant related works. Section 3 presents the methods, the dataset used, and the model architecture; then, Section 4 discusses the results we obtained. Finally, Section 5 presented the final considerations and suggestions to improve the approach.

2. Related Works

A problem in defining the search string consisted of the significant variability of terms used to refer to the addressed subject. For example, terms sometimes used to refer to counterfeits are "falsification", "tampering", "counterfeiting", "adulteration", "forgery", "manipulation", "edited", "doctored" or "altered". After some tests, we defined the following search string: "image AND (tamper OR forged OR forgery) AND (detect OR localize) AND NOT video". This search returned a considerable amount of work. To delimit the search, using the first search string, we adopted the following criteria:

- The presented detection method must follow the passive approach;
- The Detection method should primarily focus on verifying digital images' authenticity; and
- Must be in the top five most relevant results of each base that follows the other criteria.

With this research, we find 15 works. However, most of the papers returned on this initial search did not claim to be capable of detecting all types of manipulation and primarily focused on a single type, therefore, to include works related to general identification, terms such as "global" or "universal" were tested. Unfortunately, not all works that detect all types of manipulation used those terms, so the search key had to be made less specific: "image AND (tamper OR forged OR forgery) AND (detect OR localize)" and to limit the search the following criteria were employed:

- Introduce a method of detecting general-purpose manipulated images in their text;
- The presented detection method must follow the passive approach;
- The presented method should not require file formats with data compression;
- The Detection method should primarily focus on verifying digital images' authenticity;
- The paper was published in the last five years; and
- The paper must have the most relevant results of each base that follow the other criteria.

From this search, three additional works were incorporated, bringing the total to 18 selected works. Among the selected works, similarities emerge in their operational methodologies, despite variances in their detection approaches. We identified commonalities based on the detected manipulations, denoted as Det. Man. (D for duplication, S for splicing, and R for retouching with a slash used to distinguish them where applicable), their treatment of color space, techniques employed for feature extraction to extract pertinent image details, and the detection methods utilized to identify image manipulation. These shared characteristics are meticulously detailed in Table 1, which serves as a comprehensive overview of their collective attributes and methodologies.

Table 1. Comparative analysis of researched works.

ID	Det. Man.	Color Space	Feature Extraction	Detection	Dataset	References
1	D/S	YCbCr, Y values used	Blocks with DCT using doubly stochastic model	Classifiers of type: SVM and ELM	CASIA(V1,V2)	[19]
2	D	Grayscale	Keypoints by proposed method and local simetry plus LPT	Correlation of characteristics by: angle and distance	MICC-F220, MICC-F600, CMH	[5]
3	D	RGB	Blocks by LIOP and DT	Correlation of characteristics by: double g2NN	IMD, MICC-F600	[6]
4	D	YCbCr, Y values used	Blocks by SWT and DCT	Correlation of characteristics by: distance and threshold value	CoMoFoD, UCID	[20]
5	D/S	YCbCr, Cr values used	Signal Decomposition by HHT	Classifiers of type: SVM, KNN and ANN	CASIA(V1,V2), MICC-F2000, MICC-F600, MICC-F220, CoMoFoD, Proprietary	[21]
6	D	Grayscale	Keypoints 2D DWT and SIFT	Correlation of characteristics by: proposed method	CoMoFoD, MICC-F	[22]
7	D	RGB	Blocks by histogram HSV and color moments	Correlation of characteristics by: threshold value	MICC-F220, MICC-F2000, MICC-F8multi	[23]
8	D	RGB	Blocks by 2D DWT and SIFT	Correlation of characteristics by: threshold value	Proprietary	[17]
9	D	Grayscale	Blocks by DWT	Correlation of characteristics by: threshold value	Proprietary	[24]
10	D/S	RGB	none	FPN analysis	IMD, Proprietary	[25]
11	D	Grayscale	Keypoints by Harris Corner Detector and BRISK	Correlation of characteristics by: Hamming Distance and Neared Neighbot Distance Ratio	CoMoFoD, MICC-F220	[26]
12	S/R	RGB	Proposed by authors based on SRSC	Classifiers of type: FLD, LibSVM and ensemble classifier	Proprietary	[27]
13	D	RGB	Blocks by FWHT	Correlation of characteristics by: threshold value	CoMoFoD	[15]
14	D/S	Grayscale	Proposed by the authors	Classifiers of type: SVM with RBF kernels	Columbia	[28]
15	D	RGB	Blocks by QDCT	Classifiers of type: SVM with RBF kernels	Proprietary	[16]
16	D/S/R	RGB	Automatic	Machine learning on FPN data	IFS-TC, RTD	[29]
17	D/S/R	Grayscale	Bilateral Filters and DWT	Feature selection	Proprietary	[18]
18	D/S/R	RGB	Atrous spatial pyramid pooling	Machine learning on FPN data	CASIA(V1,V2), Nim.16, Korus, Coverage, DSO-1, IFC, FaceSwap, Nim.16, Nim.17dev2, MFC18dev1	[30]

The detection methods found can be categorized as classifiers capable of determining alterations in JPEG artifacts, classifiers to detect duplicated areas, and methods that compare detected features to determine correlation or fixed pattern noise (FPN).

To understand methods based on fixed pattern noise, it is first necessary to understand the image capture process. Image acquisition aims to transform an image into a discrete and numerical representation that a computer can store and process. This approach requires a sensor capable of capturing a range of energy from the electromagnetic spectrum and generating as output an electric signal proportional to the captured energy level; then, a digitizer must convert the analog signal into digital information that can be represented in binary form.

The manufacturing process of most sensors responsible for capturing images in digital cameras introduces imperfections that cause minor differences in light sensitivity [31]. The divergences of all the sensors present in a camera introduce a variation in the values of the pixels of images registered by these cameras, resulting in unevenness similar to a signature in all the images it generates [32].

The FPN of a sensor is constant; however, it varies from sensor to sensor. In Sensors of the Charged-Coupled Device (CCD) type, the FPN varies randomly, while in sensors of the Complementary Metal Oxide Semiconductor (CMOS) type, due to its perpendicular capture system, the FPN forms vertical bars, as can be seen in Figure 1 taken from [33].

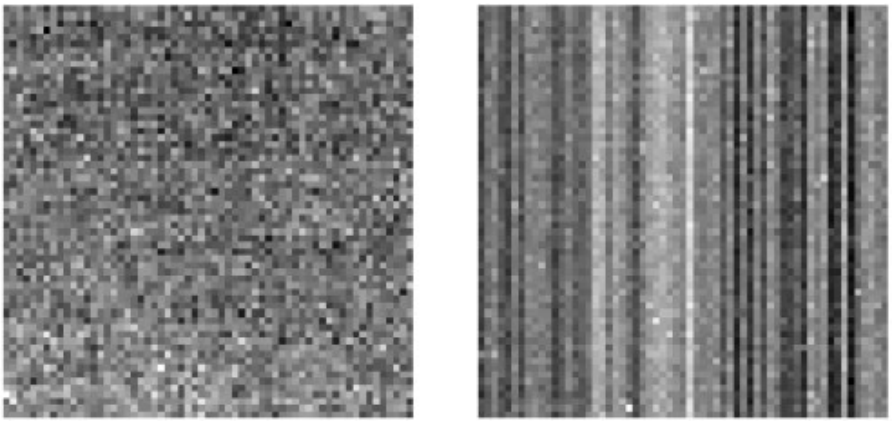

Figure 1. Comparison of FPN present in CCD and CMOS sensors. (**left**) FPN of a CCD sensor (**right**) FPN pf a CMOS sensor.

The work by [25] assumes that the FPN information of the camera used for capture is previously known and then calculates the FPN of the image for validation by comparing the two and marking significantly different areas as being possibly altered.

On the other hand, methods based on classifiers use filtered features to determine if there is a correlation between parts of the image or alteration of the compression through machine learning [34]. In deep learning [35–37], convolutional neural networks (CNNs) are increasingly being used for image classification [38–40].

In this field, several authors are working to reduce the complexity of the classification step, considering the use of big data [41–43] and making this evaluation more efficient [44–46]. Finally, correlation-based methods compare features by similarity to determine whether they contain Duplication, often using a final step to eliminate a portion of the found correlations. In this context, unsupervised learning methods are also applied [47].

