DETECTION AND LOCALIZATION OF FORGERIES IN DIGITAL IMAGES

by

Belal Ahmed
B.Sc., Ain Shams University, 2007
M.Sc., University of Northern British Columbia, 2014

A Dissertation Submitted in Partial Fulfillment of
the Requirements for the degree of

DOCTOR OF PHILOSOPHY

in the Department of Electrical and Computer Engineering

University of Victoria

Supervisory Committee

Dr. T. Aaron Gulliver, supervisor
(Department of Electrical and Computer Engineering)

Dr. Wu-Sheng Lu, department member
(Department of Electrical and Computer Engineering)

Dr. Saif alZahir, outside member
(Department of Computer Science and Engineering)

Abstract

Digital images have become a dominant source of information and means of communication in our society. However, these images can easily be altered using readily available image editing tools. Image tampering can be done in several ways such as image splicing, retouching, and copy-move forgeries. In copy-move forgery, part of an image is copied and pasted into a different part of the same image for the purpose of hiding or adding an object to the image. In image splicing, part of an image is copied and pasted into a different image. To detect image forgeries, image features must first be extracted. A feature is information related to the edges, objects or a specific region in the image.

In this dissertation, new methods for detecting copy-move forgery and image splicing are introduced. Most existing block-based forgery detection methods use large feature dimensions up to 64 per image block so the complexity is high. However, reducing the feature dimensions lowers the detection accuracy, so a new method of detecting copy-move forgery in images using only 4 features per image block. This method uses steerable pyramid and singular value decomposition (SVD) techniques to decompose and extract features from image blocks. Then the features are sorted lexicographically and matched using the Kolmogorov-Smirnov (KS) test. The proposed algorithm is compared to several well-known techniques and shown to provide better accuracy.

To detect image splicing, a new deep learning method is introduced. This method employs Mask-RCNN to generate masks for spliced regions in forged images. It is specifically designed to learn discriminative artifacts from tampered regions. In this method, a ResNet backbone is used to convert the input image into a feature map. The ResNet-50 and ResNet-101 backbones are considered. The ImageNet, He_normal, and Xavier_normal initialization techniques are employed and compared based on convergence. To train a robust model, several post-processing techniques are applied to the input images.

Several techniques have been introduced for image forgery detection. However, most only focus on detecting a certain kind of forgery and perform poorly in other cases. As a result, detecting multiple kinds of forgery using one technique remains a problem. Thus, a novel deep neural architecture called PADNET is introduced which has been specifically designed to detect multiple kinds of forgery. Unlike other solutions, PADNET is an end-to-end trainable deep neural network which employs feature pyramid network (FPN) to aggregate features from multiscale levels of a ResNet-50 backbone. The feature maps are then used to train a DeepUNet architecture designed to learn discriminative features by considering both high-level global features and low-level local features. The convergence of PADNET is tested using two loss functions, binary cross-entropy and weighted binary cross-entropy. Experimental results show that weighted binary cross-entropy is more efficient as a loss function for copy-move forgery while binary cross-entropy is more efficient for image splicing. In addition, the performance of PADNET with training on only the boundaries of the forged area is compared to the network trained on the entire forged area. Evaluation is done using the well known CoMoFoD dataset for copy-move forgery and CASIA1 for image splicing forgery. The results obtained demonstrate that PADNET outperforms state-of-the-art copy-move and image splicing forgery detection algorithms.

Contents

List of Tables

List of Figures

Acknowledgements

In the name of Allah, the most Gracious, the most Merciful. All praise be to Allah the Almighty who has given me knowledge, patience, and perseverance to finish my Ph.D. dissertation.

I am deeply grateful to my parents who stood all the way behind me with their support, encouragement, and prayers until this work was done. The completion of my dissertation would not have been possible without the support and nurturing of my wife Radwa Hammad. She was always there to provide me with her advices and suggestions. I am so grateful to my lovely daughter Malak Ahmed for her patience and encouragement.

My deepest thanks to my supervisor Dr. T. Aaron Gulliver for his invaluable scholarly advice, inspirations, help, and guidance that helped me through my Ph.D. dissertation work. I will always be indebted to him for all he has done for me, and it is a pleasure to acknowledge his guidance and support.

I would like to express my deepest appreciation to Dr. Saif alZahir for his guidance, support, and beneficial discussions. He has provided me with so much help and valuable advice.

I would like to acknowledge the advice and support from Dr. Wu-Sheng Lu for making my dissertation complete and resourceful.

Finally, I would like to thank my friends Ahmed Adel, Shady Elbassiouny, Mohamed Osama, Ramy Hussein, and Hazem Abdelhafez for their generous friendship and enlightening discussions.

Chapter 1

Introduction

The rapid growth in computer technology has made digital images a dominant source of information. Among their many uses, digital images can be used as evidence in courts or by news editors as part of event coverage [1–3]. Images may undergo alterations or modifications such as cropping or color adjustment which may or may not be acceptable. However, images presented as evidence should not be manipulated or else credibility is lost. Tampered images presented as evidence can be misleading resulting in imprisonment of the innocent or freedom of the guilty. In general, image tampering may have significant consequences in politics, economics, and justice. Whether tampering is done for illegal purposes or not, authenticity is necessary. The advent of digital images and the availability of image processing software makes authenticity uncertain. For this reason, digital image forgery detection has become an active research area in recent years.

Image tampering can be done in several ways such as image splicing, retouching, or copy-move forgery [4]. Copy-move forgery is one of the most common techniques used to manipulate images. It can be done by copying part of an image and pasting it into the same image to conceal or change information. Since the copied region is from the same image, some features such as color and noise are consistent with the rest of the image which can make forged regions indistinguishable to the naked eye. Further, post-processing techniques such as brightness change (BC), color reduction (CR), contrast adjustment (CA), and image blurring (IB) can be used to make forgeries harder to detect. Image splicing is another kind of forgery that is made by copying part of an image and pasting it into a different image. Image retouching is used to change the appearance of a subject in an image [5].

Image forgery detection techniques can be categorized into active and passive approaches according to the presence of additional information. In active approaches [6], additional information is embedded in the digital image for tampering detection. Examples of active approaches are digital watermarks and digital signatures. However, prior information about the image is required in order to remove the watermark and retrieve the original image. In contrast, passive approaches do not require prior information about the image [7] and is the focus of this dissertation.

1.1 Passive approaches

Passive approaches only use the received image to detect forgeries and so are called blind forgery detection techniques. In most cases of image forgery, only forged images are available without any prior information about the image. This makes passive approaches the most popular techniques. In these approaches, forgery detection is done by analyzing the content and statistics of the images [4]. One of the early passive approaches used for detecting image forgeries is exhaustive search. In this method, every image pixel is compared to other pixels of the same image to detect similar pixels resulting in a very high time complexity. This section presents a review of existing passive approaches for detecting image forgery.

1.1.1 Keypoint-based methods

Several techniques can be used to extract keypoint features from images such as the scale invariant feature transform (SIFT) and speeded up robust features (SURF) [8]. In these techniques, keypoint features are extracted based on regions with high entropy. Then, these features are used to extract regions with similar features in the image. Any regions with the same features are considered to be duplicates. However, keypoint-based methods have limitations such as not being invariant to geometrical transformations (rotation and scaling) [9]. For this reason, block-based methods are employed in this research.

1.1.2 Block-based methods

Several block-based methods have been introduced to detect image forgery. These methods consist of several stages which are discussed below.

Preprocessing

In block-based methods, an image can be preprocessed before further analysis. Several methods such as those in [10–14] convert an image into grayscale using the luminance formula $L = 0.299R + 0.587G + 0.114B$, where L represents the total luminance and R,G, and B represent the red, green, and blue luminances, respectively, of the colour image. Each color is represented by a vector which provides the brightness intensity, for example 0 to 255 for 8-bit images. The reason behind converting an image into grayscale is to reduce the complexity by changing a 3-dimensional (3D) image (R, G, B) into a 1-dimensional (1D) image. Further, the color information does not contribute to identifying important image features such as edges, contrast, and brightness [15].

Image partitioning

In block-based methods, an image is partitioned into overlapping or non-overlapping blocks. The output blocks can be any shape such as squares [10, 13] or circles [16, 17]. Each block has features which can be extracted. Some methods operate directly on the blocks without feature extraction such as in [18]. In this dissertation, the images are partitioned into square blocks of

size $B \times B$. The total number of blocks is

$$N_{overlapping} = (M - B + 1) \times (N - B + 1) \tag{1.1}$$

for overlapping blocks and for non-overlapping blocks the number of blocks is

$$N_{non-overlapping} = \left(\frac{M}{B}\right) \times \left(\frac{N}{B}\right) \tag{1.2}$$

where M is the number of rows and N is the number of columns. If M is not divisible by B then the image is padded with zeros.

Feature Extraction and Matching Techniques

After image partitioning, features are extracted from each block to find duplicated regions in the image. Techniques such as discrete cosine transform (DCT) and singular vector decomposition (SVD) can be used to extract feature vectors from image blocks.

DCT coefficients can be used to represent an image as a sum of sinusoids of varying magnitudes and frequencies [13]. These coefficients can be used as feature vectors for the image. The earliest block-based approaches were based on DCT coefficients of blocks [19].

SVD is a matrix factorization technique that can be used to extract singular values from an image [20]

$$A_{m \times n} = U_{m \times m} S_{m \times n} V^T_{n \times n} \tag{1.3}$$

where $A_{m \times n}$ is an image of size $m \times n$, $S_{m \times n}$ is a diagonal matrix with singular values along the diagonal, and $U_{m \times m}$ and $V_{n \times n}$ are orthogonal matrices so that

$$U^T U = I_{m \times m} \tag{1.4}$$

$$V^T V = I_{n \times n} \tag{1.5}$$

The extracted singular values can be used as feature vectors. These feature vectors are sorted using techniques such as lexicographic sorting or nearest neighbour. This reduces the complexity of the algorithm as then only blocks with similar features are compared. After sorting, a distance measure is used between feature vectors such as Euclidean distance to detect the matching blocks.

In this dissertation, a block-based method is developed for detecting copy-move forgery in images. SVD is employed to extract feature vectors from the image. These feature vectors are sorted using lexicographic sorting. Then, they are matched using the Kolmogorov-Smirnov (KS) test.

1.2 Deep learning

In the field of artificial intelligence (AI), an artificial neural network (ANN) is used to mimic the functioning of a human brain. A perceptron is one of the earliest ANN architectures [21] and is used to classify data as shown in Figure 1.1. It consists of a single layer of input neurons and one binary output. Single layer perceptrons can only be used for simple linear classification

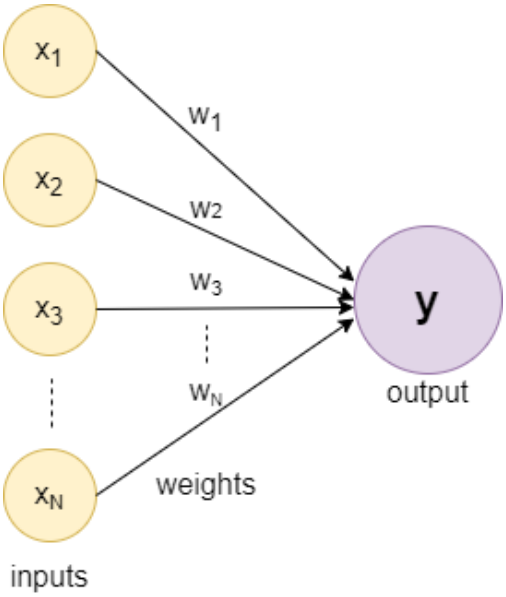


Figure 1.1: A single layer perceptron.

problems [22]. To solve non-linear classification problems, deep learning techniques have been introduced.

Deep learning is learning with ANNs constructed by stacking many layers [23]. It is used to solve complex problems such as objects detection and speech recognition. This is because it can learn without being explicitly programmed and predict or classify based on the learnt data. It consists of an input layer, hidden or intermediate layers, and an output layer as shown in Figure 1.2. Each layer consists of a number of neurons. Each neuron takes inputs, multiplies them by weights, adds bias values, and then passes the result to other neurons. This can be expressed as

$$y = f(\sum_{i=1}^{N} \omega_i x_i + b) \tag{1.6}$$

where x_i is the input value from neuron i, ω_i is the weight applied to input x_i, b is the bias term, y is the output value, N is the number of neurons, and f is an activation function. Input values pass through the network of hidden layers and the results arrive at the output layer. Each hidden layer modifies the data to produce an output. The output layer is the prediction which contains one neuron for a binary classification problem, or several neurons for a multiclass classification problem. Predictions can be improved by using backpropagation. Backpropagation is used in deep neural networks to fine tune the weights based on loss values.

1.2.1 Convolutional neural network

A convolutional neural network (CNN) is the most common deep learning model. It has been applied successfully in image classification and recognition tasks [24] and in areas such as face recognition, natural language processing, and text classification [25]. A CNN is a multistage neural network (NN) with each stage consisting of multiple layers. It derives its name from the convolutional operation. The primary purpose of a CNN is to extract features from the input

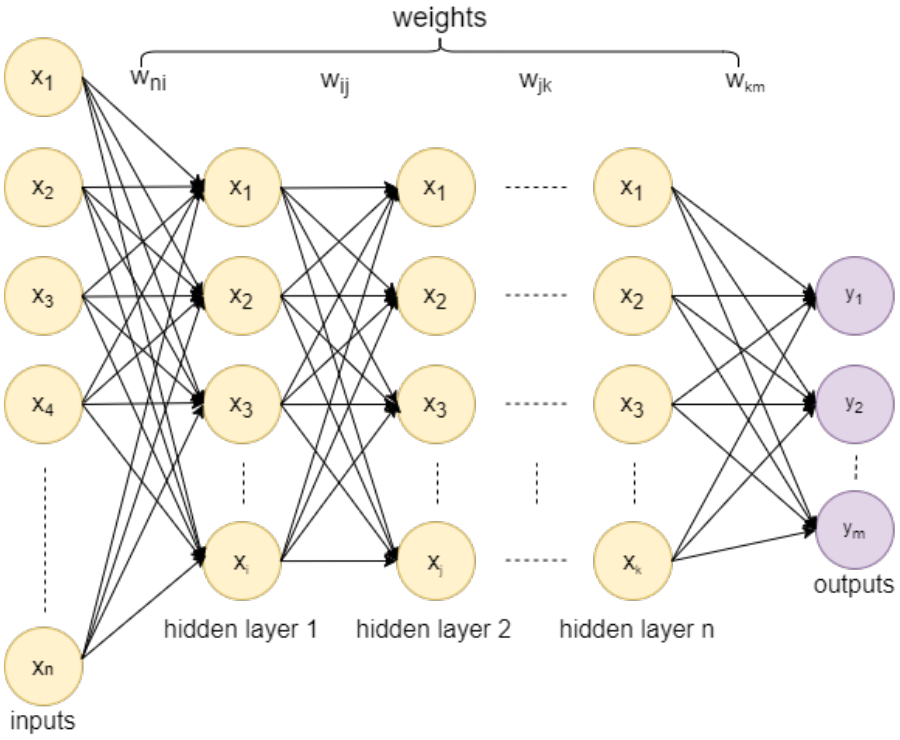


Figure 1.2: A deep neural network.

data. For images, the input layer of the CNN is responsible for converting the input image into a feature map. The feature map is a matrix that contains the extracted features of the image. These features are used to represent characteristics of elements such as edges, colors, shape, or size.

1.2.1.1 Convolutional layer

The main building block of a CNN is the convolutional layer. This layer is made up of kernels which create feature maps based on the corresponding weight and bias values. Each kernel convolves the values of a receptive field as shown in Fig. 1.3. Convolution preserves the spatial relationships between pixels by learning image features using a receptive field.

The first convolutional layer of the CNN is responsible for extracting low-level features such as edges and colors. By stacking convolutional layers, the network can also extract high-level features. This results in a network that can learn features related to the elements to be detected.

Each convolutional layer requires a number of image frames to cover the entire image. To convolve the layer kernels with these frames requires that they have equal size. The size of the output feature map is

$$y = x - (k - 1) \tag{1.7}$$

where x is the size of the input feature map and k is the kernel size. For example, if the input image size is 50×50 and the kernel size is 3×3 then the size of the output feature map is 48×48. Thus, the convolutional operation results in a reduction in size. To avoid size reduction, padding can be applied to the convolutional operation. There are two options for padding: i)

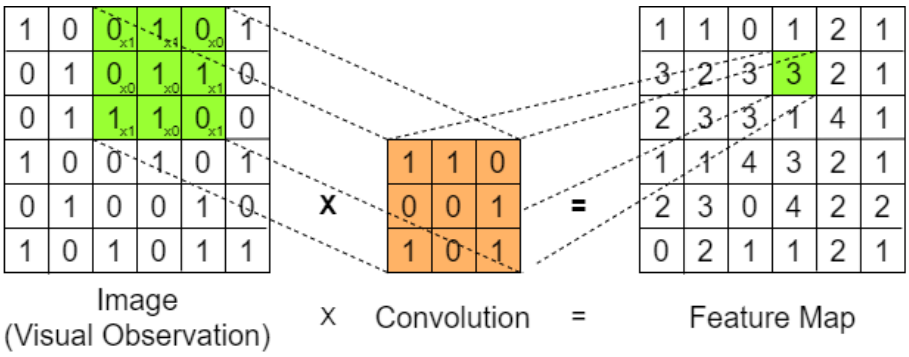
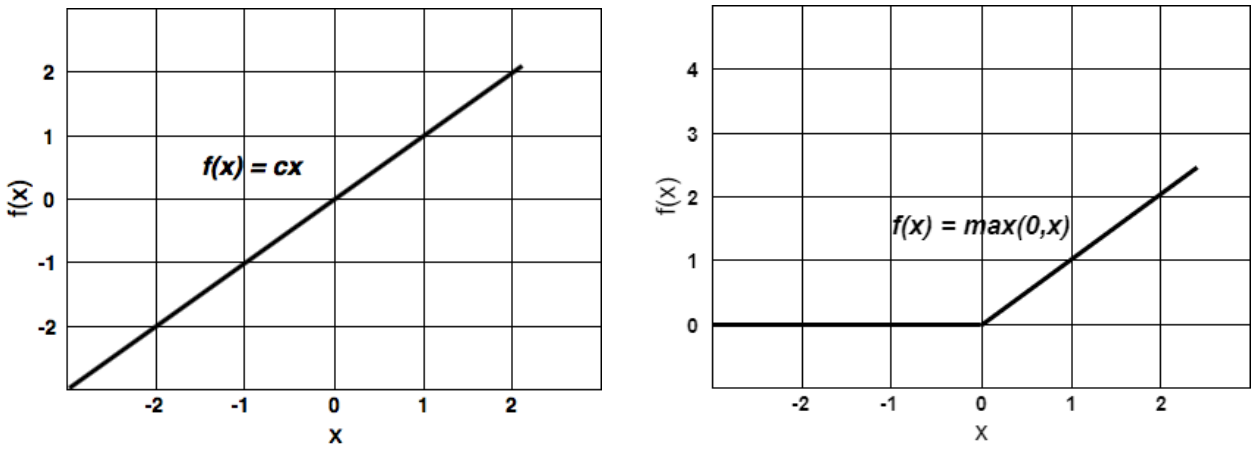


Figure 1.3: Feature extraction from an image using the convolution operation.