All datasets in our literature review focus on detecting Duplication and Splicing manipulations. Their most significant divergences are the images used for alteration, the size of the altered area, and the application of subsequent modifications to hide falsification on the manipulated areas. The work by [48], which is the only one focused on detecting manipulation retouching, had to algorithmically generate its dataset. The datasets used by analyzed works focused only on the detection of Duplication type are MICC-F220, MICC-F600, MICC-F, MICC-F2000, MICC-F8multi, IMD, CMH, CoMoFoD, and UCID.

The datasets encompassing Duplication and Splicing manipulations consist of CASIA v1.0, CASIA v2.0, Columbia, and the Image Manipulation Database. CASIA v1.0 and v2.0

feature manually tampered images with splicing and duplication manually introduced. In contrast, the Columbia and Image Manipulation Dataset are generated by algorithmically by editing patches of an image. The Columbia dataset exclusively contains splicing changes, while the Image Manipulation Database includes both duplication and splicing. Both datasets feature different alterations classified as retouching, aimed at camouflaging other modifications.

The absence of a standardized dataset across these investigations poses challenges in comparing the effectiveness of methods solely based on reported accuracy metrics. Additionally, the reliability of results obtained on datasets algorithmically generated by manipulating image patches may introduce unwanted biases and not accurately reflect the expected accuracy in real-world scenarios. Furthermore, the limited range of manipulations detected by these methods, along with other specific limitations such as the requirement for compression differences in images (e.g., [19,21,27,28]), or the necessity for information about the FPN of capture devices used (e.g., [25,29,30]), further restricts their applicability. Moreover, the unavailability of some datasets, as they were created by the authors and are not publicly accessible, adds another layer of complexity to the evaluation process.

To tackle these challenges, this study employs a meticulously curated dataset comprising four publicly available datasets featuring images manually tampered by humans. Importantly, this dataset does not impose restrictions on the types of manipulations, aiming to closely mimic real-life scenarios and facilitate generalization across a broader spectrum of manipulations. Furthermore, the multi-stream CNN approach enables the utilization of knowledge acquired from traditional passive methods. This approach allows for the extraction of data streams that may contain pertinent information for the model's analysis.

3. Materials and Methods

In this section, we present the methodology adopted for image manipulation detection, which integrates Error-Level Analysis (ELA) and Discrete Wavelet Transform (DWT) alongside a novel multi-stream neural network architecture.

Our decision to adopt a multi-stream neural network architecture was driven by the recognition that traditional passive methods could offer valuable insights to guide the learning process of a Convolutional Neural Network (CNN) for image manipulation detection. Rather than aiming to identify the optimal method outright, we viewed this choice as an initial exploration of the potential capabilities such an architecture may offer. In line with this perspective, we selected two streams based on methodologies outlined in previous literature, with the intention of extracting different facets of image data. This approach was motivated by our desire to integrate diverse aspects of passive image analysis, serving as a foundational step in our investigation into the effectiveness of multi-stream architectures for detecting image manipulation.

The multi-stream architecture comprises three distinct CNNs, each operating on a unique data stream extracted from the original image. Two of these streams were selected based on methodologies outlined in prior literature, which will be explained in detail in the subsequent sections. Each stream is designed to analyze specific data subsets, while the third stream processes the unaltered image itself. By adopting this multi-stream framework, our aim is to leverage the strengths of traditional passive methods and integrate them into a unified detection system.

3.1. Error-Level Analysis

Error-Level Analysis is a passive detection method traditionally used by human forensics specialists to make differences in format artifacts of jpeg images more evident, it works by taking the difference between a jpeg image at different quality levels, making any difference in compression rate more evident on the resulting image [48], an example of this process can be seen in the Figure 2 and the code used to generate the images is presented in Listing 1.

Listing 1. Error-Level analysis implementation in pseudocode.

```
# Static method that performs Error Level Analysis (ELA) on an image using
    JPEG compression.
# Returns a normalized difference image between the original image and a JPEG
    -compressed version of the image.
FUNCTION method_1_ela(image, quality)
    # Create another image with jpeg compression of the given quality
    save_image_as_jpeg(image, temp_image, quality)
    compressed_image = open_image(temp_image)

    # Calculate the image difference between the original and the JPEG-
    compressed image
    difference_image = image - compressed_image

    # Normalize the difference image for contrast by assigning a value of 255
     to the brightest points, while proportionally adjusting the values of
    all other points based on their distance from the brightest point.
    normalized_difference  = difference_image.normalizeContrast()
    RETURN normalized_difference
END FUNCTION
```

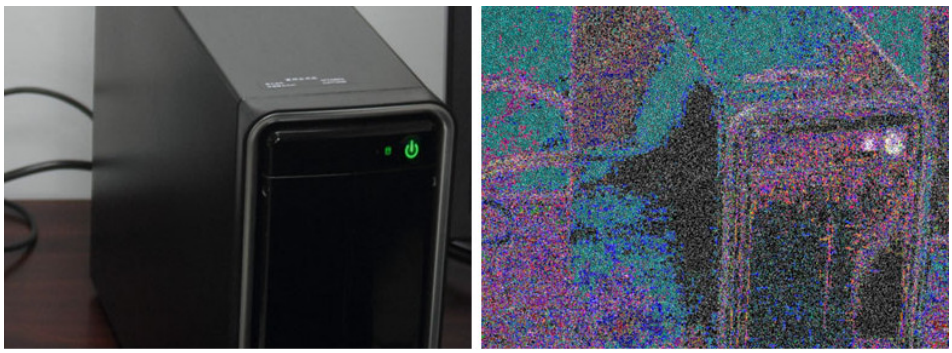

Figure 2. Example of ELA. (**Left**), image with power button manipulated. (**Right**) Resulting image after ELA, usually mostly black, except for the two manipulated areas that have differences in compression.

In this example, the resulting image on the bottom is mostly black, but edited areas have more color in them, however, ELA results aren't always so easily interpreted and traditionally need a forensics specialist to look at the results, however, this paper proposed using ELA as a feature extraction step and feeding it to a Convolutional Neural Network in order to perform the authenticity analysis of an image automatically.

3.2. Discrete Wavelet Transform

DWT stands for denoising [49], which is a mathematical technique used for analyzing signals that can be applied to images [50]. It is a way of decomposing an image into a set of frequency components, with each component representing a different level of detail or resolution [51].

In image manipulation detection, DWT is often used for feature selection as it allows for efficient compression of image data while preserving important image features. The DWT algorithm works by dividing an image into four smaller blocks or "sub-bands" of different frequencies: The LL (low-low) sub-band, which contains the low-frequency information, and the LH (low-high), HL (high-low), and HH (high-high) sub-bands, which contain the high-frequency information [52].

The equations to perform DWT can be found in Equations (1) and (2) and the equation to reverse the process known as inverse DWT is presented in Equation (3).

$$W_\varphi(j_0,k) = \frac{1}{\sqrt{M}} \sum_M f(x)\varphi_{j_0,k}(x) \tag{1}$$

$$W_\psi(j,k) = \frac{1}{\sqrt{M}} \sum_k f(x)\psi_{j,k}(x) \tag{2}$$

$$\begin{aligned} f(x) = & \frac{1}{\sqrt{M}} \sum_k W_\varphi(j_0,k)\varphi_{j_0,k}(x) \\ & + \frac{1}{\sqrt{M}} \sum_{i=i_0}^{\infty} \sum_k W_\psi(j,k)\psi_{j,k}(x). \end{aligned} \tag{3}$$

This work uses the technique proposed by [18], referred to in the rest of this work as DWT method for abbreviation purposes, which makes use of DWT, Bilateral Filters, and the Laplace operator to remove less meaningful features of an image, resulting in an image only with features containing sharp pixel variation, which is a common indicator of forgery.

This technique works by initially the image is converted to grayscale, then DWT is applied to decompose the image information into sub-bands, then the image is reconstructed after discarding the LL band and a bilateral filter, and the Laplace operator is applied, the result is an image where sharp pixel transitions are more easily visible, however when filters are used to mask the manipulation this method fails to highlight the manipulated areas. This work incorporates the image resulting from this method as one of the streams in a Convolutional Neural Network, utilizing it for feature selection to enhance the model's ability to detect large pixel transitions.