(a) (b)

Figure 1.4: Activation functions: (a) linear, and (b) non-linear (ReLU).

valid padding and ii) same padding. In valid padding, no padding is applied so the feature map is reduced in size compared to the input. In same padding, the feature map size is the same as that of the input by padding with zeros. The padding size is then

$$p = \frac{(k-1)}{2} \tag{1.8}$$

where k is the kernel size.

1.2.1.2 Activation function

After the convolutional layer, an activation function is applied to obtain the final results. This function is used to determine the output of the neural network. It is attached to each neuron in the network to determine whether it should be activated or not. A neuron is activated based on its input and whether it is relevant to the model prediction. The activation function maps the output of each neuron to a value between 0 and 1 or -1 and 1.

Activation functions can be linear or non-linear as shown in Figure 1.4. The derivative of a linear activation function is constant and has no relation to the input. Thus, if there is an error in

prediction, these functions cannot use backpropagation with gradient descent to improve the prediction. Backpropagation is a mechanism that is used to update the weights using gradient descent. It calculates the gradient of the error function with respect to the weights and this calculation propagates backwards through the network. With non-linear activation functions, the gradient is not constant and is related to the inputs. Thus, the weights can be modified to provide better prediction. Multiple layers of neurons can be stacked to create a deep neural network. These are used to learn complex datasets such as images, videos, and audio [28]. Examples of popular non-linear activation functions are the rectified linear unit (ReLU) [26] and sigmoid functions [27]

$$f(x) = \max(0, x) \tag{1.9}$$

$$s(x) = \frac{1}{1 + e^{-x}} \tag{1.10}$$

respectively, where x is the input.

1.2.1.3 Pooling layer

A pooling layer is another component of a CNN. The main objective of this layer is to reduce the size of the feature map to reduce the number of computations in the network. This can be done by removing features that are not important. Examples of pooling layers are max pooling [29] and average pooling [30].

In max pooling, the maximum value is selected for each patch of the feature map as shown in Figure 1.5(a). This is useful when the background is dark or for classification tasks where features such as sharp edges are important as in forgery detection. In average pooling, the average value is calculated for each patch of the feature map as shown in Figure 1.5(b). Thus, the output feature map is smoothed so edges may not be detected when this pooling method is used. However, average pooling uses all values to create the feature maps.

1.2.1.4 Loss function

A loss function is used to evaluate how well the proposed network models the given data. First, the difference between predicted and true values is calculated which is known as the cost. Then, the derivative of the cost with respect to the weights is determined which is known as the gradient. Finally, the weights are updated according to

$$\omega_j = \omega_j - \alpha \frac{\partial J}{\partial \omega_j} \tag{1.11}$$

where ω_j is the input weights, α is the learning rate, and $\frac{\partial J}{\partial \omega_j}$ is the gradient.

There are several types of loss functions such as binary cross-entropy and categorical cross-entropy [31]. The choice of a loss function for a deep learning model depends on the type of output. For example, binary cross-entropy is used to generate binary outputs (two classes) for segmentation problems and is given by

$$L_{BC} = -y \log(p(y)) - (1 - y) \log(1 - p(y)), \tag{1.12}$$

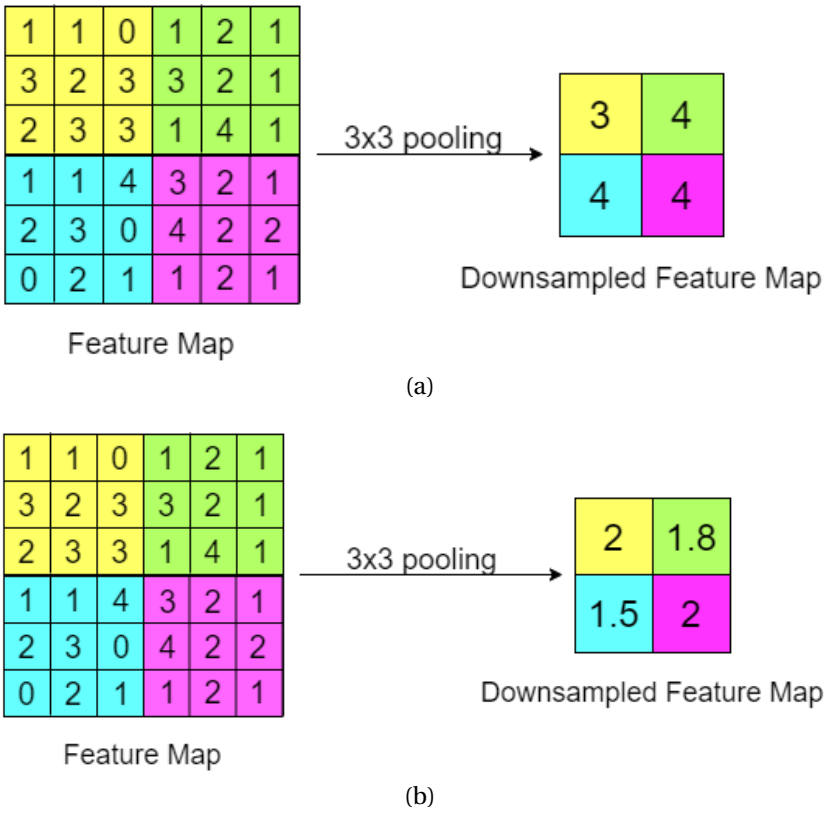


Figure 1.5: Types of pooling operations: a) max pooling, and b) average pooling.

where y is the ground truth label (0 or 1) and $p(y)$ is the prediction probability ($0 \leq p(y) \leq 1$). In the first term, for each pixel that belongs to one class ($y = 1$), it adds $\log(p(y))$ to the loss, that is, the log probability of it being a pixel in that class. Thus, if the probability is high then the log probability is low and a small value is added to the loss. Conversely, if the probability is low then the log probability is high and a large value is added to the loss. The categorical cross-entropy is used to generate categorical outputs (more than two classes) for classification problems and is given by

$$L_{CC} = \sum_{i=1}^{N} y_i \log(p(y_i)), \tag{1.13}$$

where N is the number of classes, y_i is the ground truth label for class i and $p(y_i)$ is the prediction probability for class i.

In recent years, CNNs have been used to detect image forgeries [32]. In forged images, the image feature statistics have been changed and artifacts introduced resulting in inconsistencies. In this case, a CNN is designed to learn artifacts from tampered regions. However, a robust CNN model has to be trained on a large dataset that contains different examples of forged images.

1.3 Data augmentation

Data augmentation is a technique that can be used to expand the size of a dataset by creating modified versions of images in the dataset. This can be done by applying transforms to examples from the dataset to create new and different images. These transforms include operations such as rotation, scaling, and shifting. This section discusses augmentation techniques based on geometrical transformations and other image processing functions.

A geometrical transformation G maps pixel $P = (x, y)$ from an image I to pixel $P' = (x', y')$ of image I' such that $P' = G(P)$ [33]. A rigid geometrical transformation preserves the Euclidean distance between pairs of pixels such that the size and shape of the figure remain the same. It consists of two components, translation and rotation. The translation component is represented by a two-dimensional vector with parameters x and y. The rotation component is represented by the rotation angle θ. Thus, the transformation has three parameters x, y and θ. These parameters map a pixel (x, y) from image I to a pixel (x', y') in image I'

$$I'_{x',y'} = G(I_{x,y}) = T(t_x, t_y) + R(\theta) I_{x,y} \tag{1.14}$$

where T and R represent the translation and rotation operations, respectively, and t_x and t_y are the translation parameters in the x and y directions, respectively. This can be expressed as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \end{pmatrix} + \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{1.15}$$

A geometrical transformation can also have a scaling parameter for uniform image scaling. The transformation then has four parameters, t_x, t_y, s and θ and can be expressed as

$$I'_{x',y'} = G(I_{x,y}) = T(t_x, t_y) + S_{x,y} R(\theta) I_{x,y} \tag{1.16}$$

or

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \end{pmatrix} + \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{1.17}$$

where s_x and s_y scale the object in the x and y directions, respectively. Figure 1.6 shows examples of rigid transformations.

An affine transformation is a more general form of rigid transformation. It preserves collinearity and the ratio of distances between points. Collinearity means all points lying on a line before transformation still lie on a line after transformation. Image shearing is an example of an affine transformation which changes the aspect ratio via nonuniform scaling. An affine transformation can be represented as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix} \tag{1.18}$$

where $a_{11}, a_{12}, a_{21}, a_{22}, t_x$, and t_y are the transformation parameters. Figure 1.7 shows examples of affine transformations.

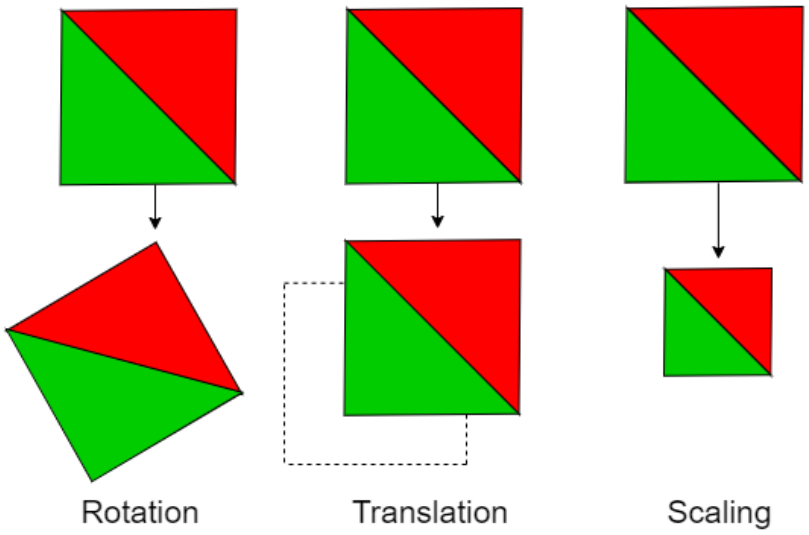


Figure 1.6: Examples of rigid transformations.

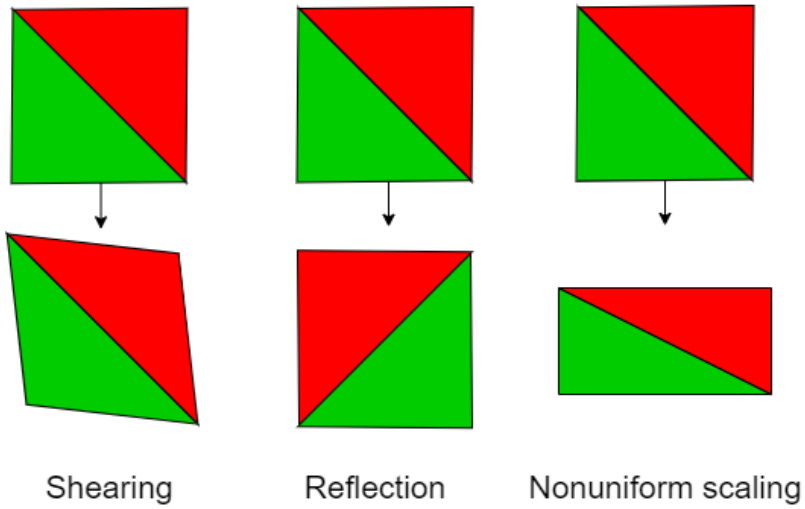


Figure 1.7: Examples of affine transformations.

1.4 Problem statement

Advances in technology along with the plethora of social media have dramatically increased the number of digital images that are produced everyday. Digital images are used to document evidence for legal purposes as well as for medical diagnostic purposes. Image editing tools have become commonplace and even sophisticated software is available free of charge. This allows anyone with a computer to easily manipulate an image. Thus, forged images can be found everywhere, in social media, in court rooms, and in newspapers.

The key research objectives addressed in this dissertation are as follows.

(i) Detect well known kinds of forgeries such as copy-move and splicing forgeries.

(ii) Create a robust and effective method for image forgery detection with low time complexity.

(iii) Study the effect of image post-processing on the prediction accuracy.

(iv) Detect multiple kinds of forgery using a single method.

This dissertation considers the use of block-based methods to detect copy-move forgeries with pixel-level accuracy. A complexity analysis is conducted to show that the proposed method has lower time complexity than other block-based methods. Despite this, block-based methods still have high time complexity.

Deep learning approaches are used to reduce the time complexity for image forgery detection. While many layers are required to train the network, training is conducted once and the trained model can then be used to detect image forgeries with low time complexity. Deep learning techniques require a vast amount of data to train a robust model. Thus, an algorithm is introduced to generate synthetic forged images for training. Furthermore, augmentation techniques are used to increase the robustness of the model in detecting forgeries in post-processed images.

Most techniques have been developed to detect only one kind of forgery. For this reason, a method is presented which can detect multiple kinds of forgeries. This method can also discriminate between kinds of forgeries based on the extracted features.

1.5 Outline

The goal of this work is to introduce new forgery detection techniques to validate images. The dissertation is organized as follows.

- Chapter 1 provided the backgroud and a description of the problems studied in this research.
- Chapter 2 introduces a new method for detecting copy-move forgeries. In this method, an image is first partitioned into blocks of fixed size. The steerable pyramid and singular value decomposition (SVD) techniques are used to decompose and extract features from the image blocks. Then, the features are sorted lexicographically and matched using the

Kolmogorov-Smirnov (KS) test. A comparison is made with other techniques which shows that this method provides better accuracy using fewer features per image block than existing methods.

- Chapter 3 introduces a new method for the detection of image splicing which is based on deep learning. It employs the Mask-RCNN architecture to generate masks for spliced regions in images. It is specifically designed to learn discriminative artifacts from tampered regions. The proposed network is initialized using three different initialization techniques for comparison. It is shown that initialization with ImageNet weights provides better results than other techniques in the literature. Different backbones are also tested and compared, and evaluation is done on a computer generated dataset.

- Chapter 4 presents a deep learning method for the detection of copy-move forgeries in post-processed images. This method is based on an encoder-decoder architecture designed to learn discriminative artifacts from the boundaries of forged regions. The performance is evaluated using six post-processing techniques, namely brightness change, contrast adjustment, color reduction, image blurring, JPEG compression, and noise addition. This method is compared with other techniques in the literature using the well-known CoMoFoD dataset.

- Chapter 5 presents a deep learning architecture called PADNET to detect multiple kinds of forgeries. PADNET is an end-to-end trainable deep neural network which employs a feature pyramid network (FPN) to aggregate features from multiscale levels of a ResNet-50 backbone. The feature maps are then used to train a DeepUNet architecture designed to learn discriminative features considering both high-level global features and low-level local features. PADNET is tested using two loss functions, binary cross-entropy and weighted binary cross-entropy. The performance of PADNET when trained on only the boundaries of the forged areas is compared to the network trained on the entire forged regions. The performance is evaluated using the well-known CoMoFoD dataset for copy-move forgery and CASIA1 for image splicing forgery.

- Chapter 6 concludes this dissertation and suggests topics for future work.

Chapter 2

Blind Copy-Move Forgery Detection Using SVD and KS Test

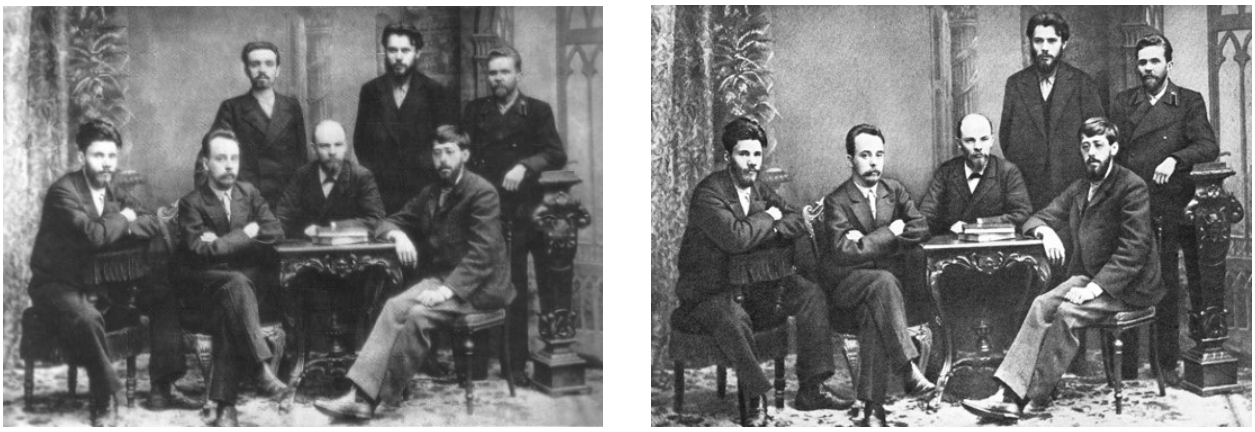
Digital forensics is a branch of forensic science which deals with the problem of determining the authenticity of digital data. Digital data such as images play a significant role in digital forensics since they are a main source of information. With the advent of image editing software, digital images can easily be manipulated. Tampered images can be found everywhere and this has eroded confidence in the reliability of digital images.

Digital image forensics can be categorized into two groups: active and passive. In the active approach [6], a digital watermark or signature is embedded into the image to verify its integrity and authenticity. A digital watermark can be visually undetectable but can be used to detect changes in image pixels and locate where the changes occurred [34]. However, watermark removal software is readily available at no cost and the images are still vulnerable to forgery. For this reason, passive methods have been introduced which require no prior information to detect tampering [7].

Images can be modified in several ways such as image splicing, retouching, and copy-move forgery [4]. Image retouching alters an image in order to change the look of a subject [10]. Image splicing refers to copying a part of an image and pasting it onto another image. Most techniques to detect image splicing rely on the sharp edges and corners of the pasted region and the inconsistency in the color of the pasted region compared to the original image [35]. In contrast, copy-move forgery refers to copying a part of an image and pasting it onto the same image. Since the pasted region comes from the same image, the color is typically consistent which makes it hard to detect. Thus, copy-move forgery is more difficult to detect than other types and is the focus of this chapter.

In this chapter, a new copy-move forgery detection technique is introduced which is based on singular vector decomposition (SVD) for features extraction and Kolmogorov-Smirnov (KS) test for decomposition. The proposed method can detect copy-move forgery in images with accuracy up to the pixel level by using only 4 features per image block of dimension 16×16. Four different post-processing techniques are considered, namely brightness change (BC), contrast adjustment (CA), color reduction (CR), and image blurring (IB). Comparisons are conducted with several well-known techniques in terms of accuracy, time complexity, and image post-processing using the well-known CoMoFoD dataset [36].

The remainder of this chapter is organized as follows. Section 2 provides a review of existing



(a) (b)

Figure 2.1: Alexander Malchenko has been edited out: (a) original image, and (b) forged image.

approaches for copy-move forgery detection. Section 3 explains the structure of block-based methods and a review of the existing block-based methods. Section 4 discusses the proposed method. The experimental results, comparisons and analysis are given in section 5. Finally, the conclusions are drawn in section 6.

2.1 Copy-move forgery

Copy-move forgery has a long history beginning in the early 1900s when Alexander Malchenko (standing on the left) was edited out from the image as shown in Fig. 2.1. This task was complicated due to the limited tools available at that time. In contrast, there are now tools that can easily perform this task such as Adobe Photoshop which is available at no cost.

Copy-move forgery is usually used to maliciously hide or add information as in Fig. 2.1. Since the forged region comes from the same image, the resulting image has at least one duplicated region. The goal of detection methods is to determine these regions. Searching for duplicated regions by comparing pixels in the image is a direct solution but is slow and computationally expensive. Further, post-processing techniques such as brightness change (BC), color reduction (CR), contrast adjustments (CA), and image blurring (IB) can be used on the image or just the copied region to make the forgery harder to detect. This makes copy-move forgery detection a challenging task.

Copy-move forgery detection can be keypoint-based or block-based. In block-based methods, an image is partitioned into fixed size overlapping or nonoverlapping rectangular or circular blocks. A feature vector is extracted for each block and these vectors are matched by calculating the distance between them. This distance can be Euclidean distance [16,37], Hamming distance [38], Hausdorff distance [39], logical distance [40], correlation coefficient [17,41], phase correlation [11,42], or local sensitive hashing [12,43]. The main concern with block-based methods is their computational complexity.

Keypoint-based methods detect and describe local features in an image using techniques such as the scale invariant feature transform (SIFT) and speeded up robust features (SURF) [44].

These features are used to find matching regions in the image and if two regions have similar keypoint features, one is assumed to be forged. However, these methods can fail when the forged regions have been modified using techniques such as image retouching [45]. For this reason, block-based methods are employed here. The next section presents a review of copy-move forgery detection using these methods.

2.2 Block-based methods

A number of block-based methods have been proposed to detect copy-move forgery [13, 19]. The goal is to find similar regions in a forged image.

2.2.1 Preprocessing

Some block-based methods require the image to be preprocessed before any further analysis. In [10–14] an image was converted to grayscale using the luminance formula $Y = 0.299R + 0.587G + 0.114B$, where R, G, and B represent the red, green, and blue luminance respectively, of the colour image. Each color is represented by a vector which provides the brightness intensity, for example 0 to 255 for 8-bit images.

2.2.2 Image partitioning

In block-based methods, an image is partitioned into fixed size overlapping or non-overlapping blocks. Square block partitioning was used in [10, 13, 19, 43, 47]. Circular partitioning was used in [16, 17] to reduce the dimension of each block compared to square blocks which lowers the computational complexity. After an image is partitioned into blocks, features can be extracted from each block to find those with similar features. Other methods operate directly on the blocks without feature extraction such as those in [18].

2.2.3 Feature extraction and matching techniques

Copy-move forgery detection is used to find replicated regions in an image. However, the copied region can be processed prior to pasting into another part of the image to make it harder to detect. The detection accuracy depends on finding matching features in a copy-move pair of blocks in an image even if the image has undergone post-processing such as blurring, color reduction, brightness change, or contrast adjustment [48]. To accomplish this goal, several procedures have been proposed.

In [19], a method for copy-move forgery detection was given which is based on the Discrete Cosine Transform (DCT). In this approach, DCT coefficients are extracted for each block. To reduce the complexity, the coefficients are sorted lexicographically and then quantized to reduce the effects of noise, JPEG compression, brightness change or other post-processing. After quantization, the coefficients for each block are compared and a block is said to be copied if these coefficients are comparable. To reduce the false positive rate, a region is detected as forged only if there is a cluster of copied blocks in the region. However, this may result in small forged regions going undetected.

In [50], the DCT coefficients were considered as eigenvalues for each block. Then, the distances between the eigenvectors of all blocks were calculated and lexicographically sorted to reduce the false positive rate [19]. If the distance between two blocks is less than a certain threshold, the block is considered to be a duplicate. This method is robust to noise and JPEG compression but not to other post-processing operations. Comparing block features makes block-based partitioning methods complex. As a result, several techniques have been developed to lower the complexity by reducing the length of the feature vectors. Cao et al. [47, 49] reduced the feature vector length by using circular blocks instead of square blocks. Each block is divided into four regions. DCT is applied to each region so each is represented by DCT coefficients. The mean of the coefficients in each region is used as a feature resulting in 4 features per block. This method is robust to noise and blurring, and can detect multiple forgeries in the same image. However, it is not robust against rotation or scaling.

Several block-based algorithms have been proposed to reduce the feature vector length while being robust to post-processing operations. A log-polar transform was used in [17,50,54]. This transform maps the image blocks to log-polar coordinates [51]. This approach is robust to post-processing operations such as rotation, scaling, mirroring, illumination modification, and chrominance modification, but not against other techniques such as noise, JPEG compression or blurring.

SVD is a matrix factorization technique that can be used to extract features from an image. Li et al. [52] proposed an algorithm based on SVD and the Discrete Wavelet Transform (DWT). The DWT is used to decompose an image into a series of coefficients corresponding to the image spatio-frequency subbands and SVD is employed to extract the feature vectors. Then these vectors are sorted lexicographically to detect duplicated regions. Zhang and Wang [96] introduced a method which combines SVD with a k-dimensional (KD) tree. A KD tree is commonly used to search for nearest neighbours. First, SVD is used to extract a feature vector for each block, and these features are organized using a KD tree for fast searching. Similar blocks are matched using the Euclidean distance

$$D(u,v) = (\sum_{i=1}^{r}((u(i)-v(i))^2)^{\frac{1}{2}},$$

where $u = (u_1, u_2, \ldots, u_r)^T$ and $v = (v_1, v_2, \ldots, v_r)^T$ are length r feature vectors. Xiaobing and Shengmin [6] used an approach similar to that in [96], but only the k largest singular values were considered and the remaining values discarded to reduce the feature vector size. Kang and Cheng [6] also retained only the largest k values. Regions were matched using the cumulative contribution which is defined as

$$\eta = (\sum_{i=1}^{k}(\lambda_i))/(\sum_{i=1}^{r}(\lambda_i)),$$

where λ is the feature vector sorted from largest to smallest and r is the number of singular values. However, this method is not robust to post-processing operations.

The size of the feature vectors is an important factor in the complexity of detection algorithms. Steerable pyramid is a technique used for image decomposition. In this method, an image is smoothed using a smoothing filter and subsampled by a factor of 2 along each coordinate. This process is repeated multiple times which looks like a pyramid if illustrated graphically. In [10], steerable pyramid decomposition was applied to image blocks to reduce the size

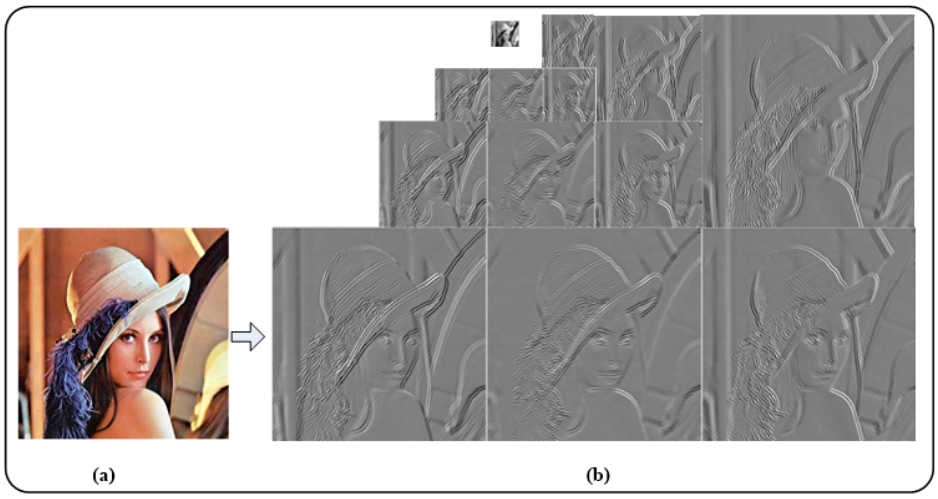


Figure 2.2: Lena Image and its decomposed steerable pyramid subbands

from 16 × 16 to 4 × 4. Then, similar blocks were extracted using the Gaussian copula which is a distribution constructed from a multivariate normal distribution.

Kolmogorov-Smirnov (KS) test

Kolmogorov-Smirnov test is used to compare two samples based on a distribution function [56]. Given samples $x_1, \ldots, x_n$ of a random variable with cumulative distribution function (CDF) F, consider the problem of testing $H_0 : F = F_n$ versus $H_1 : F \neq F_n$, where F_n is some specified distribution function. For example, in the univariate case H_0 can be tested using the KS test statistic $sup_{x \in R} \mid F_n(x) - F(x) \mid$, where $sup_{x \in R}$ is the supremum of the set of distances, F is the CDF of the reference distribution, and F_n is the empirical distribution of the samples [57]. The test statistic measures the similarity between distributions and equals 0 when they are identical.

Steerable pyramid

Steerable pyramid is a mathematical tool that is multi-orientation, multi scale image decomposition technique. This transform was first introduced in the literature in early 1990s. It is a wavelet-based representation [58]. Steerability refers to the ability of the wavelets to rotate to any orientation by forming suitable linear combinations of a primary set of equiangular directional wavelet components [59, 60]. In steerable pyramid decomposition, an image is decomposed into lowpass and highpass subbands, using steerable filters L_0 and H_0. Then, the lowpass band breaks down into a set of bandpass subbands $B_0, ..., B_k$ and lower lowpass subband H_1. The resulting lower lowpass subband is subsampled by a factor of 2 along the x and y directions. This process is repeated until we arrive at the desired scale of decomposition. Fig. 2.2 shows an image of size 128 × 128 pixels decomposed to its steerable pyramid subbands. This example shows 4 orientations and 3 scales. Each scale has 4 orientations. The first scale is 128 × 128 where the second and third scales are 64 × 64 and 32 × 32 respectively. Finally, the last subband is 16 × 16 pixels [59].

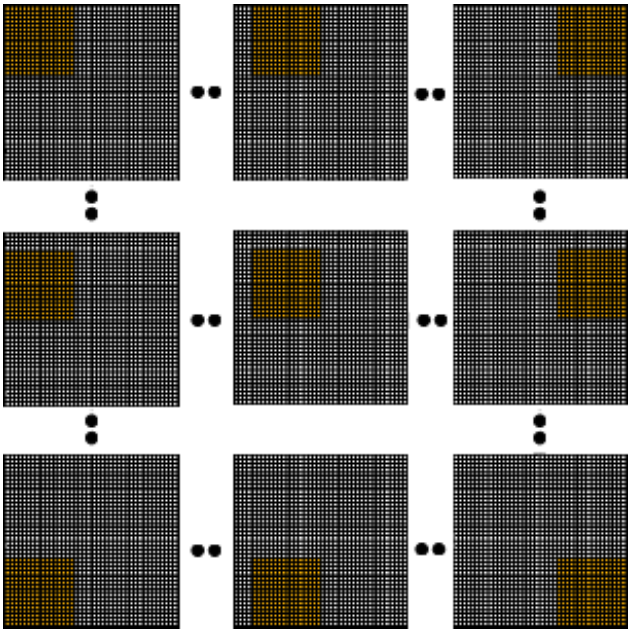


Figure 2.3: Image partitioning into 16 × 16 overlapping blocks.

Singular value decomposition (SVD)

Singular value decomposition has been used in several fields such as data compression, signal processing and pattern analysis [6, 61]. An $N \times M$ matrix $A \in R^{N \times M}$ of rank j can be factored in the form

$$A = P\Sigma Q^T \tag{2.1}$$

where

$$\Sigma = \begin{bmatrix} \Sigma_j & 0 \\ 0 & 0 \end{bmatrix} \tag{2.2}$$

is an $N \times M$ diagonal matrix and $\Sigma_r = diag(\sigma_1, \sigma_2, \ldots, \sigma_j)$ is a square diagonal matrix with positive diagonal entries called the singular values of A, and orthogonal matrices $P \in R^{N \times N}$ and $Q \in R^{M \times M}$.

2.3 The proposed method

In the proposed method, an image is partitioned into blocks of size 16×16 using a 16×16 sliding window which is shifted by one pixel per step as shown in Fig. 2.3. This results in 246,016 blocks for an image of size 512 × 512. To reduce the computational complexity, each block is decomposed into a 4 × 4 block using steerable pyramid decomposition. A 4 × 4 block has 4 nonoverlapping 2 × 2 sub-blocks. SVD is applied to each 2 × 2 sub-block to extract a single singular value which is the corresponding feature as shown in Fig. 2.4. This results in 4 features per 4×4 block. Thus, each original 16×16 block is represented by a vector of only 4 features. Fig. 2.5 shows a block diagram of the feature extraction process. The indices of the original blocks are stored with the feature vectors so that each pixel is associated with a feature vector.

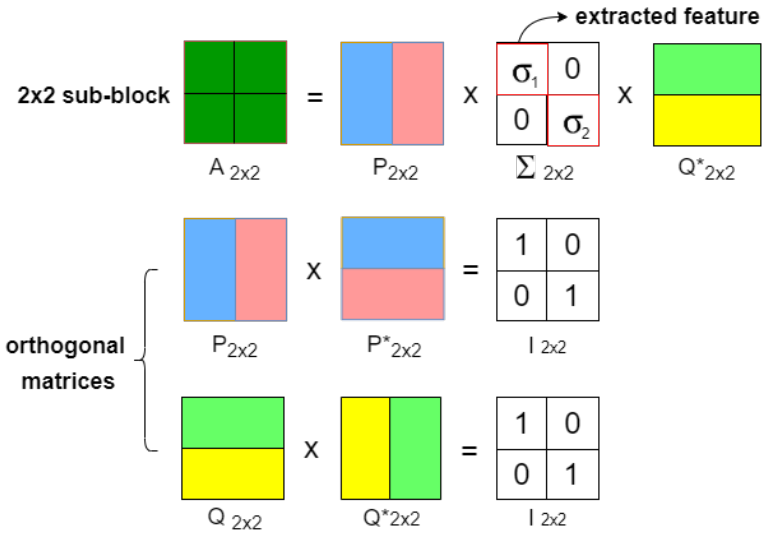


Figure 2.4: Feature extraction from 2×2 sub-blocks using SVD

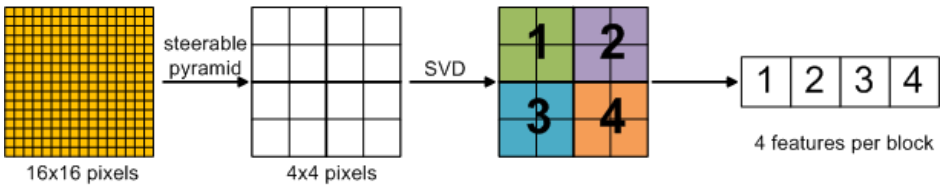


Figure 2.5: The overall feature extraction process.

The feature vectors are sorted lexicographically so that similar vectors are close which simplifies the search process. The KS test is applied to the sorted feature vectors. The empirical and reference CDFs F_n and F are represented by the two feature vectors being compared. If the test statistic is below or equal to the threshold, the two feature vectors are said to match and belong to the same distribution. Conversely, if it is above the threshold, the vectors are said to differ and belong to different distributions. Pixels belonging to the same distribution share the same features and so one group is said to be forged.

Pixels in the same region often share the same features such as brightness and color. Thus, applying the KS test to the corresponding feature vectors may increase the false positive rate. To avoid this problem, a minimum distance between pixels is employed. The distance between two pixels is defined as

$$D = |x_2 - x_1| + |y_2 - y_1|$$

where (x_1, y_1) and (x_2, y_2) are the coordinates of the pixels. If $D \geq d$ where d is a minimum distance threshold, then the pixels are said to be in different regions. This threshold should be high enough to avoid detecting pixels in the same region, but if it is too high forged pixels in different regions may be missed. A good threshold should have low false positive and false negative rates. To find the best threshold, several values were tested between 10 and 100 pixels with increments of 10 pixels. False positive and false negative rates were calculated for each

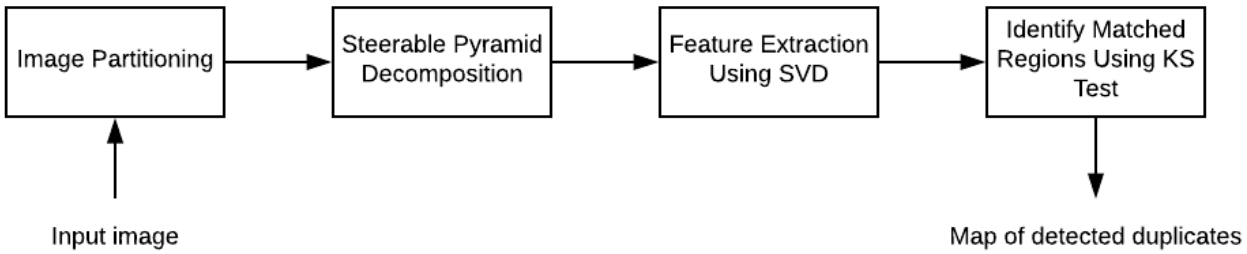


Figure 2.6: Block diagram of the proposed algorithm.

value and the best value was found to be 50 pixels. A block diagram of the proposed algorithm is shown in Fig. 2.6.

2.4 Results and discussion

In this section, experimental results are presented to evaluate the effectiveness of the proposed method for image forgery detection. This method is compared with state of the art image forgery detection methods in terms of complexity and robustness to post-processed techniques.

2.4.1 Image dataset

The CoMoFoD image database [36] is used here to evaluate the image forgery methods. This database consists of two groups of images, one with 200 images of size 512×512 and the other with 60 images of size 3000×2000. The small scale images are classified into 5 categories: translation, rotation, scaling, distortion, and combination of all previous. Each category consists of 40 images and all images are processed using 6 post-processing techniques. These techniques are brightness change (BC), contrast adjustment (CA), color reduction (CR), and image blurring (IB). The proposed method has been evaluated on the 40 images of the translation category and 4 post-processing techniques BC, CA, CR, and IB. The proposed method has also been evaluated on one high resolution image of size 3000×2000. All images have 8 bit color vectors. Forged regions have been added to these images and modified using 4 different post-processing techniques [36]. For BC, the range of the color intensity values was changed by mapping the intensity to values between lower and upper bounds. Intensity values that fall outside these bounds are set to the corresponding minimum or maximum value. For CA, the image contrast is adjusted by mapping the range of color intensity values to a new interval bounded by lower and upper bounds. For BC and CA, the bounds are $(0.01, 0.95)$, $(0.01, 0.9)$, and $(0.01, 0.8)$, denoted by 1, 2 and 3, respectively. For CR, the intensity values for each color are quantized to a smaller range. The ranges are $(0, 32), (0, 64)$, and $(0, 128)$ denoted by CR1, CR2 and CR3, respectively. For IB, the images are blurred by adding Gaussian noise to the intensity values. The mean of the noise is $\mu = 0$ and the variances are $\sigma = [0.009, 0.005, 0.0005]$ denoted by IB1, IB2 and IB3, respectively [36]. This is known as Gaussian blur. Note that IB1 is the highest level of blurring, whereas BC3, CA3, and CR3 are the highest levels of change for the other techniques. Fig. 2.7 shows examples of forged images from CoMoFoD database. Column 1 is showing the original

image, column 2 is showing the binary mask to show the duplicated regions, and columns 3 to 6 are showing examples of forged images post-processed with techniques BC, CA, CR, and IB respectively. The post-processing techniques showing in this table are at the highest level.