An example of the result of this process can be seen in Figure 3 that was taken from the RTD and the code used to generate the images is presented in Listing 2.

Listing 2. DWT based method implementation in pseudocode.

```
FUNCTION method_2_dwt(image)
    # Convert image to grayscale and perform discrete wavelet transform
    gray_image = convertToGrayscale(image)
    coeffs = discreteWaveletTransform(gray_image)
    (LL, (LH, HL, HH)) = coeffs

    # Reconstruct the image using only the high-frequency components
    high_freq_components = (None, (LH, HL, HH))
    joinedLhHlHh = inverseDiscreteWaveletTransform(high_freq_components)

    # Apply bilateral filter to smooth the image while preserving edges
    blurred = bilateralFilter(joinedLhHlHh, 9, 75, 75)

    # Apply Laplacian edge detection to highlight edges
    kernel_size = 3
    imgLapacian = laplacianEdgeDetection(blurred, kernel_size)

    # Convert negative values to zero
    final_image = convertScaleToAbs(imgLapacian)

    RETURN final_image
END FUNCTION
```

Figure 3. Example of DWT method. (**Left**), image power button manipulated. (**Right**) Results of the DWT (variations are enhanced).

3.3. Proposed Method

The proposed approach first consists of applying the ELA method and the method proposed by [18], both used as a feature selection step to generate two extra sets of images serving as the distinct streams for the subsequent model architecture.

Subsequently, the original dataset and the two new sets of images are shuffled and utilized for training and evaluation of three distinct CNNs. Following training, the three models are frozen to preserve the acquired knowledge and then merged. Additional learning layers are appended, and the combined model is fine-tuned through training. These steps are visually represented in Figure 4, providing an overview of the system architecture.

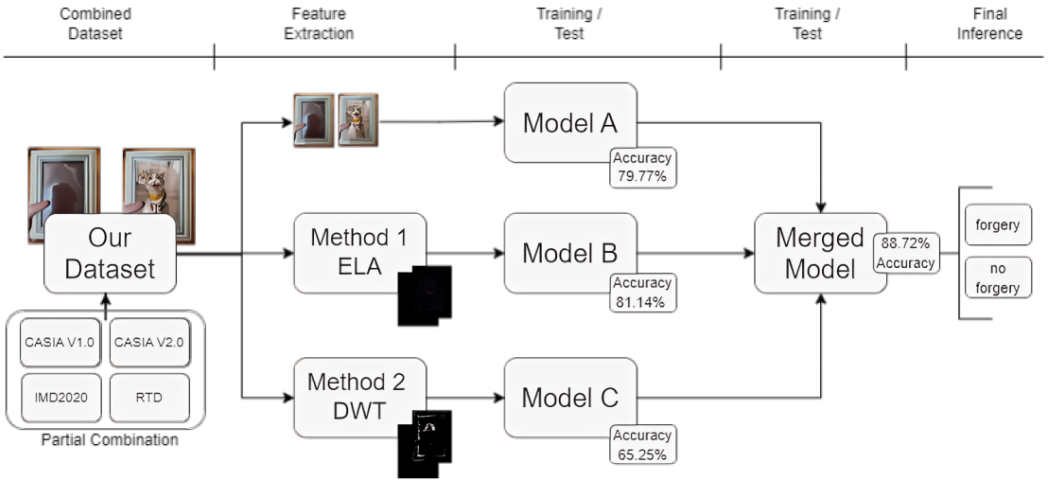

Figure 4. Illustration of the proposed approach.

This approach aims to explore the accuracy of each stream individually in addition to their combination. For this, we individually trained three distinct models and a model composed of all three streams, plus additional learning layers. Stream A uses only the original images without alterations as input, Stream B uses only images with feature selection by ELA, and Stream C uses only images with feature selection by the DWT-based method. Finally, the models are combined and four layers are added to generate the Merged model, which uses images from all three streams as input.

Initially, we considered employing pre-trained models to integrate passive methods into our research. For an initial assessment of their performance compared to custom models, we selected the MobileNetV2 pre-trained model due to its lightweight nature. We conducted preliminary tests, comparing the performance of MobileNetV2 combined with four dense layers, each incorporating l2 regularization and followed by a batch normalization layer, against Model 'A' as detailed later. Results of this comparison are presented in Figures 5 and 6.

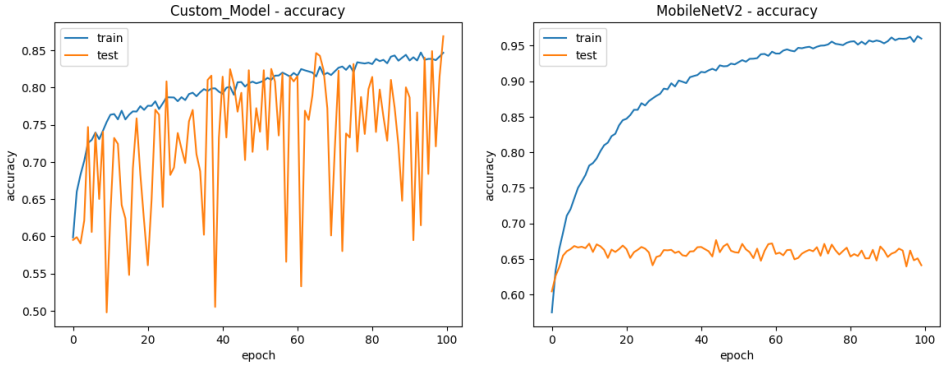

Figure 5. Comparing the accuracy of a custom model to MobileNetV2. (**left**) Custom model accuracy during training (**right**) Pre trained MobileNetV2 accuracy during training.

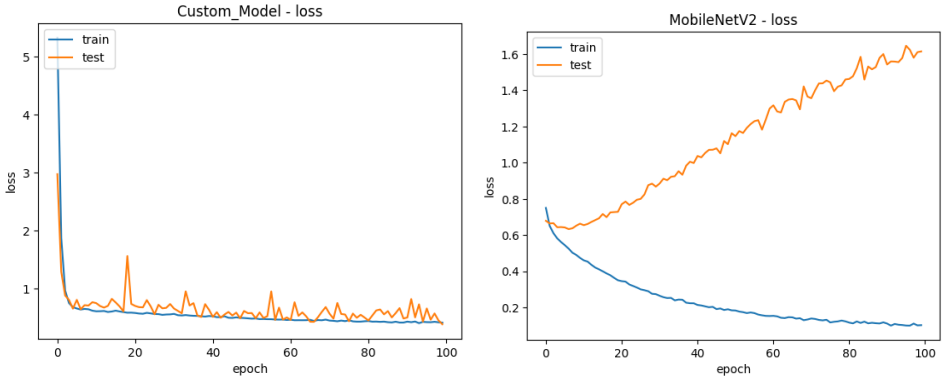

Figure 6. Comparing loss of a custom model to MobileNetV2. (**left**) Custom model loss during training (**right**) Pre trained MobileNetV2 loss during training.

The tests revealed a tendency for pre-trained models to overfit to the training images. Additionally, attempts to mitigate overfitting by unfreezing some of the pre-trained layers yielded similar results. Moreover, employing another pre-trained model, such as InceptionResNetV2, did not demonstrate superior performance compared to the custom model. Consequently, these findings lead us to conclude that pre-trained models may not be well-suited for image manipulation detection.

With that in mind, we decided to create three identical custom CNNs and implement regularization methods to minimize overfitting to produce results capable of generalizing to a more extensive set of images and facilitating the comparison of the three individual models.

The primary challenge in developing the neural networks for this study was addressing the issue of overfitting, where the models tend to perform well on training data but struggle to generalize to new, unseen data. To address this, we conducted a series of experiments aimed at evaluating the impact of various model architectures on accuracy. These experiments involved exploring different configurations such as altering the number and size of convolutional and dense layers, as well as implementing diverse regularization methods. While we did not gather specific data from these experiments, they were instrumental in guiding our selection process and ultimately led to the development of the final model architecture outlined in Figure 7.

Through this exploration, we discovered that simply increasing the number of layers did not yield significant accuracy improvements. On the contrary, it heightened the risk of overfitting. Consequently, we opted to craft a model comprising five convolutional layers, each followed by a batch normalization layer and a max pooling layer. The outputs from these layers were then flattened and channeled through four dense layers, each supported by a batch normalization layer. Additionally, to mitigate overfitting, L2 regularization was applied to both the convolutional and dense layers, as illustrated in Figure 7.