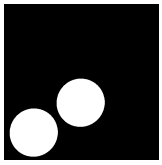
	Original Image	**Binary Mask**	**Forged Images**			
			BC	CA	CR	IB
1						
2						
3						
4						
5						

Figure 2.7: Examples of forged images from the CoMoFoD database.

2.4.2 Performance analysis

Fig. 2.8 shows the performance of the proposed algorithm on 5 images from the CoMoFoD database. These images were selected because they show the performance for different sizes and numbers of forged regions. The first two columns show the original and forged images. Columns 3 to 6 show the results with the images after post-processing has been applied to the forged region, i.e. the detection results after BC1, CA1, CR1 and IB1 post-processing. The original region is circled in green and the forged region is circled in red. Thus in the first image, the bird was copied along with part of the adjacent area to make the color more consistent with the

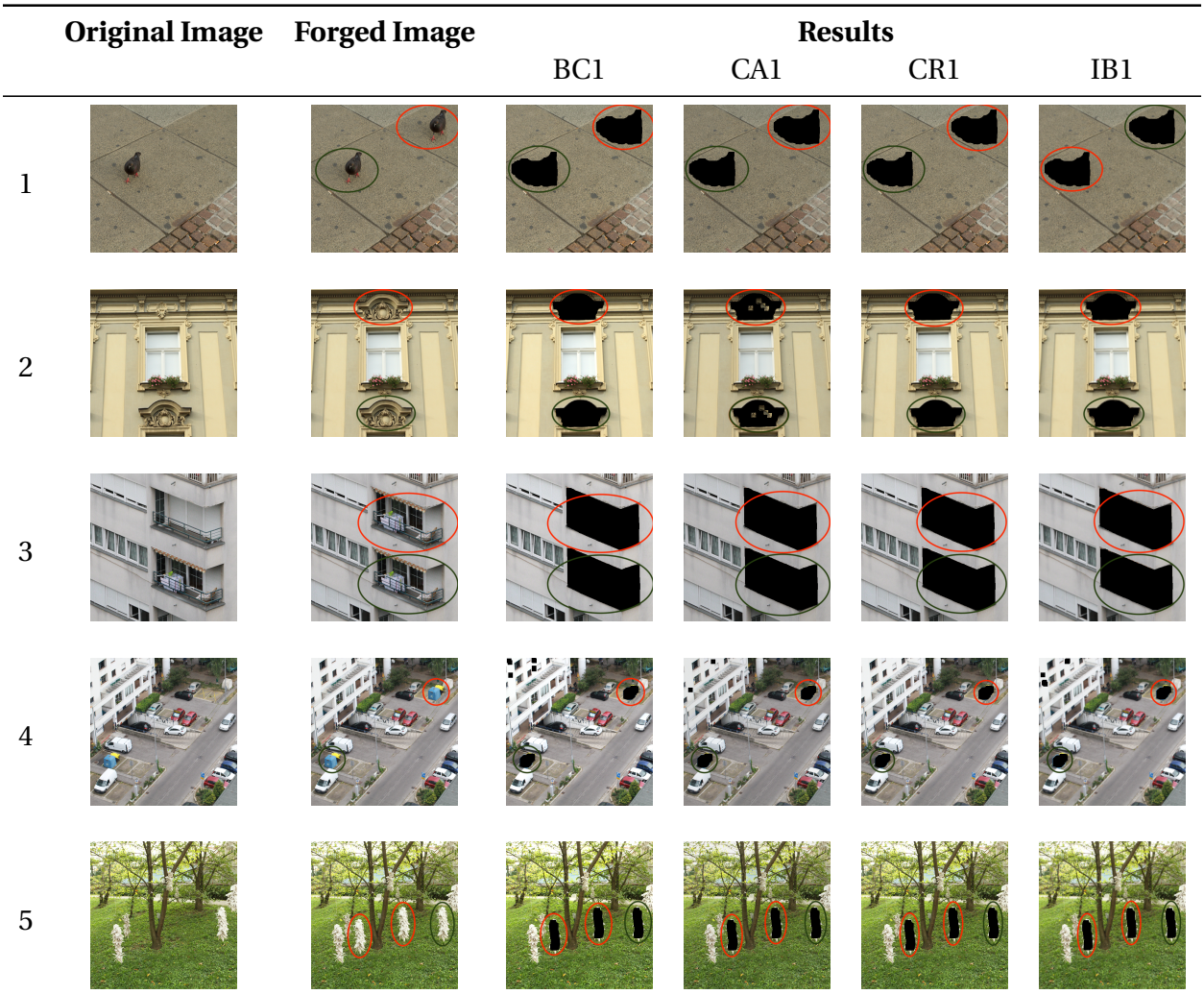


Figure 2.8: The results obtained using the CoMoFoD database.

surrounding area after pasting. This makes the forged region hard to detect. In all cases, the proposed algorithm was able to detect the forged regions. The proposed method was able to locate and mask the forged region accurately even with a small region as in image 4 and multiple regions as in image 5.

To evaluate the robustness of the proposed algorithm against post-processing, all 3 levels of each technique are considered. Fig. 2.9 shows the results for four images from the database. This shows that the forged regions were detected in all 12 cases. The accuracy of the proposed algorithm for the 200 images in the database is evaluated using precision and recall. Typically, precision and recall are calculated at the image level, i.e. based on how many images are correctly classified as forged, but to provide more accurate results, the precision and recall are calculated per pixel here. In other words, precision and recall are calculated based on the percentage of pixels in the image that are correctly classified.

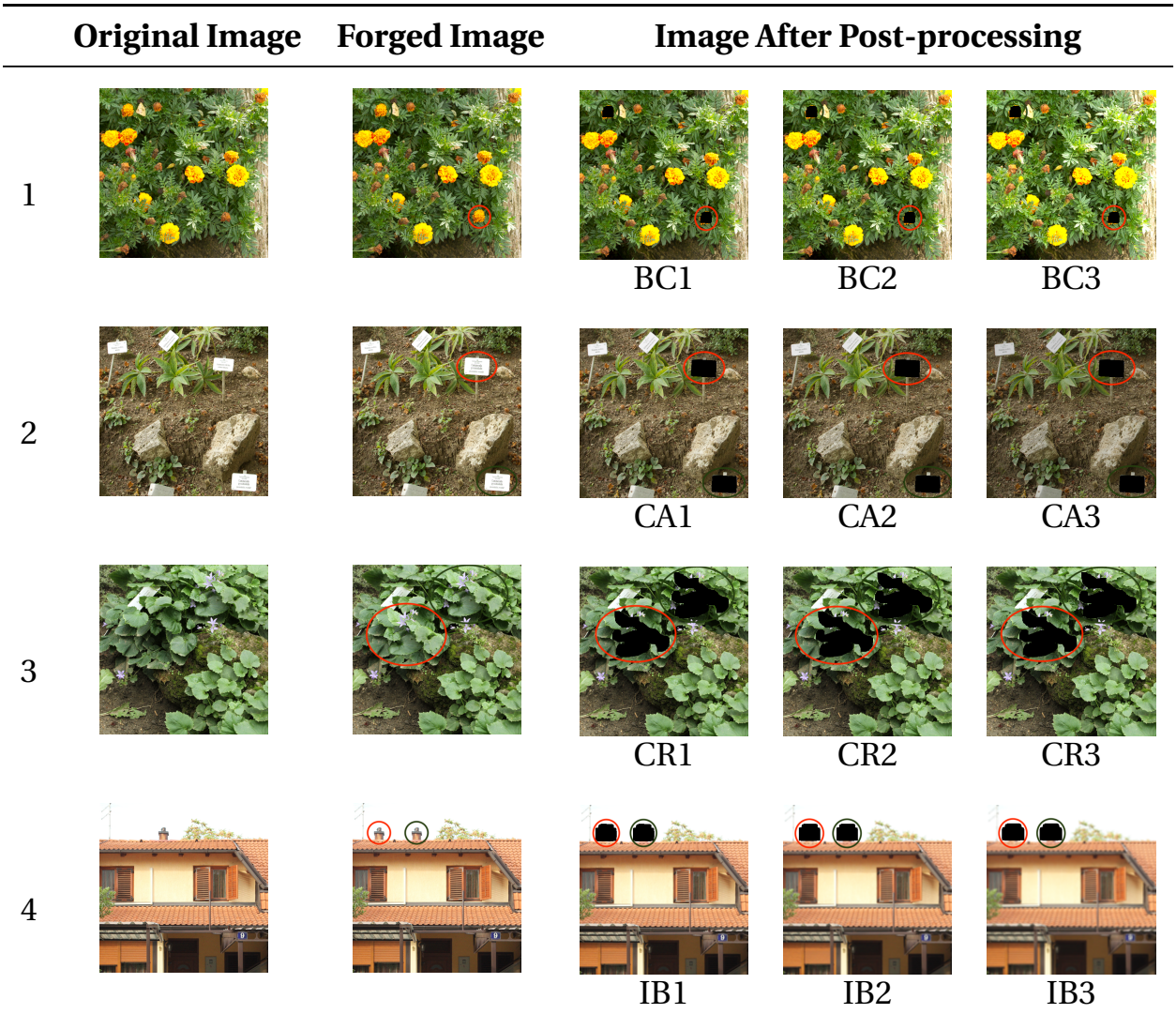


Figure 2.9: Four images altered using different levels of post-processing.

The precision, recall and F1_score are given by

$$\text{Precision} = \frac{\text{Forged region} \cap \text{Detected region}}{\text{Detected region}} = \frac{TP}{TP+FP}, \tag{2.3}$$

$$\text{Recall} = \frac{\text{Forged region} \cap \text{Detected region}}{\text{Forged region}} = \frac{TP}{TP+FN}, \tag{2.4}$$

$$\text{F1\_score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{2.5}$$

where TP is the true positive rate which is the number of forged pixels detected as forged, FP is the false positive rate which is the number of pixels incorrectly detected as forged, and FN is the false negative rate which is the number of forged pixels incorrectly detected as original. Precision is a good measure of the false positive rate whereas recall is a good measure of the false negative rate. F1_score is the weighted average of precision and recall which means it takes both false positives and false negatives into account. A good detection technique should have high precision, high recall, and high F1_score.

To calculate TP, FP and FN, the results are binarized using the Otsu method [46] for comparison with the binary masks from the database. In a binary mask, the intensity values of the forged pixels are set to 1 and the intensity values of the original pixels are set to 0. Fig. 2.10 illustrates the precision and recall for the three levels of each of the four post-processing techniques. The results obtained are calculated by averaging over all 40 images chosen for evaluating this method. These results show that the precision and recall rates are almost constant over the levels with a maximum variation of ±2% except for blurring which reduces the precision to 34% at the highest level. This consistency reflects the robustness of the proposed algorithm even with image post-processing. The precision achieved is more than 95% for three of the post-processing techniques, and except for the highest level of blurring it is greater than 75%.

Table 2.1 presents FP for the three levels of each post-processing technique. This shows that FP increases as the level of post-processing increases and reaches a maximum of just over 1% except for image blurring which is 4% for the highest level of blurring ($\mu = 0$, $\sigma = 0.0005$).

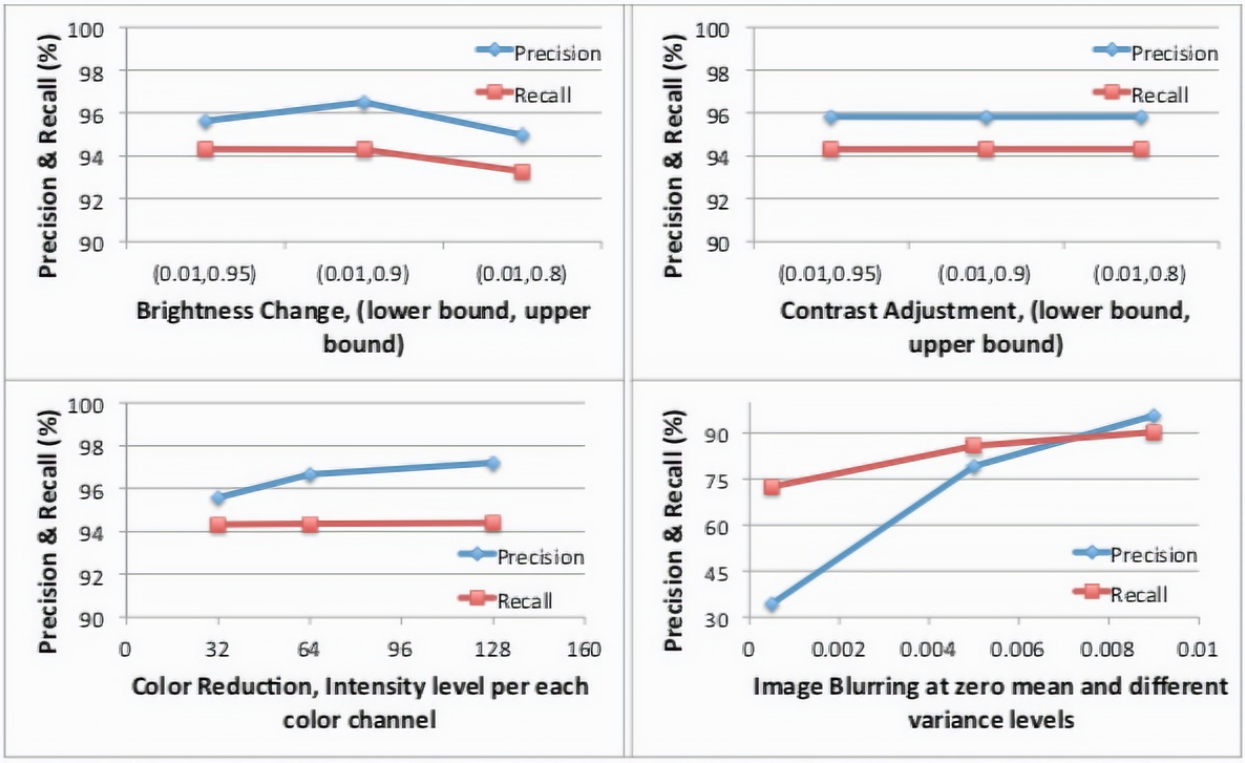


Figure 2.10: Precision and recall with image post-processing (a) brightness change, (b) contrast adjustment, (c) color reduction, and (d) image blurring.

Table 2.2 presents the results for one image with the proposed and three other methods [10, 13, 19]. The binary mask is shown along with the result for the other three methods for comparison with the proposed method. The forged region and the region where it was copied from are shown in white and the reminder of the image is shown in black. The feature size per block for each method is given in the last column. Table 2.3 presents the results for one image of size 3000×2000 from the CoMoFoD high resolution database. The proposed method was able to detect the forged regions with an F1_score of 68.5%. The high precision and low recall mean that the proposed method has a low false positive as well as a high false negative ratio. Conversely, the method [19] was not able to detect the forgeries in this image due to the high complexity of the feature vectors used per block (i.e. 64 features) which exceeded the maximum array size of the software used.

The size of the feature vectors has a significant effect on the complexity. Thus, the main goal of the methods in [10, 13, 19] is to reduce the feature size while providing good detection performance. However, applying post-processing to an image may result in losing important features that can be used to find similar image blocks. For this reason, reducing feature size may result in the loss of more information which will affect the robustness against post-processing techniques. The last column in Table. 2.3 shows that the proposed method was able to accurately detect forged regions using only 4 features per block compared to the other methods which require 64, 32, and 16 features per block. The performance using the 200 small size images in the CoMoFoD database is given in Tables 2.4 and 2.5. In terms of precision, recall, and F1_score, the proposed method is the best while the technique in [19] is better in terms of the compu-

Post-processing technique	Parameters	FP (%)
BC (lower bound, upper bound)	(0.01, 0.95)	0.60
	(0.01, 0.9)	0.76
	(0.01, 0.8)	1.08
CA (lower bound, upper bound)	(0.01, 0.95)	0.57
	(0.01, 0.9)	0.57
	(0.01, 0.8)	0.57
CR (lower bound, upper bound)	(0, 32)	0.58
	(0, 64)	0.56
	(0, 128)	0.54
IB	$\mu = 0$, $\sigma_2 = 0.009$	1.11
	$\mu = 0$, $\sigma_2 = 0.005$	1.80
	$\mu = 0$, $\sigma_2 = 0.0005$	4.08

Table 2.1: FP for different levels of post-processing.

tation time. This is because the Big-O complexity of computing the DCT coefficients is $O(N)$ while the Big-O complexity of SVD is $O(N^2)$. However, the SVD features are better suited to image post-processing as the proposed method performed better than the other techniques for all four post-processing techniques using the lowest number of features.

2.5 Conclusion

In this chapter, a new copy-move forgery detection algorithm was presented which is based on SVD and the KS test. The singular values are used as features. The feature vectors are sorted lexicographically to reduce the false positive rate. The KS test is then used to extract similar features. The performance of this method was tested using the CoMoFoD database. The results obtained show that the proposed method can detect forgeries at a pixel level and has a low false positive rate of less than 4%. Furthermore, this method is robust to post-processing techniques such as brightness change, contrast adjustment, color reduction, and image blurring. Although this method has a slightly longer computation time compared to one other method, the proposed method had the highest precision, recall, and F1_score which averaged 95% for brightness change, contrast adjustment, and color reduction. For image blurring, it achieved the highest precision, recall, and F1_score which are 70%, 82.7%, and 75%, respectively. The proposed method uses only 4 features per block which is smaller than the number used in the other methods.

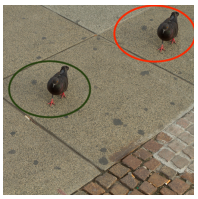
Original image	Forged image	Binary mask	Method	Results	Feature size
			Ref. [19]		64
			Ref. [13]		32
			Ref. [10]		16
			Proposed		4

Table 2.2: Copy-move forgery detection results for the proposed and three other methods [10, 13, 19].

Original Image	Forged Image	Binary Mask	Results	Precision	Recall	F1_score
				99.8	52.2	68.5

Table 2.3: The results obtained for a 3000 × 2000 image from the CoMoFoD database.

Method	Brightness Change			Contrast Adjustment			Computation time (s)
	Precision (%)	Recall (%)	F1-score (%)	Precision (%)	Recall (%)	F1-score (%)	
Ref. [10]	59.3	46.4	49.7	57.1	44.5	47.7	2495
Ref. [19]	57.5	76.9	58.4	53.7	78.8	55.0	51
Proposed	95.7	93.9	94.8	95.9	94.2	95.0	604

Table 2.4: Comparison of the proposed and two other methods [10, 19].

Method	**Color Reduction**			**Image Blurring**			**Computation time (s)**
	Precision (%)	Recall (%)	F1-score (%)	Precision (%)	Recall (%)	F1-score (%)	
Ref. [10]	59.1	46.4	49.5	39.2	30.7	32.6	2495
Ref. [19]	57.0	78.8	58.5	53.0	70.6	51.0	51
Proposed	96.5	95.3	95.9	70.0	82.7	75.8	604

Table 2.5: Comparison of the proposed and two other methods [10, 19].

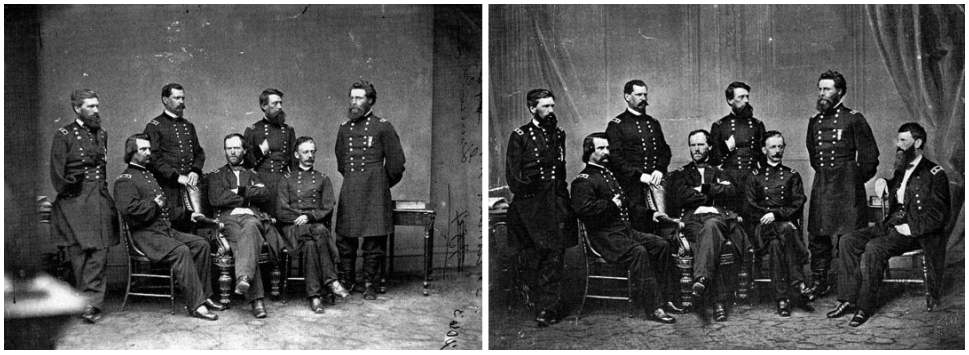


Figure 3.1: Example of image splicing [62].

Chapter 3

Image Splicing Detection Using Mask-RCNN

The advent of the internet along with the plethora of social media and other applications has made digital images a dominant source of information. They are used to document evidence for legal purposes as well as in medical imaging for diagnostic purposes, sports, and many other fields [1–3]. While digital imaging provides numerous possibilities for creation, it can also be used to produce forged documents. Image forgery is almost as old as photography itself and started as early as 1865 when photographer Mathew Brady added General Francis P. Blair to an original photograph to make it appear that he was present as shown in Fig. 3.1.

Image editing tools have become commonplace and even sophisticated software is available free of charge. This allows anyone with a computer to easily manipulate an image. As a consequence, forged images can be found everywhere, on the covers of magazines, in courtrooms, and on the internet. Therefore, a robust and effective method for image forgery detection is of great importance in digital image forensics.

Image forgery detection techniques can be classified into two categories: active and passive. In active methods, certain information is embedded into the image during creation such as a watermark or signature. With these methods, image tampering is detected by analyzing the watermark or signature. Although watermarking can protect an image from theft, its application is limited because human intervention is required to recover the original watermark-free image. Conversely, passive techniques do not require manual processing [7]. Forgery changes the image feature statistics and introduces artifacts resulting in inconsistencies. Most passive techniques use these inconsistencies to identify forged images.

Image tampering can be done in several ways such as image splicing, retouching, and copy-move forgery [4]. Copy-move forgery refers to copying a part of an image and pasting it into the same image to conceal or change information. Several methods have been introduced to detect and localize copy-move forgeries [63]. Image retouching is used to change the appearance of a subject in an image [5]. In image splicing, part of an image is copied and pasted into another image to hide or add information. Image splicing is widely used to create forgeries in images. Several approaches have been proposed to detect image splicing based on the abnormal transients at splicing boundaries. A method for image splicing detection based on a natural image model was introduced in [64]. This model uses statistical features extracted from the image and 2-D arrays generated by applying a multi-size block discrete cosine transform (MBDCT) to the image. The statistical features include moments of characteristic functions of wavelet subbands and Markov transition probability matrices. In [65], the method in [64] was improved by capturing intra-block and inter-block correlations using DCT coefficients. The original Markov model obtained using a discrete wavelet transform (DWT) was used to extract addiitonal features. Then, the cross-domain features were used to train a support vector machine (SVM) classifier.

In [66], grey level run length matrix (GLRLM) texture features for forged and original images were determined. Then, statistical features extracted from the GLRLM were used to train an SVM for classification. In [67], an approach based on statistical features obtained from the run length and image edges was proposed. The method in [67] was improved in [68] by using a detection algorithm based on approximate run lengths. This improved the detection accuracy from 69% to 75% in less time.

Deep neural networks such as deep belief network [69], deep autoencoder [70] and Convolutional Neural Network (CNN) [71] have recently been used to extract useful, high-level structure representations. This allows deep learning networks to generalize well across a wide variety of tasks such as image classification [28] and speech recognition [72]. This has led to their use in source identification [73,74] and image and video manipulation detection [32,75–77]. A CNN model was introduced by in [32] to detect image splicing by extracting dense features from image patches. These features are concatenated and then a pooling operation (max or mean) is applied on each patch feature. Then, an SVM classifier is trained on these features for classification. In [78], a two-stage deep learning approach was introduced to detect tampering in images. First, a stacked autoencoder model is trained on the wavelet features of the images to learn complex features for each image patch. Then, the contextual information of each patch is integrated and used for detection.

A good detection method should consider low resolution images resulting from compression or resizing. Thus in [79], a shallow CNN (SCNN) was trained to distinguish the boundaries of forged regions from the original edges in low resolution images by discriminating changes in chroma and saturation. This was done by first converting the image from RGB space to YCrCb space. Then, only the CrCb channels were used in the convolutional layers to exclude the illumination information. A different approach to copy-move forgery detection (CMFD) based on a CNN was proposed in [80]. In this method, a pre-trained CNN model was fine tuned using 3000 forged images from the ImageNet database. These images were generated by randomly moving a rectanglular block from the upper left corner of the images to the centre.

In this chapter, a supervised Mask-RCNN is used to learn the hierarchical features resulting from forgery operations such as image splicing. To generate the initial feature map, a new

ResNet architecture called ResNet-conv is introduced. This is obtained by replacing the feature pyramid network FPN in ResNet-FPN [81] with a set of new convolutional layers. A comparison is made between ResNet-FPN and ResNet-conv in terms of the convergence speed. Different ResNet architectures have been tested and compared including ResNet-50 and ResNet-101 [82]. For faster convergence, a transfer learning strategy is used to initialize the proposed network. Different initialization techniques have been developed and compared such as ImageNet [83], Xavier_normal [84] and He_normal [24]. The proposed method is shown to have a higher efficiency than these techniques.

The remainder of this chapter is organized as follows. Section 2 briefly discuss the framework of Mask-RCNN, Section 3 presents the algorithm used to generate the dataset, the experimental results, comparisons and analysis. Finally, the conclusions are drawn in section 4.

3.1 Mask-RCNN

A CNN consists of several convolutional and pooling layers and ends with one or more fully connected layers. Each convolutional layer consists of three steps, convolution, non-linear activation and pooling. After each convolutional layer, a feature map is generated and passed to the next layer. A convolutional layer can be represented by [32]

$$F^n(X) = pooling(f^n(F^{n-1}(X) * W^n + B^n)) \tag{3.1}$$

where $F^n(X)$ is the feature map for layer n of the convolution with kernel and bias given by W^n and B^n, respectively, and $*$ denotes convolution.

The mask regional convolutional neural network (Mask-RCNN) model was developed in [65] for semantic segmentation, object localization, and object instance segmentation. Mask-RCNN outperformed all existing single model entries on every task in the 2016 COCO challenge including large-scale object detection, segmentation, and captioning dataset [86]. Mask-RCNN consists of two stages. The first stage, called region proposal network (RPN), scans the initial feature maps and generates region proposals or regions of interest (RoI) which is the same process employed by the faster-RCNN model [87]. In the second stage, an operation known as RoI-pooling [88] is applied to each RoI to downsample the feature map by using a nearest neighbour approach. This process selects important features from the feature map. RoI-pooling can result in misalignment between the RoI and the extracted features, so RoI-align is applied to each RoI to create more accurate RoIs. In RoI-align, the value of each sample point is calculated using bilinear interpolation from the nearby grid points on the feature map. In addition to predicting the class and bounding boxes for each object, Mask-RCNN also generates a binary mask for each RoI using a fully convolutional network (FCN). Fig. 3.2 shows the framework of the Mask-RCNN network.

Both stages of the Mask-RCNN are connected to the backbone structure. The backbone is another deep neural network that is used to create the initial feature map. In principle, the backbone network could be any CNN pre-trained on an image dataset such as ResNet [82]. ResNet is an artificial neural network (ANN) that is based on residual learning. In residual learning, a network is trained by feeding the output of an initial layer to advanced layers to share the earlier residuals [82].

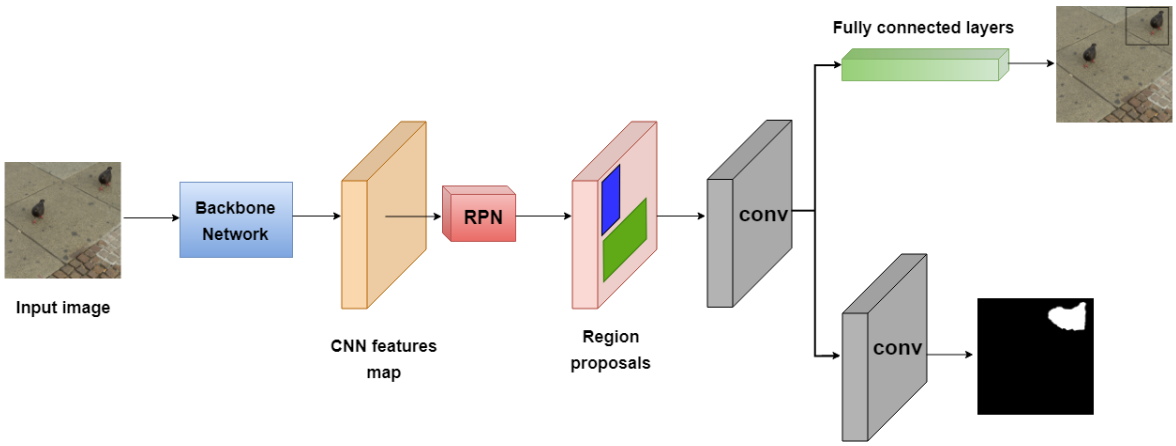


Figure 3.2: The Mask-RCNN framework [65].

3.2 Proposed method

With the advent of new techniques, a forged image can easily be processed to make it more difficult for a human to detect the forgery [89]. In this section, the proposed method is presented and experiments are conducted to evaluate the effectiveness in detecting image forgery. To create a robust model, it is trained on a large dataset that contains different examples of forgery.

3.2.1 Image dataset

In order to generate a robust deep learning model, a sufficiently large dataset is required that contains different kinds of forgery. Currently available image forgery datasets are not large and do not cover a wide range of forgeries. For this reason, a computer generated dataset is used to train the proposed model.

Computer generated forged images have been created using the COCO dataset [86] and a set of random objects with transparent backgrounds [90]. This dataset consists of 80 classes, 80,000 training images and 40,000 validation images. The images have dimensions 480×640 pixels. To create image splicing, an object Y was chosen randomly and pasted into a random image X from the COCO dataset to create the spliced image Z which can be represented by

$$X + Y \Longrightarrow Z, M \tag{3.2}$$

where M represents the output binary mask. The COCO dataset was originally developed for detecting different types of objects such as cars, people and vegetables. However, the proposed network is specifically designed to detect forged regions in images. Thus, the original COCO dataset labels are not used for training. Instead, a new binary mask M is generated for each spliced image with the forged region shown in white and the original region in black.

Fig. 3.3 gives three examples of forged image generation using image splicing. A subset of 21,000 images was selected randomly from the 80,000 training images to create forged images and an equal number of original images was selected for a total of 42,000 training images. Validation images were created similarly by selecting 905 original images and an equal number for

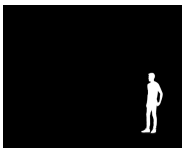
Original image (X)	**Object** (Y)	**Forged image** (Z)	**Binary mask** (M)
			
			
			

Figure 3.3: Examples of forged image generation.

forged images from the validation images in the dataset for a total of 1810 images. Validation images were selected randomly and are different from the images chosen for training. This ensures the model is evaluated on images not used during training for accurate evaluation. To complete the dataset, binary masks were created for each image with the forged pixels labeled by ones and the others labeled by zeros.

A forged image can be post-processed to alter any artifacts resulting from the forgery process. To create a robust model that can detect forgeries in post-processed images, image augmentation techniques were used on the computer generated dataset. These techniques are rotation, shift, shear, and zoom. In rotation, the image is rotated with an angle chosen randomly between 0° and 360°. In shifting, a spatial shift is applied to the image with a width and height shift chosen randomly between 0 and 0.4. After shifting an image, it is cropped to its original dimensions. In shearing, a random transformation intensity between 40° and $-40°$ is used to create a shear matrix. Then this matrix is applied to the image using an affine transformation which results in the upper part of the image shifted to the right and the lower part to the left. In zooming, a zoom is applied to the image by randomly selecting two zoom values from the range $[1, 10]$ for the image width and height. These values are used to create the zoom matrix which is applied to the image using an affine transformation. After zooming, the image is cropped to its original dimensions. Each of the four augmentation techniques is applied to an image during the training process with probability 0.50. Thus, an image has no augmentation or all four techniques applied with probability 0.0625. Adding augmented images expands the image dataset which improves the generalization ability and helps prevent overfitting.

3.2.2 Implementation

A Mask-RCNN model is used with the ResNet model to extract the initial feature map. This is based on an existing implementation by Matterport Inc. released under MIT License [91]. Stochastic gradient descent (SGD) optimization is used to optimize the proposed model with a

momentum of 0.99 and a weight decay of 0.001. Using SGD with momentum helps accelerate the gradient vectors in the correct direction, thus leading to faster convergence [92]. Further, a small weight decay is multiplied by the weights after each update iteration to prevent the weights from growing too large. An initial learning rate of 0.01 is used and this is reduced by 10% if the validation loss does not decrease for 3 consecutive epochs where an epoch denotes an iteration over all training examples. A reducing learning rate helps fine tuning the model to reach its local minimum. The proposed method was implemented using Keras [93] and evaluated on an NVIDIA GeForce GTX 1080 Ti GPU with a memory bandwidth of 11 Gbps.

3.2.3 Initialization

The initialization plays an important role in the convergence speed of a neural network. A good initialization strategy can reduce the feasible parameter space and help the network learn robust features related to the tampering operations rather than complex image content. Initialization can be done using random weights. Examples of random weight initializations are Xavier_normal [84], He_normal [24], Random_normal [94], and Random_uniform [95]. In Xavier_normal, the weights of the network are initialized from a distribution with zero mean and variance

$$\sigma^2 = 1/N_{in}, \tag{3.3}$$

where N_{in} is the number of incoming neurons. He_normal has a similar variance given by

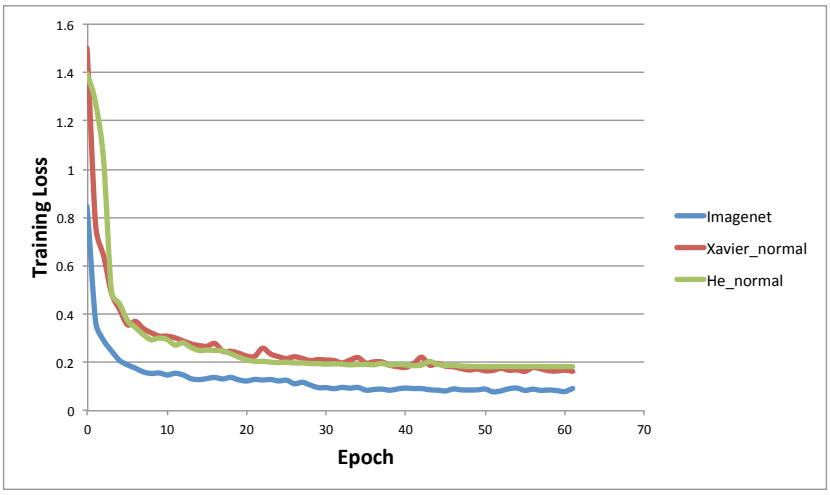
$$\sigma^2 = 2/N_{in}. \tag{3.4}$$

Initialization can also be done using the weights of a network trained on a large dataset. In this case, the weights of a network are used to initialize another network to perform a different task. The ImageNet dataset has been used to pre-train networks such as ResNet [83]. This is because ImageNet contains over 14 million images which belong to more than 20,000 classes. Hence, networks pre-trained on ImageNet can learn a wide variety of features and so can be a good backbone.

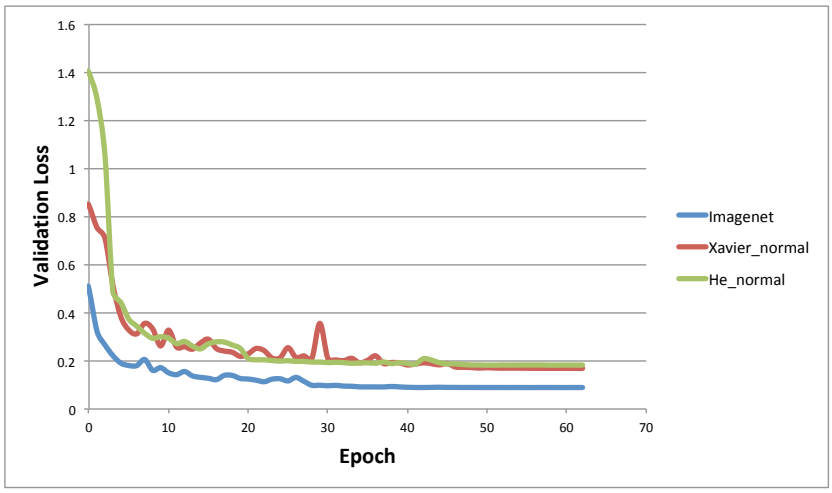
Fig. 3.4 presents the network convergence with initialization using a ResNet network pre-trained on ImageNet [83], Xavier_normal [84] and He_normal [24]. The training and validation losses are defined as $L = L_{cls} + L_{box} + L_{mask}$ where L_{cls} is the classification loss, L_{box} is the bounding-box loss, and L_{mask} is the mask loss [82]. Classification loss is calculated using sparse categorical cross-entropy, bounding box loss is calculated using smooth L_1 loss and mask loss is calculated using binary cross-entropy [82]. These results show that the ResNet network pre-trained on the ImageNet dataset outperforms the other techniques by about 0.1 in both training loss and validation loss. Xavier_normal and He_normal converge to the same minimum value which is about 0.2. The convergence speed is the same for both training and validation for all three methods which indicates that the model can achieve the same accuracy on new examples and overfitting did not occur.