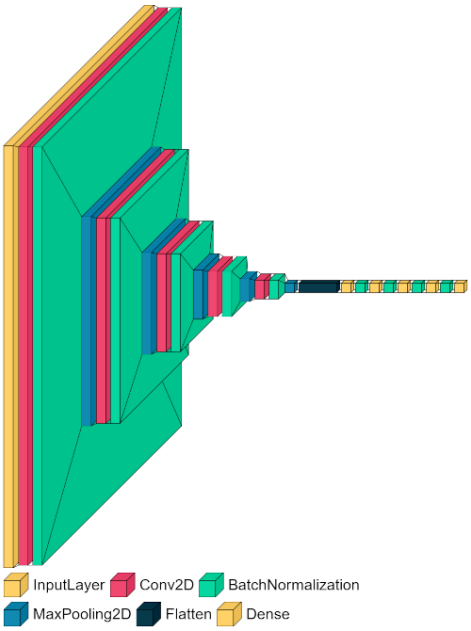

Figure 7. Illustration of the architecture used in the three individual models.

The chosen values for the convolutional layers were set as 16, 32, 64, 128, and 64, each utilizing a 3 by 3 kernel. Similarly, the dense layers were configured with sizes of 128, 128, 64, and 32.

The final proposed merged model is a culmination of individual models A, B, and C. Initially, the last non-output layers of models A and B were combined through a concatenation layer. Subsequently, this combined output was further merged with the last non-output layer of model C through another concatenation layer. To facilitate learning, two dense layers with 64 neurons each were added, accompanied by batch normalization layers. Notably, all layers of the individual models were frozen to preserve the acquired knowledge during training.

While Dropout layers were initially considered, our experimentation revealed that Batch Normalization exhibited superior performance within the same training time, aligning with findings by Singh et al. [53]. The optimizer employed was Adam, with accuracy serving as the primary metric during training. For the Loss function, Binary Cross Entropy was utilized. Rectified Linear Unit activation functions were applied across all layers, with the exception of the output layer, which employed the sigmoid function for classification purposes.

3.4. Dataset Assembly

The initial step in executing our experiments involved compiling the final dataset. To achieve this, we meticulously curated authentic and manipulated images from four distinct datasets. These datasets were chosen for their robust representation of real-life scenarios, as they involved human manipulation of images, with an emphasis on creating manipulations that were challenging to detect. The size of the dataset was limited due to the low availability of humanly manipulated images, which require significant time and effort to create. Additionally, biases may have been introduced due to the skill set of individuals selected to manipulate the images. Nevertheless, efforts were made to ensure that biases were minimized by selecting datasets that inherently did not impose restrictions on the types of manipulations that could be performed.

- CASIA V2.0: proposed in [54], contains 7491 authentic images and 5123 manipulated images containing Splicing and or Duplication operations with retouching operations applied on top to mask alterations;

- Realistic Tampering Dataset: Proposed by [55,56] containing 220 authentic and 220 Splicing and or Duplication manipulations made to the original images with the objective of being realistic. Retouching operations are sometimes applied to help hide Compositing and Duplication manipulations. In addition, this dataset provides masks of tampered areas and information about capture devices used;
- IMD2020: Proposed by [57], it consists of four parts, first a dataset containing 80 authentic images manipulated to generate 1930 images tampered realistically and using all types of manipulation, with their respective manipulation masks. Then the second part consists of 35,000 authentic images captured by 2322 different camera models, the images were collected online and reviewed manually by the authors. The third has 35,000 algorithmically generated images with retouching manipulations. Finally, the last part has 2759 authentic images acquired by the authors with 19 different camera models designed for sensor noise analysis;
- CASIA V1.0: Proposed in [54], Contains 800 authentic images, 459 Duplicate-type manipulation images, and 462 Splicing images. This dataset has no retouching operations applied.

By not imposing limitations on the number of manipulations used, certain types of manipulations may be overrepresented in the dataset. Furthermore, only one of the selected datasets provides information on the types of manipulation performed on each image, leaving uncertainty regarding the exact biases present. However, the hope is that the proportion of manipulations in the dataset mirrors that of real-world scenarios.

To balance the dataset and incorporate realistically manipulated images, we divided the images into two folders. The first folder consists of 7491 authentic images from the CASIA V2.0 dataset. The second folder contains 7491 manipulated images sourced from the realistic images part of the IMD2020 dataset, Realistic Tampering Dataset (RTD), CASIA V2.0, and an additional 218 images from CASIA V1.0, in order to make number of manipulated and pristine images equal. However, it remains uncertain whether there is a 50% incidence of manipulation in real-world applications, potentially introducing classifier bias. Table 2 illustrates the distribution of images from each dataset.

Table 2. Images used from each Dataset for to Assemble our Dataset.

	CASIA V1.0	CASIA V2.0	IMD2020	RTD	Total
Authentic	0	7491	0	0	7491
Tampered	218	5123	1930	220	7491

Therefore, as explained in Table 2 the final dataset used in this paper consists of 14,982 images in total, half original and half Manipulated, since some of the images were not supported by model all images were converted to .jpg format.

The second step of the final program is to apply the methods described to the dataset, as both methods generate an image as output. The result is two new sets of images with specific inconsistencies enhanced, using methods from the TensorFlow library whenever applicable.

The dataset was divided into three parts: 70% for training, 20% for validation, and 10% for testing. Each image was resized to 224 by 224 pixels to match the input size required by MobileNetV2, a pre-trained neural network that was initially considered but ultimately not used in favor of a custom model due to improved performance.

The experiment consisted of first creating two additional sets of images using the two selected passive methods. Subsequently, three convolutional neural networks were trained: Model A using the original dataset, and Models B and C utilizing the outputs of the selected passive methods. These models were then combined by concatenating them at the penultimate layer. Additionally, an extra dense layer followed by a batch regularization layer was introduced so the model could learn how to combine the results, this combination is the final merged model.

To enhance accuracy and mitigate overfitting, three techniques were employed. First, a model checkpoint callback was implemented, which saved the model weights corresponding to the minimum validation loss achieved during training. Second, an early stopping condition was defined, terminating training after 50 epochs without any improvement in validation loss.

The third technique used to reduce overfitting was the use of dataset augmentation techniques applied at the end of every epoch to the training images, which randomly flipped the image in the four-axis and changed the brightness, contrast, saturation, and hue of the images randomly.

The experiments were conducted on a system with the following specifications: Windows 11 operating system, Intel(R) Core(TM) i5-8300H CPU 2.30GHz, 16GB RAM, GeForce GTX 1050 graphics card, Python version 3.10, CUDA version 11.2, cuDNN version 8.1.1, and TensorFlow version 2.10.0. The training was performed with a batch size of 32 over a total of 500 epochs, as the dataset exhibited significant variation in manipulations, leading to fluctuation in the loss of validation images.

4. Results and Discussion

After training, the final Accuracy obtained by the merged model was 89.59% in the set of test images, higher than the model trained just with original images, which obtained 78.64%.

The Figures 8–11 illustrate the area under the curve (AUC) of the Receiver Operating Characteristic (ROC) for the models, assessed across both validation and test datasets, highlighting their performance capabilities. Complementarily, Figures 12–15 present detailed graphics depicting the accuracy and loss metrics during the training phase for models A, B, C and the specifically proposed merged model, showcasing each model's progression and comparative effectiveness throughout the training process.