3.2.4 Backbone

Convergence speed can also be improved based on the number of layers in the backbone. The number of layers depends on how many features exist in the dataset and that the network needs



(a)

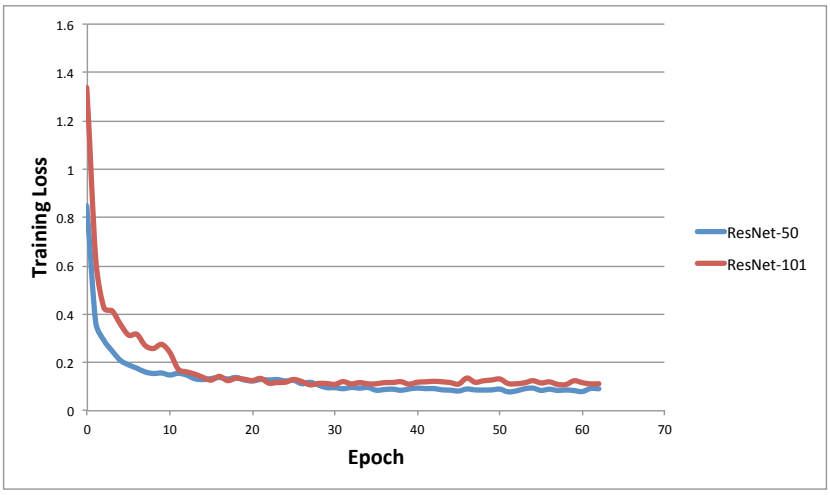


(b)

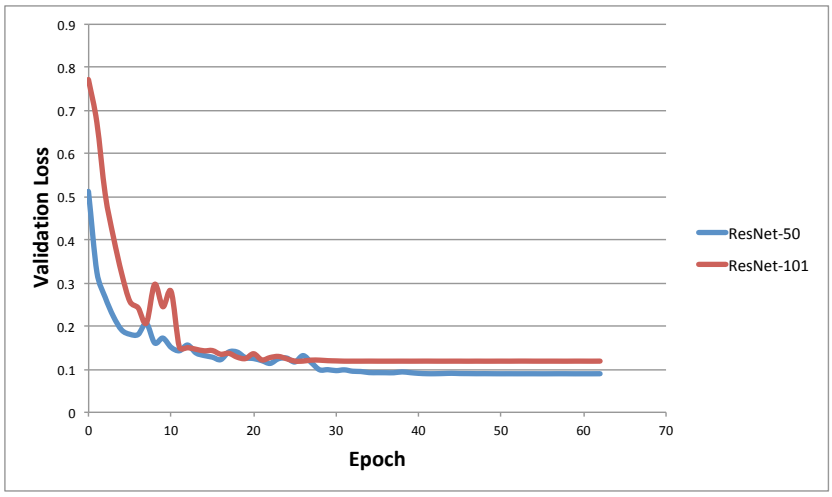
Figure 3.4: Convergence performance for three different initialization methods, (a) training and (b) validation.

to be trained on. For this reason, two different ResNet architectures were considered which are ResNet-50 and ResNet-101 [82]. ResNet-50 consists of 5 stages. Stage 1 is the input stage which consists of one convolutional layer followed by batch normalization and activation functions to generate the initial feature map. Stages 2 to 5 have convolutional blocks and identity blocks. Both blocks consist of convolutional layers followed by batch normalization and activation functions. The convolutional blocks have an extra bridge to add residuals learned in the input layer to the output layer. The ResNet-101 architecture is the same as ResNet-50 except that it has 22 convolutional blocks in stage 4 compared to only 5 in ResNet-50. In total, ResNet-50 has 50 layers while ResNet-101 has 101 layers.

To study the performance of the proposed model with ResNet-50 and ResNet-101 backbones, the convergence is compared. Both models were initialized with ImageNet weights [83] and trained for an equal number of epochs. All layer weights have been used to initialize the



(a)



(b)

Figure 3.5: Convergence performance for ResNet-50 and ResNet-101, (a) training and (b) validation.

proposed model except the output layer. This is because the ImageNet weights are for 1000 output categories while our network output has only two categories: original and forged. Fig. 3.5 shows the training loss and validation loss versus the number of epochs. Although ResNet-101 contains twice the number of convolutional layers as ResNet-50, the training loss with ResNet-50 converges to a lower minimum. The difference in validation loss is even greater. This is because the number of layers in ResNet-50 is sufficient to learn the features in the dataset. On the other hand, ResNet-101 has extra layers that cause overfitting, resulting in an increase in the validation loss. Both models were tuned by reducing the learning rate by 0.005 if the validation loss did not improve for 5 consecutive epochs.

ResNet-FPN has been shown to improve both the accuracy and speed of some tasks [91]. A feature pyramid network (FPN) is a feature extractor designed in a pyramid for the purpose of generating multi-scale feature maps. FPN requires its own backbone network in order to create

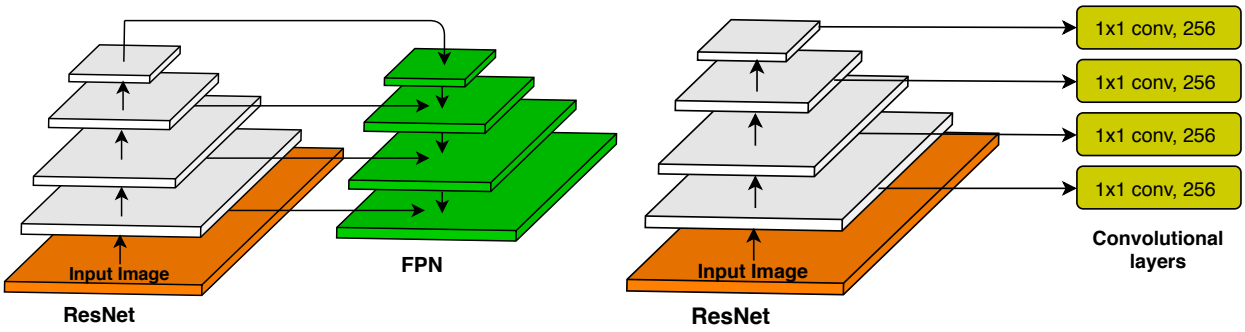


Figure 3.6: The backbone architectures for (a) ResNet-FPN [81] and (b) ResNet-conv.

the feature pyramid.

To detect image forgeries, the proposed model needs to learn features based on the artifacts resulting from forgeries. Adding more layers to the backbone network may increase the convergence speed, but can also lead to overfitting. To choose a backbone suitable for our problem, ResNet-FPN is compared with a simplified version of ResNet called ResNet-conv. In ResNet-conv, the FPN network is replaced by four convolutional layers each with 256 filters, one layer for each of the 4 feature maps created by ResNet. Fig. 3.6 shows the ResNet-FPN and proposed ResNet (ResNet-conv) structures.

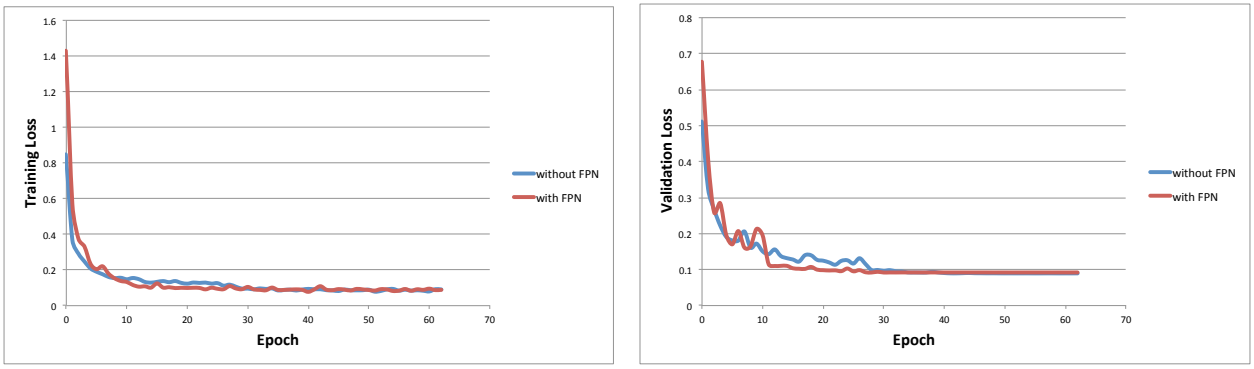
Fig. 3.7 presents the convergence performance with ResNet-FPN and ResNet-conv. For a fair comparison, both networks were trained for the same number of epochs. Although, ResNet-FPN has lower validation and training losses than ResNet-conv up to epoch 30, the losses subsequently converge to the same minimum for both networks. Thus, adding FPN to ResNet does not improve the training which indicates that the feature maps created by ResNet have sufficient features to differentiate between the original and forged regions in an image. This is because the features in a forged region are related to sharp edges, color consistency with surrounding pixels, and differences in contrast and brightness which are basic features that can be learned in backbone training. FPN is a very complex network that may be necessary for more complex problems such as instance segmentation of objects [85], In summary, the results presented here show that an FPN is not required.

Fig. 3.8 presents the receiver operating characteristic (ROC) curve for the proposed network. This illustrates the ability of the network to discriminate between original and forged regions. It was created by plotting the true positive ratio (TPR) against the false positive ratio (FPR) at different threshold values. TPR and FPR are given by

$$TPR = \frac{T_P}{T_P + F_N}, \tag{3.5}$$

$$FPR = \frac{F_P}{T_N + F_P}, \tag{3.6}$$

where T_P is true positive which represents the number of pixels that are correctly detected as forged, F_N is false negative which represents the number of pixels that are falsely detected as original, and T_N is true negative which represents the number of pixels correctly detected as



(a) (b)

Figure 3.7: Convergence performance for ResNet-50 with and without an FPN, (a) training and (b) validation.

original. Thus, TPR and FPR are determined at the pixel level. They can be calculated by comparing the output binary mask with the truth binary mask. Threshold values from 0 to 1 in steps of 0.01 were used to calculate TPR and FPR for the ROC curve. The network performance can also be measured by calculating the area under the curve (AUC) which has a value between 0 and 1. A high AUC means high TPR and low FPR and thus precise model predictions. The proposed network has an AUC value of 0.967, which is excellent.

3.3 Conclusion

In this chapter, a new image forgery detection method based on a Mask-RCNN network was introduced. A new backbone architecture called ResNet-conv was designed to create the initial feature map to train the Mask-RCNN. This new backbone is a simplified version of ResNet-FPN which is obtained by replacing the FPN with convolutional layers. ResNet-conv was shown to have the same convergence speed as ResNet-FPN. Two ResNet architectures were considered, ResNet-50 and ResNet-101. The convergence of ResNet-50 was shown to be faster. This is because features related to forgeries are basic features that can be learned in the early layers of the network. Using additional layers does not improve the detection accuracy but rather slows convergence. The proposed method was trained and evaluated using computer generated forged images. Different augmentation techniques were used to create a model that is robust to several post-processing techniques. The ImageNet, Xavier_normal and He_normal initialization techniques were considered. Initialization with ImageNet weights outperforms other techniques by about 0.1 in both training and validation losses. The network performance was evaluated using the AUC. The proposed network achieved a high AUC of 0.967. Thus, it is able to classify forgeries with high precision.

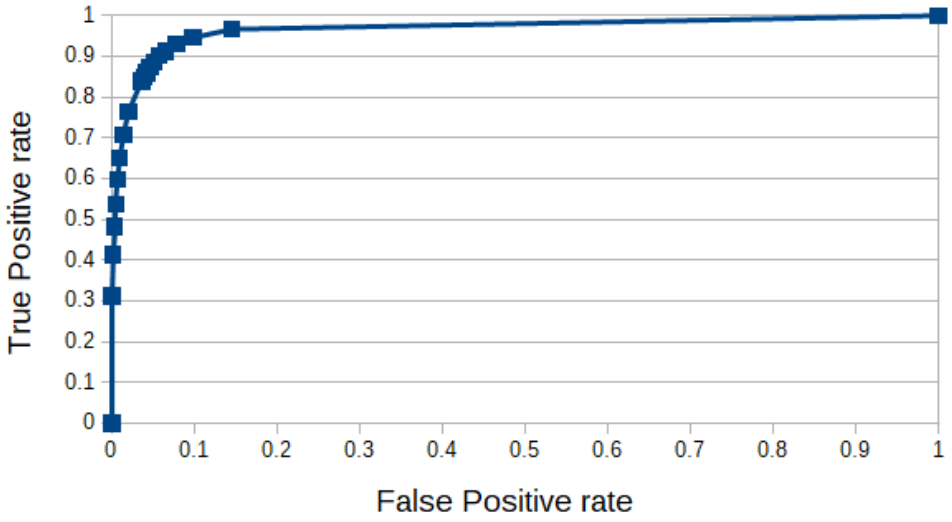


Figure 3.8: The ROC curve for the proposed network.

Chapter 4

Localization and Detection of Copy-Move Forgery in Post-processed Images using U-Net

The rapid growth in digital technologies has significantly increased the popularity of social media services such as Facebook and Instagram. This has led to a huge growth in the number of images distributed every day through the internet. Many of these images have been manipulated such that they may mislead viewers and distort the truth. The distribution of such images can have very serious consequences, so determining the authenticity of digital images is of great importance.

Several methods to detect copy-move forgeries have been developed. Most of these are block based, key-point based or transform based methods. In [96], a singular value decomposition (SVD) technique was proposed to extract image features. First the image is partitioned into square blocks and then SVD is applied to each block to obtain the feature vectors. These vectors are sorted lexicographically and matched by calculating the distances between them. The main advantage of this approach is the ability to represent an image by feature vectors of small dimension. However, SVD requires many matrix multiplications to extract the feature vectors which results in high computational complexity.

Although block based methods can detect forgeries with high accuracy, they often fail when post-processing such as rotation, scaling, and JPEG compression is employed on the forged images. Further, the computational complexity of these methods is high due to the image partitioning required before feature extraction. Conversely, key-point based methods do no not need this partitioning. Instead, features are extracted directly from the images using techniques such as the scale invariant feature transform (SIFT) and speeded up robust features (SURF) [42]. These techniques extract features based on entropy. Then, the features are matched using approaches such as clustering or Euclidean distance. If two regions have similar features, then one is considered to be forged. Key-point based methods are faster than block based methods, but they may fail to detect flat copied regions, i.e. regions with a highly uniform texture. They also have a higher false positive rate (FPR) than block based methods especially with highly textured areas [97].

While block based and key-point based methods can provide excellent accuracy in image forgery detection, this comes at the cost of high computational complexity. Thus, developing

low complexity techniques that can detect image forgeries without sacrificing accuracy are critical to the field of image forensics. In this chapter, a new image forgery detection technique is proposed which is based on a U-Net model. A computer-generated dataset is used to train this model. This dataset is created using the COCO dataset [86] and a set of random objects [90]. To train a robust model against image post-processing, data augmentation is applied to images in the dataset. Six different post-processing techniques are considered, namely brightness change (BC), contrast adjustment (CA), color reduction (CR), image blurring (IB), JPEG compression (JC), and noise addition (NA). The proposed method is compared with several well-known techniques in terms of accuracy, time complexity and image post-processing using the well-known CoMoFoD dataset [36].

The remainder of this chapter is organized as follows. Section 2 provides a review of existing deep learning approaches for image forgery detection. Section 3 discusses augmentation techniques, the proposed U-Net framework and the algorithm used to generate the augmented dataset. The experimental results, comparisons and analysis are given in Section 4. Finally, some conclusions are drawn in Section 5.

4.1 Deep learning

Deep learning networks such as deep autoencoder [70] and convolutional neural network (CNN) have achieved considerable success in many fields such as classification [28], semantic segmentation [82], and speech recognition [72]. More recently, deep learning has been employed in the field of image forensics. In [32], a deep learning approach was introduced which is based on a CNN model to detect image splicing. First, an image is divided into non-overlapping 128×128 patches. Second, image features are extracted from image patches using convolutional layers, and then these features are concatenated and pooled using a max pooling layer. Finally, a support vector machine (SVM) classifier is trained on the extracted features. Another CNN based technique was introduced in [35]. This method uses ResNet-50 as a backbone to convert the input image into a feature map. The Feature Pyramid Network (FPN) is replaced by a set of convolutional layers to generate multiscale feature maps. Then these maps are input to a Mask-RCNN to locate forgeries.

In [98], a ringed residual U-Net (RRU-Net) for image splicing detection was introduced. RRU-Net combines residual propagation and residual feedback to learn features related to image splicing. Residual propagation acts like the recall mechanism of the human brain and is used to solve the degradation problem in the deep network. Residual feedback consolidates the input feature information to increase the differences in the image attributes of the original and forged regions. In [99], a dense U-Net approach was proposed to detect image forgeries. This network uses the cross-layer intersection technique to determine the multiscale contribution of shallow-layer features of the predicted segmentation. It was evaluated using several well known datasets but tested only against JPEG post-processing.

Deep learning techniques have also been used to detect copy-move forgeries. In [100], a deep learning architecture called BusterNet was introduced to detect these forgery. A BusterNet network consists of two branches to locate potential manipulated regions via visual artifacts and copy-move forgery via visual similarities, respectively. These features are combined in a fusion layer to localize the forged region and the source. A data-driven strategy to solve the

problem of small datasets was introduced in [101]. This approach is based on two branches. The first consists of a generator to generate forged images and a discriminator to detect if an image is forged or real, while the second is a similarity detection network to extract regions with similar features. The feature vectors from these branches are combined in a fusion layer and an SVM classifier is trained using these features to locate forged regions.

Many techniques have addressed the problem of detecting image forgeries, however several problems remain open such as image post-processing, overfitting, and computational complexity. Post-processing is used to slightly change the appearance of an image. It is often applied to forged images to hide artifacts resulting from the forgery operation. Overfitting is a challenge whenever deep learning is employed. This occurs when a model achieves a good fit with the training data but fails to achieve the same results on new, unseen data. Computational complexity is a significant problem with most non deep learning approaches. While the complexity of training deep learning networks can be high, this need only be done once and the resulting model can provide results quickly.

4.2 Proposed method

This section presents the proposed method and the experiments to evaluate the performance in detecting image forgeries.

4.2.1 Network architecture

U-Net is based on a deep end-to-end encoder-decoder architecture that is used for segmentation. It has an encoder to convert the input image into a feature map and a decoder for upsampling the feature maps resulting from downsampling to obtain a segmentation map.

In this chapter, a new U-Net architecture is presented which is specifically designed to detect copy-move forgeries. It consists of a convolutional layer to convert the input image into feature maps, 3 downsampling stages (encoder) followed by 3 upsampling stages (decoder) and two convolutional layers to generate the output mask. Figure 4.1 shows the architecture of the proposed network. First, an image of size 480×640 is converted into feature maps using a convolutional layer with a receptive field of size 3×3 and 64 filters. Then, these feature maps are downsampled using 3 downsampling stages. Each downsampling stage consists of two convolutional layers with receptive fields of sizes 3×3 and 2×2, and 64 and 32 filters, respectively. Different receptive field sizes are used to extract features related to forgeries at different scales. A rectified linear unit (ReLu) is applied to the extracted feature maps to ensure only useful information is retained. Finally, a 2×2 max pooling layer with stride 2 is used to downsample the output feature maps by a factor of 2. Max pooling is used to improve convergence by discarding less important features while keeping those related to forgeries. This also reduces the computational complexity of the convolution operations by reducing the size of the feature maps. Each upsampling stage consists of a layer with a receptive field of size 2×2 and stride 2 for upsampling the input feature maps by 2. This is followed by two convolutional layers with a receptive field of size 3×3 and 64 and 32 filters, respectively. Then, a ReLu function is applied to the extracted feature maps. Finally, a convolutional layer with a receptive field of size 3×3 and stride

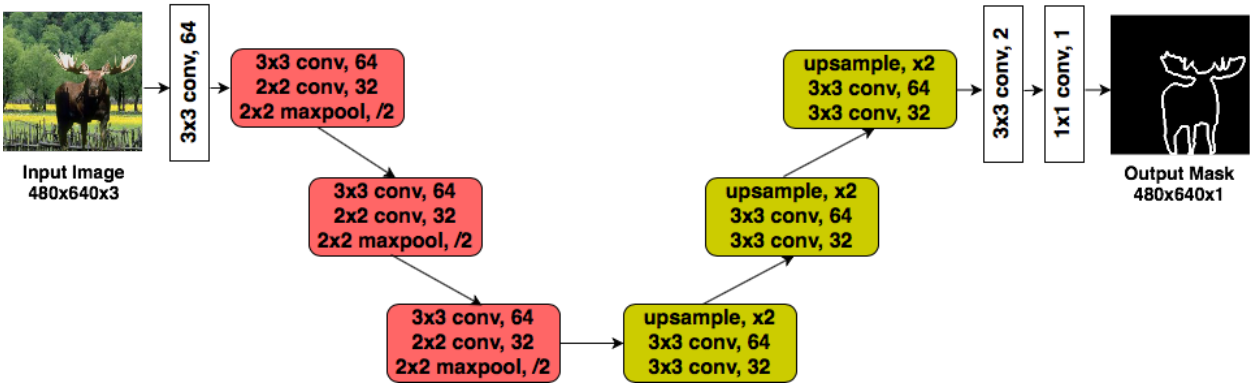


Figure 4.1: The proposed encoder-decoder network framework.

2 followed by a convolutional layer with a receptive field of size 1×1 and stride 1 are used to generate the output mask.

4.2.2 Training dataset

It is well-known that deep learning techniques can suffer from overfitting of the dataset used for training. One means of avoiding this problem is to use a sufficiently large dataset that contains different kinds of forgery. However, currently available datasets are not large and do not cover a wide range of forgeries. In addition, it is time consuming to manually annotate a large number of images. Several techniques have been introduced to generate image forgery datasets by simulating the forgery process [80]. However, most use a window of a specific shape such as a rectangle to copy and paste forged regions. This results in a network that overfits the generated dataset and can only detect forgeries with a similar shape. A more robust model can be obtained by training on a dataset that has a greater variety of forged images.

In this chapter, forged images are created using the COCO dataset [86] and a set of random objects [90]. These random objects have transparent backgrounds to avoid generating forged regions with specific boundaries such as a rectangle or circle. The COCO dataset was originally developed for detecting objects such as people and animals and consists of 80 classes, 80,000 training images and 40,000 validation images. This enables us to generate different kinds of forged images and to avoid overfitting certain types of objects. All images have dimensions 480×640 pixels. To create a forged image, an object is chosen randomly and pasted into a random image from the COCO dataset.

The proposed network is specifically designed to detect forged regions in images so new labels are created to represent the forged regions. Most techniques label all forged region as forged. This results in a network that learns features related to the objects such as color, brightness, and shape which are not forgery features. However, the key features related to forgeries are the artifacts that appear at the boundaries of the forged regions such as sharp edges. Hence, forgery labels are given to those pixels that lie across these boundaries. This produces a model that has learned features related to the actual forgeries. Figure 4.2 gives three examples of forged images from the generated dataset.

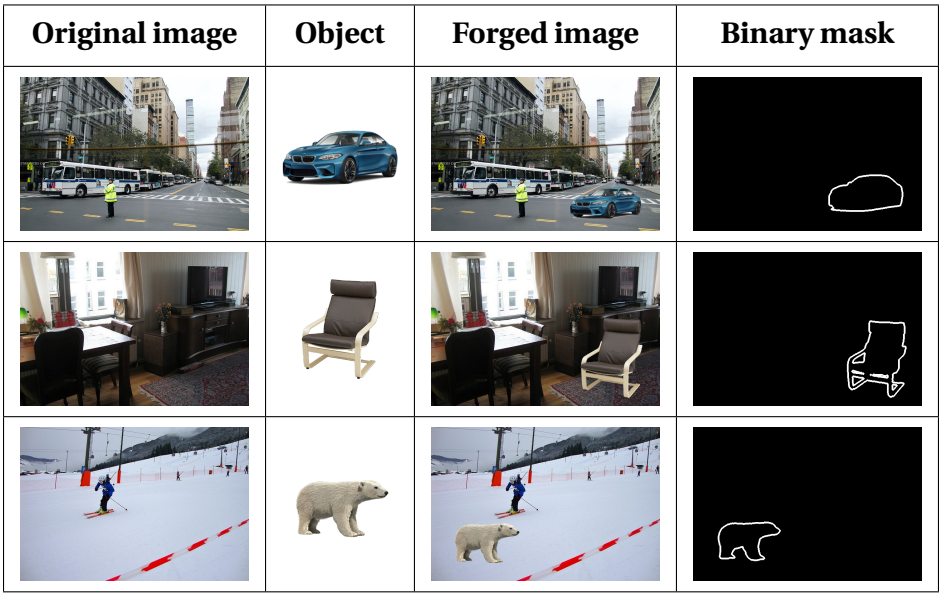


Figure 4.2: Examples of the forged images generated.

4.2.3 Evaluation dataset

In this chapter, the CoMoFoD image dataset is used to evaluate the robustness of the proposed network for image forgery detection. This dataset consists of 200 images of size 512×512 classified into five categories, namely translation, rotation, scaling, distortion, and a combination of these four techniques. Each category consists of 40 images and all images are processed using 6 post-processing techniques. These techniques are brightness change (BC), contrast adjustment (CA), color reduction (CR), image blurring (IB), noise addition (NA) and JPEG compression (JC). For BC, the color intensity values are mapped to a new range defined by upper and lower bounds. Any intensity values beyond this range are set to the corresponding minimum or maximum value. For CA, the color intensity values are mapped to a new interval bounded by upper and lower bounds. The bounds used for BC and CA are $(0.1, 0.95)$, $(0.1, 0.9)$ and $(0.1, 0.8)$ denoted by BC1, BC2 and BC3, respectively. With CR, the color intensity values are quantized from $(0, 256)$ to a smaller range. These ranges are $(0, 32)$, $(0, 64)$ and $(0, 128)$ denoted by CR1, CR2 and CR3, respectively. For IB, noise is added to the intensity values by applying averaging filters. These filters have sizes $[3 \times 3, 5 \times 5$, and $7 \times 7]$, denoted by IB1, IB2 and IB3, respectively. For NA, Gaussian noise is added to the intensity values to create what is known as Gaussian blur [102]. This noise has mean $\mu = 0$ and variance $\sigma = [0.009, 0.005, 0.0005]$ denoted by NA1, NA2 and NA3, respectively. For JC, images are compressed with quality factors $[20, 30, 40, 50, 60, 70, 80, 90, 100]$ denoted by JC1 to JC9, respectively.

4.2.4 Data augmentation

Data augmentation is used to increase the size of a dataset without having to annotate more images. This can be done by post-processing images to create altered versions which are added to the dataset. An advantage of this approach is that the resulting model is better able to detect forgeries in post-processed images. Four augmentation techniques are used here, namely rota-

tion, shift, shear, and zoom. With rotation, the image is rotated by a random angle between 0° and 360°. Shifting applies a spatial shift to the image with a width and height chosen randomly between 0 and 0.4. After shifting, the image is cropped to its original dimensions. In shearing, a random transformation intensity between 40° and −40° is used to create a shear matrix. This matrix is applied to the image using an affine transformation which results in the upper part of the image shifted to the right and the lower part to the left. With zoom, two values are randomly chosen from the range [1,10] corresponding to the image width and height. These values are used to create a zoom matrix which is applied to the image using an affine transformation. After zooming, the image is cropped to its original dimensions. Each of the four augmentation techniques is applied to an input image during the training process with probability 0.50. Thus, an image has no augmentation or all four techniques applied with probability 0.0625.

4.2.5 Implementation

A deep learning network learns by means of a loss function which evaluates how well the network can predict the given data. In the proposed method, binary cross entropy is used as the loss function during training [103]

$$L = -y\log(p(y)) - (1-y)\log(1-p(y)), \tag{4.1}$$

where y is the ground truth label (0 or 1) and $p(y)$ is the prediction probability. Stochastic gradient descent (SGD) optimization is used to minimize L with momentum 0.9 and weight decay 0.001. SGD with momentum accelerates moving the gradient vectors in the correct direction, leading to faster convergence [92]. Further, the weights are multiplied by a small decay value after each update to prevent them from growing too large. An initial learning rate of 0.01 is used and this is reduced by 10% if the validation loss does not decrease for 3 consecutive epochs where an epoch denotes an iteration over all training data. In addition, a reducing learning rate is employed to help the model reach a local minimum. The proposed method was implemented using Keras [93] and evaluated on an NVIDIA GeForce GTX 1080 Ti GPU with a memory bandwidth of 11 Gbps.

4.2.6 Initialization

The initialization plays an important role in the convergence of a neural network. This can be done using the weights of a network trained on a very large dataset such as ImageNet [83]. Such networks have been shown to be able to learn a wide variety of features and thus are a good backbone. However, this can lead to overfitting for tasks that have different kinds of features. Initialization can also be done using random weights such as Xavier_normal [84], He_normal [24], Random_normal [94] and Random_uniform [95].

Here, Xavier_normal [84] is used to initialize the proposed U-Net model. With Xavier_normal, the weights of the network are initialized from a distribution with zero mean and variance

$$\sigma^2 = 1/N_{in}, \tag{4.2}$$

where N_{in} is the number of incoming neurons to a layer. This leads to bounded layer weights given by

$$-\sqrt{(6/n_i^{in} + n_i^{out})} \leq \omega_i \leq \sqrt{(6/n_i^{in} + n_i^{out})}, \tag{4.3}$$

Method	Precision (%)	Recall (%)	F1 score (%)	Computation time (seconds per image)	Number of images detected as forged (out of 200)
[100]	57.3	49.4	49.3	**0.62**	117
[63]	39.9	47.6	41.8	1.78	93
[104]	45.8	34.4	37.4	5.11	90
[105]	36.3	40.4	31.1	0.95	53
[106] (AlexNet)	51.1	**83.5**	63.3	-	-
[106] (VGG)	49.7	72.0	58.8	-	-
[8]	61.6	71.0	65.9	-	-
Proposed	**72.2**	72.6	**71.7**	1.67	**181**

Table 4.1: Performance results for the CoMoFoD dataset with no post-processing.

where ω_i is a weight on layer i, n_i^{in} is the number of incoming network connections to layer i, and n_i^{out} is the number of outgoing network connections from that layer. With Xavier_normal, the variance of the input data is kept the same which helps keep the weights from exploding or vanishing [84].

4.3 Performance results

This section presents performance results obtained using the CoMoFoD dataset. The metrics employed for evaluation purposes are

$$\text{Precision} = \frac{\text{Forged region} \cap \text{Detected region}}{\text{Detected region}} = \frac{T_P}{T_P + F_P}, \tag{4.4}$$

$$\text{Recall} = \frac{\text{Forged region} \cap \text{Detected region}}{\text{Forged region}} = \frac{T_P}{T_P + F_N}, \tag{4.5}$$

$$\text{F1 score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{4.6}$$

where T_P is the true positive rate which is the number of forged pixels detected as forged, F_P is the false positive rate which is the number of pixels incorrectly detected as forged, and F_N is the false negative rate which is the number of forged pixels incorrectly detected as original. Precision is the percentage of pixels successfully detected as forged while recall is the percentage of forged pixels successfully detected by the network. F1 score is the weighted average of precision and recall so it takes both false positives and false negatives into account.

To validate our results, we compare them with seven recently proposed deep learning based methods which are described below. BusterNet [100] is a two branch deep neural network that has been used to detect and localize image forgeries. First branch consists of an end-to-end encoder-decoder architecture designed to learn features related to copy- move forgery. Second branch is a selfcorrelation module designed to extract similar features between image blocks and localize real and forged regions. The dense-field based method given in [63] can be used to extract features from each pixel in the input image. The similarity between features is calculated and a set of heuristics is used to improve the detection performance. The block based method in [104] used Zernike moments to extract features from each block in the input image.

Deep matching and validation network (DMVN) [105] is a two-branch deep neural network that extracts features from donor and query images using a CNN feature extractor. Both feature maps are fed into a deep dense matching module to compare the two images. The predicted masks are fed into a visual consistency validator module to focus on the segmented areas in both images and similar regions are extracted using a siamese-like module. The domain adversarial neural network (DANN) given in [106] has three components: i) feature extractor, ii) source classified, and iii) domain classifier. First, features are extracted by using two different pretrained networks, AlexNet and VGG. Second, a domain classification (DC) network is used to predict the domain of the input image. Finally, a source classification (SC) network is used to predict the label for source samples. The keypoint-based method given in [8] used speeded up robust features (SURF) to localize similar features resulting from copy-move forgery.

Table 4.1 gives the precision, recall, F1 score, computation time and number of detected images for the proposed and other methods using the CoMoFoD dataset (without any post-processing). An image is correctly detected if the F1 score is greater than 0.5. The best results are indicated in bold. These results show that the proposed method outperforms the other methods in terms of precision, F1 score and number of detected images. The recall of the method in [106] is better than the proposed method by only 1.9%. However, the precision and F1 score of the proposed method is better than the AlexNet based method in [106] by 21.1% and 8.4% respectively. The computation time of the methods in [100, 105] is lower than the proposed method, but only by 0.72 and 1.05 seconds, respectively. Further, the proposed method outperforms the methods in [100, 105] in all other metrics. Note that the computation time and number of images detected for the methods in [106] and [8] were not available.

To evaluate the effectiveness with post-processing, the F1 score was calculated for the entire CoMoFoD dataset. Figure 4.3 presents the F1 scores for the proposed and other methods. This figure consists of 6 sub-figures representing the F1 scores for each of the 6 post-processing techniques in CoMoFoD dataset. These results show that the proposed method outperforms the other methods for all but the IB post-processing technique. The results for the proposed method are as follows. For BC, increasing the level of post-processing from BC1 to BC3 only reduced the F1 score by 11%. For CA, the values were also reduced by 11% from CA1 to CA3. For CR, the values are very similar and only changed by 0.01% from CR1 to CR3. For IB, increasing the level of post-processing from IB1 to IB3 reduced the F1 score by 14%. For NA, increasing the level of post-processing from NA3 to NA1 reduced the F1 score by 19%. For JC, increasing the level of post-processing from JC9 to JC1 reduced the F1 score by 12%. Table 4.2 shows the F1 scores for the proposed and other methods. This table contains the F1 scores for each of the 6 post-processing techniques in the dataset.

Figure 4.4 presents the receiver operating characteristic (ROC) curves for the proposed method. The false positive rate (FPR) is given on the x-axis and the true positive rate (TPR) on the y-axis for different threshold values. The threshold values range from 0 to 1 in steps of 0.01. The area under the curve (AUC) indicates how well U-Net can discriminate between original and forged regions and has a value between 0 and 1. The higher the AUC, the better the model can distinguish between the regions. For BC, increasing the level of post-processing from BC1 to BC3 reduced the AUC by only 2.6%. For CA, increasing the level of post-processing from CA1 to CA3 reduced the AUC by 3.1%. For CR, increasing the level of post-processing from CR1 to CR3 reduced the AUC by only 0.1%. For IB, increasing the level of post-processing from IB1 to IB3 reduced the AUC by 5.2%. For NA, increasing the level of post-processing from NA3 to

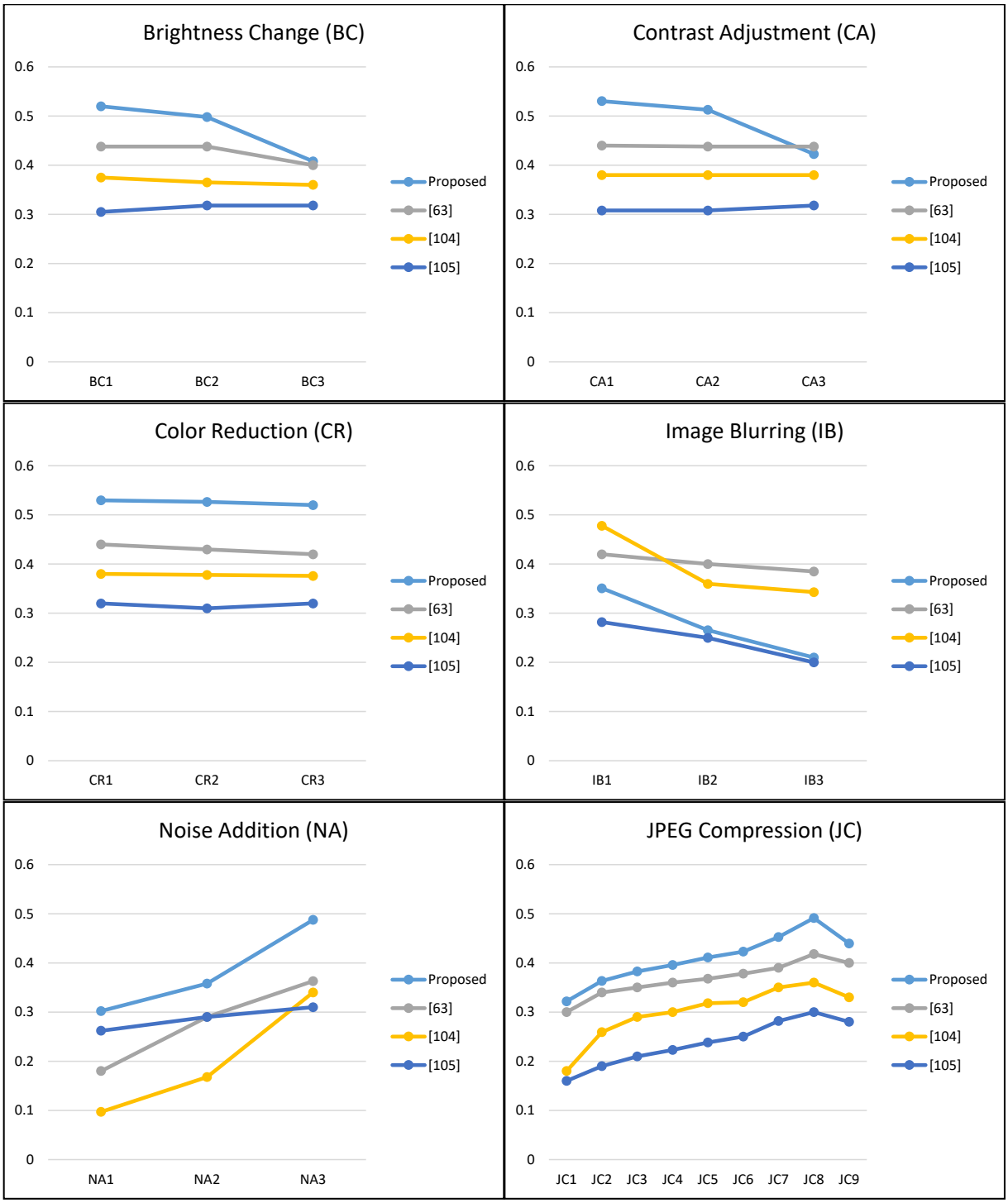


Figure 4.3: F1 scores for the CoMoFoD dataset with different post-processing techniques.