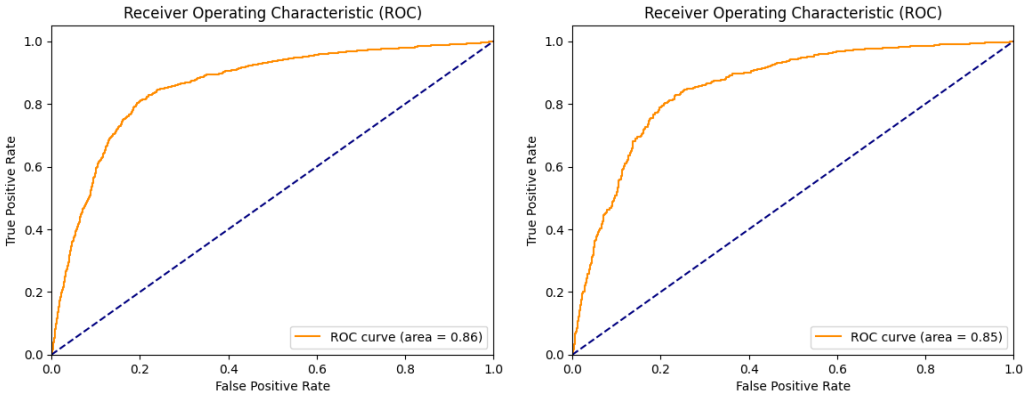

Figure 8. Model A area under curve (ROC) for the training and test datasets. (**left**) Results obtained on the validation dataset. (**right**) Results obtained on the test dataset.

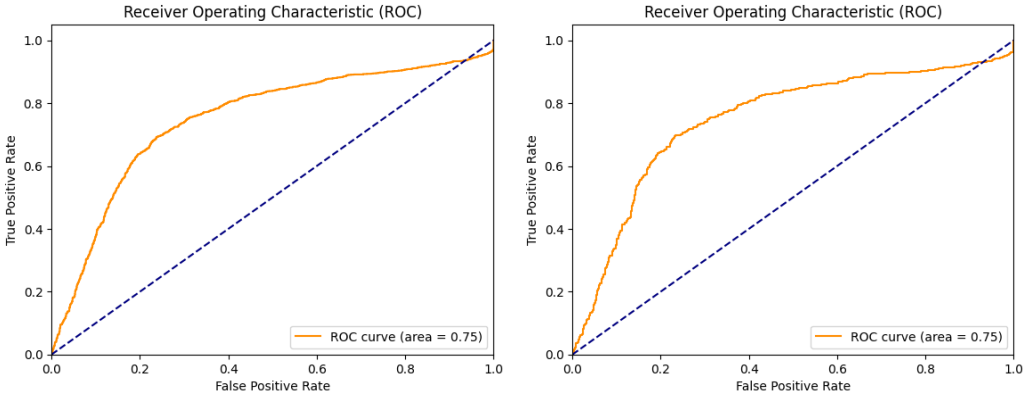

Figure 9. Model B area under curve (ROC) for the training and test datasets. (**left**) Results obtained on the validation dataset. (**right**) Results obtained on the test dataset.

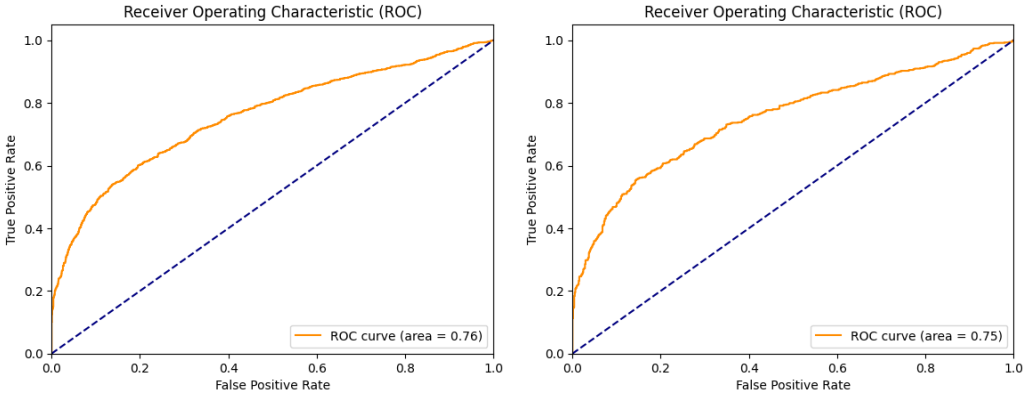

Figure 10. Model C area under curve (ROC) for the training and test datasets. (**left**) Results obtained on the validation dataset. (**right**) Results obtained on the test dataset.

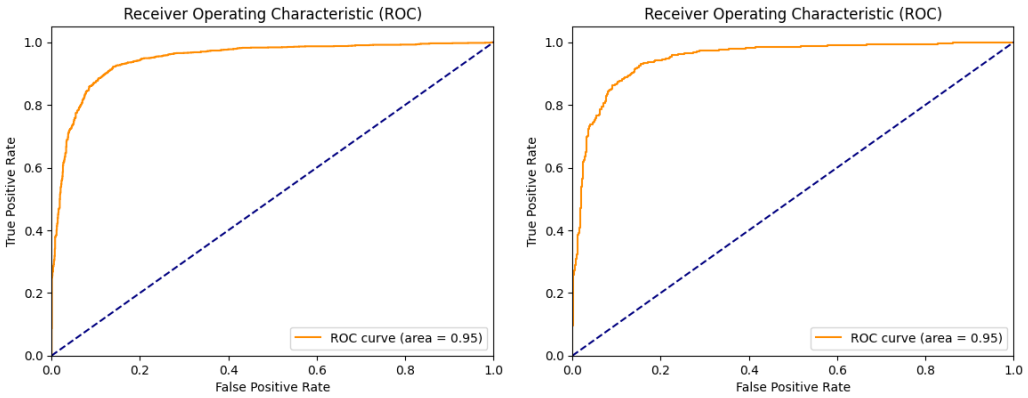

Figure 11. Proposed Model area under curve (ROC) for the training and test datasets. (**left**) Results obtained on the validation dataset. (**right**) Results obtained on the test dataset.

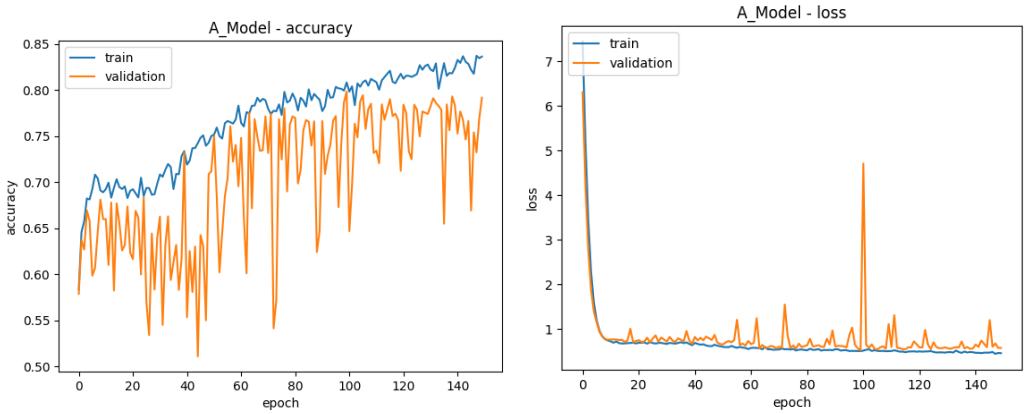

Figure 12. Model A Training Data, the lines in blue refer to results on the 80% of images used for training while those in orange refer to the 20% of images dedicated to validation. The X-axis shows the current training generation. (**left**) Y-axis shows Accuracy during model A training (**right**) Y-axis shows loss during model A training.

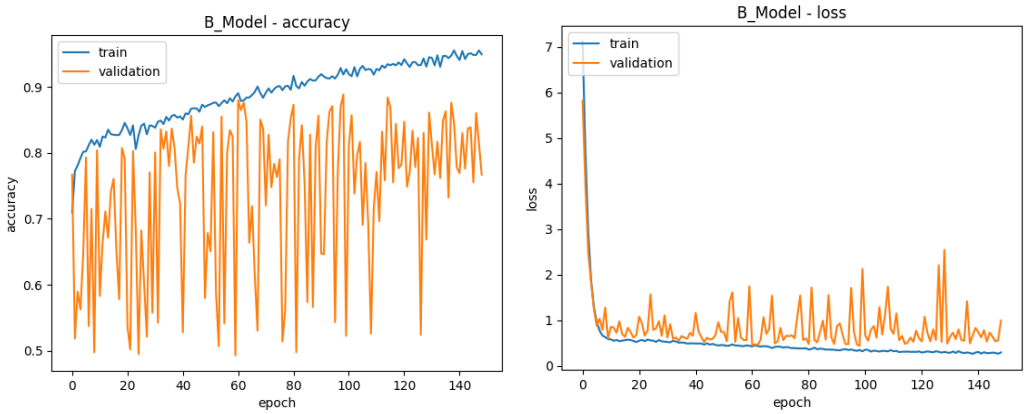

Figure 13. Model B Training Data, the lines in blue refer to results on the 80% of images used for training while those in orange refer to the 20% of images dedicated to validation. The X-axis shows the current training generation. (**left**) Y-axis shows Accuracy during model B training (**right**) Y-axis shows loss during model B training.

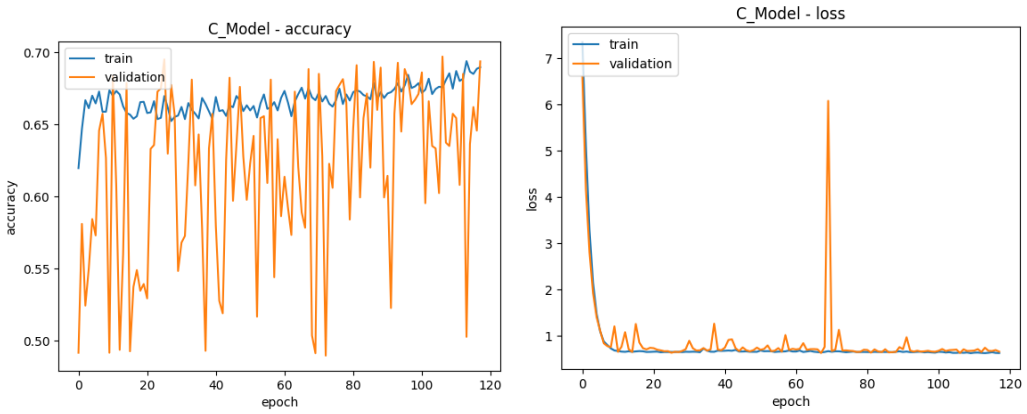

Figure 14. Model C Training Data, the lines in blue refer to results on the 80% of images used for training while those in orange refer to the 20% of images dedicated to validation. The X-axis shows the current training generation. (**left**) Y-axis shows Accuracy during model C training (**right**) Y-axis shows loss during model C training.

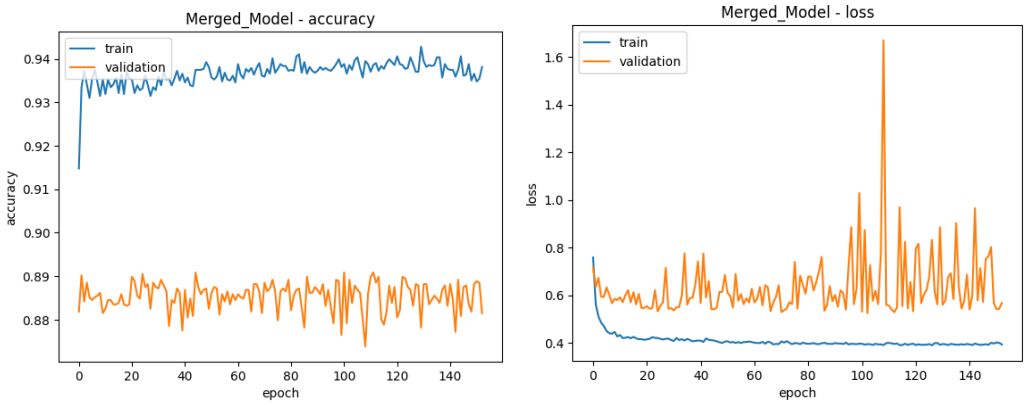

Figure 15. Proposed Model Training Data, the lines in blue refer to results on the 80% of images used for training while those in orange refer to the 20% of images dedicated to validation. The X-axis shows the current training generation. (**left**) Y-axis shows Accuracy during the proposed training (**right**) Y-axis shows loss during the proposed model training.

The culmination of these analyses is encapsulated in Table 3, which details the final results achieved after activating the predetermined stop conditions, offering a comprehensive overview of model efficacious and operational efficiencies.

Table 3. Performance Metrics on Lowest Validation Loss Epoch and Total Training Epochs.

	Model A	**Model B**	**Model C**	**Merged Model**
Training Accuracy	80.73%	91.85%	71.06%	93.85%
Validation Accuracy	78.43%	88.81%	68.85%	88.91%
Test Accuracy	78.64%	68.02%	50.70%	89.59%
Test ROC	0.87	0.76	0.75	0.96
Total Epochs	125	148	117	152
Best Epoch	75	98	67	102

5. Conclusions and Future Work

This study provides a comprehensive review of passive forensic methods designed to detect manipulated images, emphasizing their underlying principles and efficacy. Building upon this foundation, we employ a novel approach that integrates two distinct detection methods within a multi-stream convolutional neural network (CNN) framework. This innovative methodology allows us to harness the strengths of each individual method while mitigating their respective limitations, thereby enhancing overall detection accuracy. Moreover, our analysis is conducted on a meticulously curated dataset comprising a diverse range of image manipulations, ensuring the robustness and generalizability of our findings. Through this combined effort, we aim to advance the field of image manipulation detection by offering a unified and effective solution capable of addressing real-world challenges.

After training, our novel merged method demonstrated a remarkable accuracy of 89.59% on validation images, showcasing the advantages of our multi-stream CNN approach compared to individual streams. In contrast to models utilizing singular detection methodologies, such as Error Level Analysis (ELA) and the method proposed by [18], which were employed as separate streams in our multi-stream model, our merged model showcased superior performance. This comparison highlights the advantage of analyzing multiple streams concurrently, showcasing the synergistic effects of integrating diverse detection methods within a unified framework. Additionally, the individual streams achieved the following accuracies: the model trained solely on original images obtained an accuracy of 78.64%, while the ELA-based model achieved 68.02%, and the model utilizing the method proposed by [18] achieved 50.70%.

The amalgamation of differently trained models on the same image dataset highlights the potential of leveraging various passive detection techniques to improve overall per-

formance. This concept is validated by the superior accuracy achieved by our final model compared to its constituent models, indicating the efficacy of using insight of passive methods to select streams for CNN analysis in a multi-stream model.

The study's limitations primarily revolve around the size and scope of the dataset, coupled with the lack of comprehensive knowledge regarding the manipulations performed on the images. The absence of detailed documentation regarding the specific manipulations applied to each image impedes a thorough analysis of detection performance across different manipulation types. Furthermore, the study lacks a thorough statistical analysis and discussion of false positives/negatives. While the study demonstrates the potential for multi-stream CNN architectures, the absence of localization of manipulated areas represents a further avenue for exploration. Additionally, the study did not thoroughly explore the selection of optimal streams based on passive methods.

However, the findings hold broader implications, highlighting the potential of multi-stream CNN architectures in image manipulation detection. By integrating diverse passive detection methods, this approach not only enhances detection accuracy but also underscores the importance of combining complementary techniques for robust detection. Moreover, the study emphasizes the necessity for standardized datasets and rigorous evaluation metrics in the field of image forensics. These insights can inform future research aimed at developing more effective and reliable methods for detecting manipulated images, thereby enhancing the integrity and trustworthiness of digital media.

This research lays the groundwork for several avenues of expansion and real-world application. Firstly, further exploration could focus on enhancing the dataset used, incorporating more diverse and meticulously documented manipulations to improve the model's robustness and generalization capabilities. Additionally, future studies could delve into developing methods for localizing manipulated areas within images, which would significantly enhance the practical utility of image manipulation detection systems.

Moreover, the multi-stream CNN architecture demonstrated in this research presents a promising framework for integrating various detection techniques. Expanding upon this, researchers could explore the selection of optimal streams based on passive methods, thereby refining the model's ability to detect a wide range of manipulation types with greater accuracy.

In real-world scenarios, the findings of this research could be applied in various fields, including digital forensics, media authentication, and content moderation on social media platforms. By deploying robust image manipulation detection systems developed from this research, organizations and individuals can better safeguard against the dissemination of misleading or falsified visual content, thereby upholding the integrity and trustworthiness of digital media.

Author Contributions: Writing—original draft, Methodology, Software, Investigation and Formal analysis; A.L.A.; Writing—original draft, Methodology, Formal analysis and Project administration; M.D.L.; Writing—review & editing, Validation and Visualization; J.C.S.d.A.; Writing—review & editing, Validation and Visualization; J.F.D.P.S.; Supervision, Funding acquisition, and Resources; Fernandes, A.M.d.R.F.; Supervision, Funding acquisition, and Resources; V.R.Q.L. All authors have read and agreed to the published version of the manuscript.

Funding: The authors would like to acknowledge the support provided by the Fundação de Amparo à Pesquisa e Inovação do Estado de Santa Catarina (Fapesc) under the project FAPESC Nº 29/2021-Programa Estruturante Acadêmico-Apoio à Infraestrutura de Laboratórios Acadêmicos do Estado de Santa Catarina. This support was partial and significantly contributed to the success of this research. The authors would like to acknowledge the Portuguese FCT program, Center of Technology and Systems (CTS) UIDB/00066/2020/UIDP/00066/2020, for partially funding the research. Finally, the authors would like to acknowledge the CAPES Finance Code 001. Also, partial funding for this research was provided by the CEREIA Project (# 2020/09706-7) São Paulo Research Foundation (FAPESP), FAPESP–MCTIC-CGI.BR in partnership with Hapvida NotreDame Intermedica Group.

Data Availability Statement: The data can be shared up on request. The data are not publicly available due to the Brazilian Law (LGPD).

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

Abbreviations

The following abbreviations are used in this manuscript:

CID	Content identifier
CNN	Convolutional neural network
COV	Computer vision
DEL	Deep learning
DIF	Digital image forensics
DWT	Discrete Wavelet Transform
IMP	Image processing
FP	False positive
FN	False negative
HCI	Human-computer interaction
MLP	Multilayer perceptron
R-CNN	Region-based convolutional neural networks
RI	Region of interest
RPN	Region proposal network
TC	Totally connected
TF	True negative
TP	True positive

References

1. Lazer, D.M.J.; Baum, M.A.; Benkler, Y.; Berinsky, A.J.; Greenhill, K.M.; Menczer, F.; Metzger, M.J.; Nyhan, B.; Pennycook, G.; Rothschild, D.; et al. The science of fake news. *Science* **2018**, *359*, 1094–1096. [CrossRef]
2. De Domenico, M.; Lima, A.; Mougel, P.; Musolesi, M. The Anatomy of a Scientific Rumor. *Sci. Rep.* **2013**, *3*, 2980. [CrossRef]
3. Nash, R.A.; Wade, K.A.; Lindsay, D.S. Digitally manipulating memory: Effects of doctored videos and imagination in distorting beliefs and memories. *Mem. Cogn.* **2009**, *37*, 414–424. [CrossRef]
4. López-Cantos, F. The Impact on Public Trust of Image Manipulation in Science. *Informing Sci. Int. J. Emerg. Transdiscipl.* **2019**, *22*, 45–53. [CrossRef] [PubMed]
5. Vaishnavi, D.; Subashini, T. Application of local invariant symmetry features to detect and localize image copy move forgeries. *J. Inf. Secur. Appl.* **2019**, *44*, 23–31. [CrossRef]
6. Lyu, Q.; Luo, J.; Liu, K.; Yin, X.; Liu, J.; Lu, W. Copy Move Forgery Detection based on double matching. *J. Vis. Commun. Image Represent.* **2021**, *76*, 103057. [CrossRef]
7. Cho, S.H.; Agarwal, S.; Koh, S.J.; Jung, K.H. Image Forensics Using Non-Reducing Convolutional Neural Network for Consecutive Dual Operators. *Appl. Sci.* **2022**, *12*, 7152. [CrossRef]
8. Liu, B.; Pun, C.M. Exposing splicing forgery in realistic scenes using deep fusion network. *Inf. Sci.* **2020**, *526*, 133–150. [CrossRef]
9. Rocha, A.; Scheirer, W.; Boult, T.; Goldenstein, S. Vision of the unseen. *ACM Comput. Surv.* **2011**, *43*, 26. [CrossRef]
10. Sharma, V.; Jha, S. Image Forgery and it's Detection Technique: A Review. *Int. Res. J. Eng. Technol.* **2016**, *3*, 756–762.
11. Qazi, T.; Hayat, K.; Khan, S.U.; Madani, S.A.; Khan, I.A.; Kołodziej, J.; Li, H.; Lin, W.; Yow, K.C.; Xu, C.Z. Survey on blind image forgery detection. *IET Image Process.* **2013**, *7*, 660–670. [CrossRef]
12. Lubna, J.I.; Chowdhury, S.M.A.K. Detecting Fake Image: A Review for Stopping Image Manipulation. In Proceedings of the Advances in Computational Intelligence, Security and Internet of Things, Agartala, India, 13–14 December 2019; pp. 146–159. [CrossRef]
13. Sharma, S.; Ghanekar, U. A hybrid technique to discriminate Natural Images, Computer Generated Graphics Images, Spliced, Copy Move tampered images and Authentic images by using features and ELM classifier. *Optik* **2018**, *172*, 470–483. [CrossRef]
14. Agarwal, R.; Khudaniya, D.; Gupta, A.; Grover, K. Image Forgery Detection and Deep Learning Techniques: A Review. In Proceedings of the 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 13–15 May 2020; pp. 1096–1100. [CrossRef]
15. Soni, B.; Das, P.K.; Thounaojam, D.M. Blur Invariant Block based Copy-Move Forgery Detection Technique using FWHT Features. In Proceedings of the International Conference on Watermarking and Image Processing, Paris, France, 6–8 September 2017. [CrossRef]
16. Li, C.; Ma, Q.; Xiao, L.; Ying, S. An Image Copy Move Forgery Detection Method Using QDCT. In Proceedings of the International Conference on Internet Multimedia Computing and Service, Xi'an, China, 19–21 August 2016. [CrossRef]
17. Sanap, V.K.; Mane, V.M. Region duplication forgery detection in digital images using 2D-DWT and SVD. In Proceedings of the 2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), Davangere, India, 29–31 October 2015; pp. 599–604. [CrossRef]

18. Ravi, K.; Devraj, N.; Shylaja, S.S. A new approach to detect paste forgeries in an image. In Proceedings of the 2017 Fourth International Conference on Image Information Processing (ICIIP), Shimla, India, 21–23 December 2017; pp. 1–6. [CrossRef]
19. Dua, S.; Singh, J.; Parthasarathy, H. Detection and localization of forgery using statistics of DCT and Fourier components. *Signal Process. Image Commun.* **2020**, *82*, 115778. [CrossRef]
20. Mahmood, T.; Mehmood, Z.; Shah, M.; Saba, T. A robust technique for copy-move forgery detection and localization in digital images via stationary wavelet and discrete cosine transform. *J. Vis. Commun. Image Represent.* **2018**, *53*, 202–214. [CrossRef]
21. Kasban, H.; Nassar, S. An efficient approach for forgery detection in digital images using Hilbert–Huang transform. *Appl. Soft Comput.* **2020**, *97*, 106728. [CrossRef]
22. Hashmi, M.F.; Hambarde, A.R.; Keskar, A.G. Copy move forgery detection using DWT and SIFT features. In Proceedings of the 2013 13th International Conference on Intellient Systems Design and Applications, Bangi, Malaysia, 8–10 December 2013; pp. 188–193. [CrossRef]
23. Malviya, A.V.; Ladhake, S.A. Region duplication detection using color histogram and moments in digital image. In Proceedings of the 2016 International Conference on Inventive Computation Technologies, Coimbatore, India, 26–27 August 2016; Volume 1, pp. 1–4. [CrossRef]
24. Khan, S.; Kulkarni, A. Robust method for detection of copy-move forgery in digital images. In Proceedings of the 2010 International Conference on Signal and Image Processing, Chennai, India, 15–17 December 2010.
25. Fahmy, M.F.; Fahmy, O.M. A natural preserving transform based forgery detection scheme. In Proceedings of the 2015 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), Abu Dhabi, United Arab Emirates, 7–10 December 2015; pp. 215–220. [CrossRef]
26. Isaac, M.M.; Wilscy, M. Copy-Move forgery detection based on Harris Corner points and BRISK. In Proceedings of the Third International Symposium onWomen in Computing and Informatics, Kochi, India, 10–13 August 2015. [CrossRef]
27. Liu, Q.; Li, X.; Cooper, P.A.; Hu, X. Shift recompression-based feature mining for detecting content-aware scaled forgery in JPEG images. In Proceedings of the Twelfth International Workshop on Multimedia Data Mining, Beijing, China, 12 August 2012. [CrossRef]
28. Liu, Q.; Sung, A.H. A new approach for JPEG resize and image splicing detection. In Proceedings of the First ACM workshop on Multimedia in Forensics, Beijing, China, 23 October 2009. [CrossRef]
29. Liu, Y.; Guan, Q.; Zhao, X.; Cao, Y. Image Forgery Localization Based on Multi-Scale Convolutional Neural Networks. *arXiv* **2017**, arXiv:1706.07842. https://doi.org/10.48550/ARXIV.1706.07842.
30. Rao, Y.; Ni, J.; Xie, H. Multi-semantic CRF-based attention model for image forgery detection and localization. *Signal Process.* **2021**, *183*, 108051. [CrossRef]
31. Lukáš, J.; Fridrich, J.; Goljan, M. Detecting digital image forgeries using sensor pattern noise. In Proceedings of the Electronic Imaging 2006, San Jose, CA, USA, 15–19 January 2006. [CrossRef]
32. Lin, X.; Li, C.T. PRNU-Based Content Forgery Localization Augmented with Image Segmentation. *IEEE Access* **2020**, *8*, 222645–222659. [CrossRef]
33. Mohammadnejad, S.; Roshani, S.; Sarvi, M. Fixed pattern noise reduction method in CCD sensors for LEO satellite applications. In Proceedings of the 11th International Conference on Telecommunications, Graz, Austria, 15–17 June 2011.
34. Stefenon, S.F.; Corso, M.P.; Nied, A.; Perez, F.L.; Yow, K.C.; Gonzalez, G.V.; Leithardt, V.R.Q. Classification of insulators using neural network based on computer vision. *IET Gener. Transm. Distrib.* **2021**, *16*, 1096–1107. [CrossRef]
35. Corso, M.P.; Stefenon, S.F.; Singh, G.; Matsuo, M.V.; Perez, F.L.; Leithardt, V.R.Q. Evaluation of visible contamination on power grid insulators using convolutional neural networks. *Electr. Eng.* **2023**, *105*, 3881–3894. [CrossRef]
36. Dos Santos, G.H.; Seman, L.O.; Bezerra, E.A.; Leithardt, V.R.Q.; Mendes, A.S.; Stefenon, S.F. Static attitude determination using convolutional neural networks. *Sensors* **2021**, *21*, 6419. [CrossRef]
37. Souza, B.J.; Stefenon, S.F.; Singh, G.; Freire, R.Z. Hybrid-YOLO for classification of insulators defects in transmission lines based on UAV. *Int. J. Electr. Power Energy Syst.* **2023**, *148*, 108982. [CrossRef]
38. Stefenon, S.F.; Yow, K.C.; Nied, A.; Meyer, L.H. Classification of distribution power grid structures using inception v3 deep neural network. *Electr. Eng.* **2022**, *104*, 4557–4569. [CrossRef]
39. Lu, J.; Tan, L.; Jiang, H. Review on convolutional neural network (CNN) applied to plant leaf disease classification. *Agriculture* **2021**, *11*, 707. [CrossRef]
40. Yu, C.; Han, R.; Song, M.; Liu, C.; Chang, C.I. A simplified 2D-3D CNN architecture for hyperspectral image classification based on spatial-spectral fusion. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2020**, *13*, 2485–2501. [CrossRef]
41. Yamasaki, M.; Freire, R.Z.; Seman, L.O.; Stefenon, S.F.; Mariani, V.C.; dos Santos Coelho, L. Optimized hybrid ensemble learning approaches applied to very short-term load forecasting. *Int. J. Electr. Power Energy Syst.* **2024**, *155*, 109579. [CrossRef]
42. Starke, L.; Hoppe, A.F.; Sartori, A.; Stefenon, S.F.; Santana, J.F.D.P.; Leithardt, V.R.Q. Interference recommendation for the pump sizing process in progressive cavity pumps using graph neural networks. *Sci. Rep.* **2023**, *13*, 16884. [CrossRef]
43. Surek, G.A.S.; Seman, L.O.; Stefenon, S.F.; Mariani, V.C.; Coelho, L.S. Video-based human activity recognition using deep learning approaches. *Sensors* **2023**, *23*, 6384. [CrossRef]
44. Glasenapp, L.A.; Hoppe, A.F.; Wisintainer, M.A.; Sartori, A.; Stefenon, S.F. OCR Applied for Identification of Vehicles with Irregular Documentation Using IoT. *Electronics* **2023**, *12*, 1083. [CrossRef]

45. Vieira, J.C.; Sartori, A.; Stefenon, S.F.; Perez, F.L.; de Jesus, G.S.; Leithardt, V.R.Q. Low-Cost CNN for Automatic Violence Recognition on Embedded System. *IEEE Access* **2022**, *10*, 25190–25202. [CrossRef]
46. Corso, M.P.; Perez, F.L.; Stefenon, S.F.; Yow, K.C.; Ovejero, R.G.; Leithardt, V.R.Q. Classification of Contaminated Insulators Using k-Nearest Neighbors Based on Computer Vision. *Computers* **2021**, *10*, 112. [CrossRef]
47. Wilbert, H.J.; Hoppe, A.F.; Sartori, A.; Stefenon, S.F.; Silva, L.A. Recency, Frequency, Monetary Value, Clustering, and Internal and External Indices for Customer Segmentation from Retail Data. *Algorithms* **2023**, *16*, 396. [CrossRef]
48. Gunawan, T.S.; Hanafiah, S.A.M.; Kartiwi, M.; Ismail, N.; Za'bah, N.F.; Nordin, A.N. Development of Photo Forensics Algorithm by Detecting Photoshop Manipulation using Error Level Analysis. *Indones. J. Electr. Eng. Comput. Sci.* **2017**, *7*, 131. [CrossRef]
49. Stefenon, S.F.; Seman, L.O.; Aquino, L.S.; dos Santos Coelho, L. Wavelet-Seq2Seq-LSTM with attention for time series forecasting of level of dams in hydroelectric power plants. *Energy* **2023**, *274*, 127350. [CrossRef]
50. Sopelsa Neto, N.F.; Stefenon, S.F.; Meyer, L.H.; Ovejero, R.G.; Leithardt, V.R.Q. Fault Prediction Based on Leakage Current in Contaminated Insulators Using Enhanced Time Series Forecasting Models. *Sensors* **2022**, *22*, 6121. [CrossRef]
51. Branco, N.W.; Cavalca, M.S.M.; Stefenon, S.F.; Leithardt, V.R.Q. Wavelet LSTM for fault forecasting in electrical power grids. *Sensors* **2022**, *22*, 8323. [CrossRef]
52. Klaar, A.C.R.; Stefenon, S.F.; Seman, L.O.; Mariani, V.C.; Coelho, L.d.S. Optimized EWT-Seq2Seq-LSTM with Attention Mechanism to Insulators Fault Prediction. *Sensors* **2023**, *23*, 3202. [CrossRef]
53. Singh, G.; Stefenon, S.F.; Yow, K.C. Interpretable visual transmission lines inspections using pseudo-prototypical part network. *Mach. Vis. Appl.* **2023**, *34*, 41. [CrossRef]
54. 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. [CrossRef]
55. Korus, P.; Huang, J. Multi-scale Analysis Strategies in PRNU-based Tampering Localization. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 809–824. [CrossRef]
56. Korus, P.; Huang, J. Evaluation of Random Field Models in Multi-modal Unsupervised Tampering Localization. In Proceedings of the IEEE International Workshop on Information Forensics and Security, Abu Dhabi, United Arab Emirates, 4–7 December 2016.
57. Novozámský, A.; Mahdian, B.; Saic, S. IMD2020: A Large-Scale Annotated Dataset Tailored for Detecting Manipulated Images. In Proceedings of the IEEE Winter Applications of Computer Vision Workshops, Snowmass, CO, USA, 1–5 March 2020; pp. 71–80. [CrossRef]