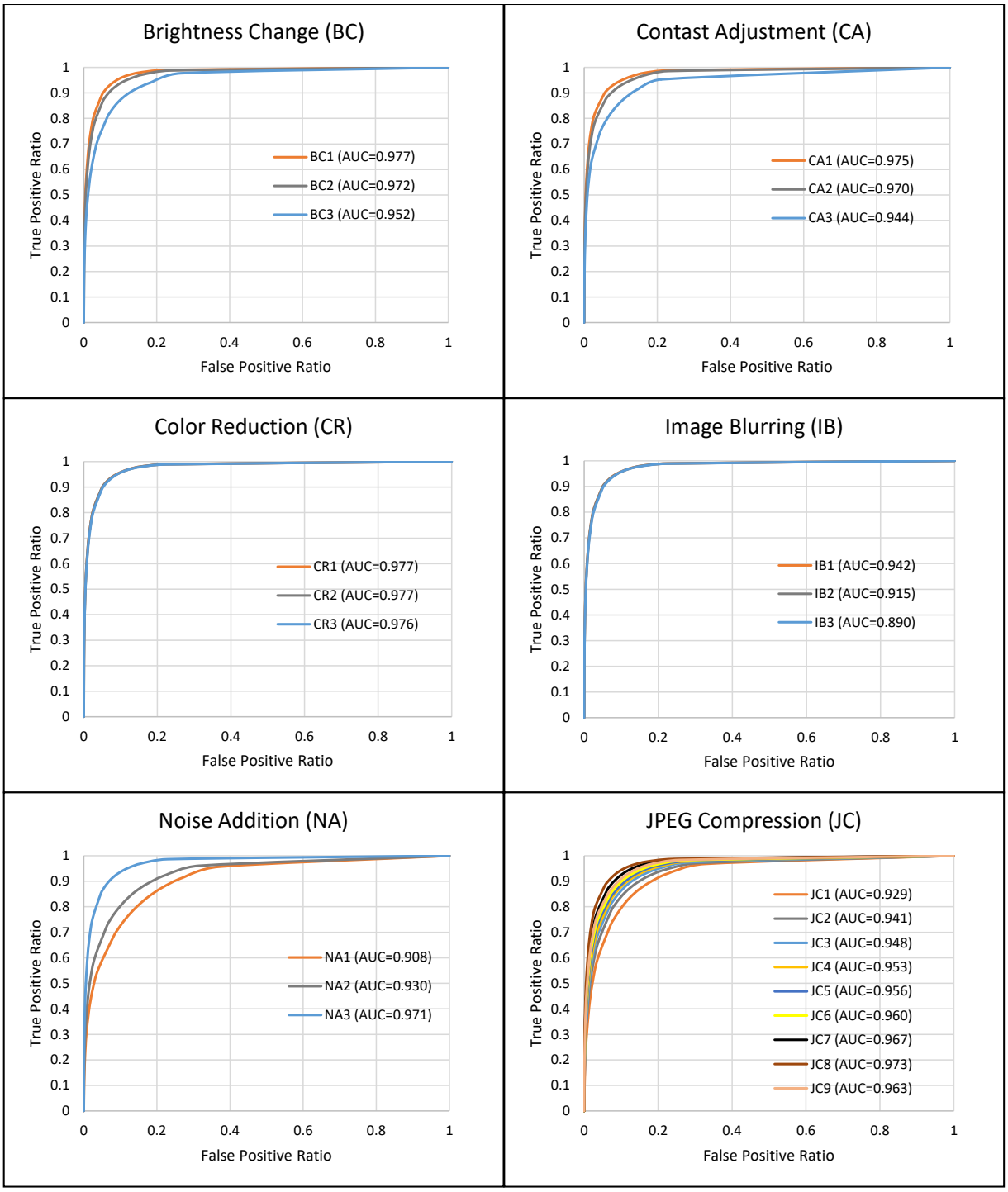


Figure 4.4: ROC curves for the proposed method using the CoMoFoD dataset with different post-processing techniques.

Post-processing technique	[63]	[104]	[105]	Proposed
BC1	43.8	37.5	30.5	**52.0**
BC2	43.8	36.5	31.8	**49.8**
BC3	40.0	36.0	31.8	**40.8**
CA1	44.0	38.0	30.8	**53.0**
CA2	43.8	38.0	30.8	**51.3**
CA3	**43.8**	38.0	31.8	42.3
CR1	44.0	38.0	32.0	**53.0**
CR2	43.0	37.8	31.0	**52.6**
CA3	42.0	37.6	32.0	**52.0**
IB1	42.0	**47.8**	28.2	35.1
IB2	**40.0**	36.0	25.0	26.6
IB3	**38.5**	34.3	20.0	21.0
NA1	18.0	9.7	26.2	**30.2**
NA2	29.0	16.8	29.0	**35.8**
NA3	36.3	34.0	31.0	**48.7**
JC1	30.0	18.0	16.0	**32.2**
JC2	34.0	25.9	19.0	**36.3**
JC3	35.0	29.0	21.0	**38.3**
JC4	36.0	30.0	22.3	**39.6**
JC5	36.8	31.8	23.8	**41.1**
JC6	37.8	32.0	25.0	**42.3**
JC7	39.0	35.0	28.2	**45.3**
JC8	41.8	36.0	30.0	**30.0**
JC9	40.0	33.0	28.0	**43.9**

Table 4.2: F1 score results for the CoMoFoD dataset with different post-processing techniques.

NA1 reduced the AUC by 6.3%. For JC, increasing the level of post-processing from JC1 to JC9 reduced the AUC by only 3.4%. Overall, the proposed method was able to effectively detect forgeries in post-processed images and was robust to increases in the level of post-processing. Table 4.3 summarizes the AUC results for the proposed method with different post-processing techniques.

4.4 Conclusion

In this chapter, a new image forgery detection method based on deep learning was introduced. This method employs an encoder-decoder network that encodes an input image into a feature map and decodes the feature map into a binary mask with the forged regions segmented. Synthetic data was created to train the network and evaluation was done using the well-known CoMoFoD dataset. Unlike other techniques in the literature, the generated data only masks the boundaries of the forged areas. This helps the network learn only features related to the forged areas such as sharp edges. Four different augmentation techniques, namely rotation, shear, shift and zoom, were randomly applied to the training images. This augmentation helps in cre-

Post-processing technique	AUC (%)
BC1	97.7
BC2	97.2
BC3	95.2
CA1	97.5
CA2	97.0
CA3	94.4
CR1	97.7
CR2	97.7
CA3	97.6
IB1	94.2
IB2	91.5
IB3	89.0
NA1	90.8
NA2	93.0
NA3	97.1
JC1	97.1
JC2	92.9
JC3	94.1
JC4	94.8
JC5	95.3
JC6	95.6
JC7	96.0
JC8	96.7
JC9	97.3

Table 4.3: AUC results for the proposed method using the CoMoFoD dataset with different post-processing techniques.

ating a robust network that can detect forgeries in post-processed images. It also reduces the probability of overfitting.

Compared to recent techniques in the literature, the proposed method achieved the highest precision, recall, and F1 scores with 72.2%, 72.6% and 71.7%, respectively. It also achieved the highest F1 score for 5 of the 6 post-processing techniques in the CoMoFoD dataset. The computation time for the proposed method was 1.67 s per image which is only 1.05 s higher than the lowest computation time. In addition, the number of forged images detected was 181 out of 200 (90.5%), which is very high for this kind of forgery. The other methods had a success rate of less than 60%. ROC analysis was conducted to evaluate the effectiveness of the proposed network in discriminating between original and forged regions. The AUC values were greater than 90% for all post-processing techniques except IB3 which was 89%.

Chapter 5

Image Splicing and Copy-Move Forgery Detection using PADNET

The internet has made digital images and video a main source of information. They can be presented as evidence in a court of law, used in news items, or form part of a medical record. Several tools have recently become available at little or no cost to easily manipulate images. These tools are becoming more sophisticated which makes forgeries harder to detect. Further, manipulated images can be rapidly and easily distributed through the internet to mislead viewers and distort the truth. Thus, there is an urgent need for quick and effective image validation.

There are three main image forgery techniques, copy-move forgery, image splicing forgery, and image retouching [4]. In copy-move forgery, part of an image is copied and pasted into another part of the same image to conceal or change information. In this case, the source and target are the same so the photometric characteristics are similar. Image splicing refers to copying part of one image and pasting it into another image. This results in different photometric characteristics such as brightness, color, and contrast. Image retouching refers to the process of altering an image to slightly change the appearance of a subject [5]. Several post-processing techniques such as noise addition, resizing, and blurring can be employed to disguise forgeries.

The remainder of this chapter is organized as follows. Section 5.1 presents a review of existing approaches to image forgery detection. The PADNET architecture is introduced in Section 5.2. The implementation details and loss function are given in Section 5.3. Section 5.4 presents the algorithm used to generate the training dataset, augmentation techniques and the evaluation dataset. Section 5.5 presents the experimental results, comparisons and analysis. Finally, some concluding remarks are given in Section 5.6.

5.1 Related work

In this section we provide a review of existing approaches for detecting copy-move and image splicing forgeries.

5.1.1 Copy-move forgery

There is a considerable amount of literature on copy-move forgery detection. Most approaches are either block-based or keypoint-based. Typically, block-based methods partition an image into overlapping blocks and features are extracted from the blocks using a suitable technique [10]. Conversely, keypoint-based methods require no image partitioning prior to feature extraction.

In [47], a method was introduced to detect copy-move forgeries using DCT coefficients as eigenvalues for each image block. Then, the Euclidean distance between the eigenvectors was calculated and lexicographically sorted to reduce the false positive rate. Two image blocks are considered to be duplicates if their distance is below a threshold. This method is robust to Gaussian blurring and noise addition, but not to other post-processing techniques. In [96], singular vector decomposition (SVD) was used to extract features from image blocks. Then, the distances between the feature vectors are calculated and sorted lexicographically to identify matching blocks. The main advantage of using SVD is the ability to decompose the image features to a smaller dimension. However, SVD requires many matrix-vector multiplications which increases the computational complexity.

Block-based methods can detect forgeries with high accuracy. However, most existing approaches fail to detect forgeries after post-processing operations such as rotation and scaling have been performed. Another drawback is that that these methods have high computational complexity. Conversely, keypoint-based approaches require no image partitioning prior to feature extraction. Instead, features are extracted directly from the image based on identifying and selecting high-entropy image regions. Techniques such as the scale invariant feature transform (SIFT) and speeded up robust features (SURF) can be used to extract keypoint features from the image [42]. The main advantage of keypoint-based methods is lower computational complexity compared to block-based methods. However, a major drawback is that they often fail to locate homogeneous forged regions.

Recently, deep learning has been the subject of considerable research in the field of digital forensics. In [100], a deep learning architecture called BusterNet was introduced to detect copy-move forgeries. This network consists of two branches, one for locating potential manipulated regions via visual artifacts and the other to locate copy-move forgeries via visual similarities. Then, these features are combined in a fusion layer to localize the forged regions and the source. In [110], a data-driven strategy was presented to solve the problem of small datasets. First, a generator is used to generate fake images and then a discriminator to detect if an image is fake or real. Then a self correlation convolutional neural network (CNN) is used to extract regions with similar features. Finally, feature vectors are combined in a fusion layer and an SVM classifier trained using these features to locate forged regions.

5.1.2 Image splicing forgery

Several approaches have been proposed for detecting image splicing forgeries [35]. A method for image splicing detection based on a grey level run length matrix (GLRLM) was introduced in [66]. The GLRLM is used to extract features from forged and original images. Then, these features are used to train an SVM for classification. In [67], statistical features extracted from the run length and image edges were used to localize forgeries. The method in [111] improved

the detection accuracy obtained in [67] from 69% to 75% in less time. This was achieved using a detection algorithm based on approximate run lengths.

Deep learning techniques have also been used to detect image splicing forgeries [35, 79]. In [79], a shallow CNN (SCNN) architecture was designed to detect forgeries in low resolution images. First, the image is converted from RGB space to YCrCb space. Then, the network is trained on CrCb channels only to exclude the illumination information. Finally, forged regions are detected by discriminating changes in chroma and saturation. A deep learning approach was introduced in [35] to detect image splicing forgeries. A modified FPN architecture is used to extract the feature maps and then a Mask-RCNN is trained using these maps to localize forged regions.

Most techniques have been developed to detect only one kind of forgery. For example, some methods have been designed to detect copy-move forgeries [10] while others focus on detecting image splicing [35] or forgeries in images that have been subjected to lossy compression such as JPEG [77]. However, there is no technique which can detect multiple types of forgeries. Current approaches have limitations so cases exist where they perform poorly [63]. For this reason, the goal here is to present a solution which can detect multiple kinds of forgery.

In this chapter, a novel deep learning architecture is proposed for detecting image forgeries. This method addresses the two major limitations of state-of-the-art image forgery detection algorithms

(i) it can detect multiple kinds of forgery, and

(ii) it can detect forgeries in low resolution images and images that have been subjected to post-processing.

The proposed architecture, denoted PADNET, has two stages. The first stage is a feature pyramid network (FPN) [81] which is used for feature aggregation from multiscale levels. FPN consists of a bottom-up and a top-down networks. The bottom-up network is used to extract features map as a usual convolutional network. However, the spatial resolution decreases as we go up. Thus, a top-down network is used to extract higher resolution maps from a semantic rich map. In the second stage, the feature maps are used to train a DeepUNet architecture [112] designed to learn discriminative features considering both high-level global features and low-level local features. The combination of low-level and high-level features is necessary to detect features from multiple kinds of forgery such as copy-move and image splicing. For copy-move forgery detection, PADNET is trained on only the boundaries of the forged areas while for image splicing detection, it is trained on the area around the forged regions. A class imbalance can occur for copy-move forgery detection due to a low number of pixels in the forgery class compared to the original class. To solve this problem, weighted binary cross-entropy is employed to balance the two classes. PADNET is evaluated on the well known CoMoFoD dataset for copy-move forgery detection and CASIA1 for image splicing forgery detection. The results obtained show that PADNET outperforms state-of-the-art copy-move and image splicing forgery detection algorithms.

5.2 Proposed method

This section presents the PADNET architecture and the experiments conducted to evaluate the performance in detecting copy-move and image splicing forgeries.

5.2.1 Network architecture

PADNET is a deep end-to-end architecture specifically designed to detect multiple kinds of image forgeries. It consists of two networks (1) ResNet-50 FPN and (2) DeepUNet. In this section, the architecture of both networks is discussed in detail.

5.2.1.1 Feature pyramid network (FPN)

A feature pyramid network (FPN) is one of the most popular networks for object detection. It detects objects at different scales in the corresponding pyramid levels. An FPN consists of a bottom-up and a top-down networks. ResNet-50 is used as a bottom-up network to extract features at different scales. It consists of 5 stages. Each stage consists of a convolutional block and identity block. Both blocks consist of 3 convolutional layers followed by batch normalization and activation functions. The convolutional blocks have an extra bridge to add residuals learned in the input layer to the output layer. A total of 5 multiscale bottom-up feature maps are extracted from the 5 stages of ResNet-50. The bottom feature maps have high resolution but the semantic value is not high enough to be used for detection. Only upper feature maps are used and therefore perform much worse for small objects. Thus, a top-down network is used to reconstruct high resolution maps from semantic rich maps. The recontructed maps are semantic strong, however the locations of objects are not precise after all the downsampling and upsampling. Thus, lateral connections are used between bottom-up and top-down layers. These connections help the detector to predict the location better. Finally, the extracted feature maps are used to train the DeepUNet. The FPN architecture is shown on the left side of Fig. 5.1.

5.2.1.2 DeepUNet

A DeepUNet has a symmetric architecture similar to a UNet [112]. The proposed DeepUNet architecture is specifically designed to learn discriminative features considering both high-level global features and low-level local features. It consists of downsampling and upsampling paths. The downsampling path (encoder) consists of 3 repeated downsampling blocks which are connected to the corresponding upsampling blocks. There are also short connections between successive downsampling or upsampling blocks. These connections reduce the possibility of losing important features when going deeper into the network. Each downsampling block consists of two convolutional layers with receptive fields of sizes 3×3 and 2×2 and 64 and 128 filters, respectively. Then, a rectified linear unit (ReLu) is applied to the feature maps to ensure only useful information is retained. Finally, a max pooling layer is used to reduce the computational complexity by discarding less important features. The upsampling blocks are similar to the downsampling blocks. Each contains two convolutional layers with receptive fields of size 3×3 and 64 and 128 filters, respectively. However, it has an upsampling layer instead of a max pooling layer to upsample the encoded feature maps back to their original size. The DeepUNet

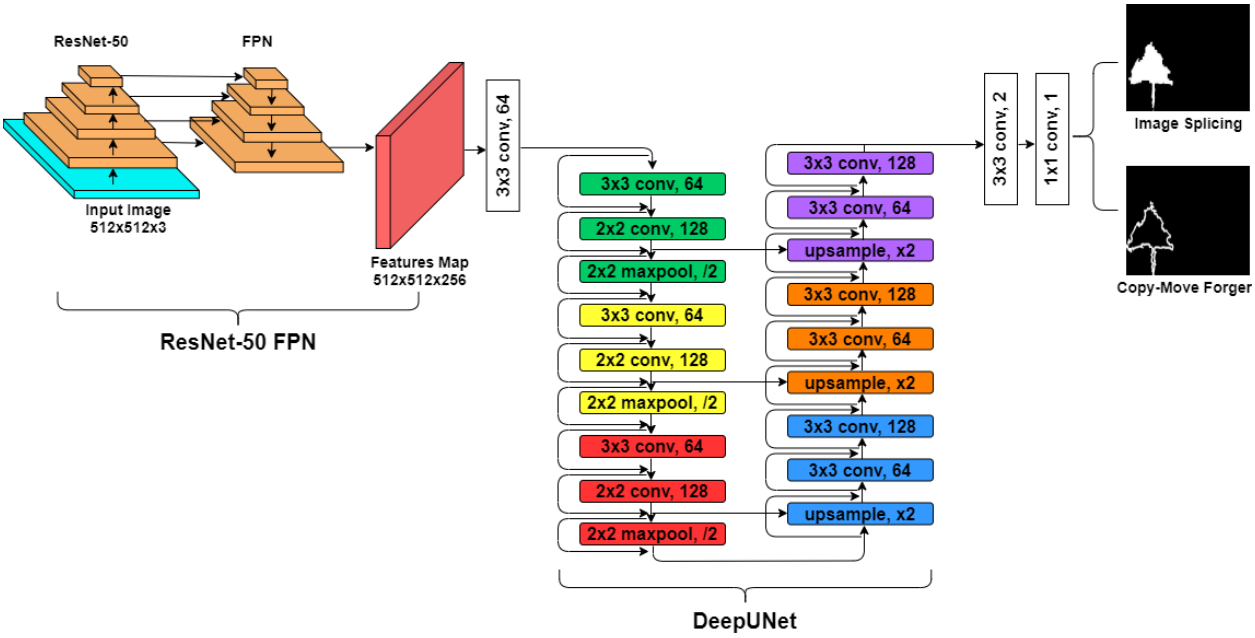


Figure 5.1: The PADNET architecture.

architecture is shown on the right side of Fig. 5.1. The blocks are represented by different colors for identification.

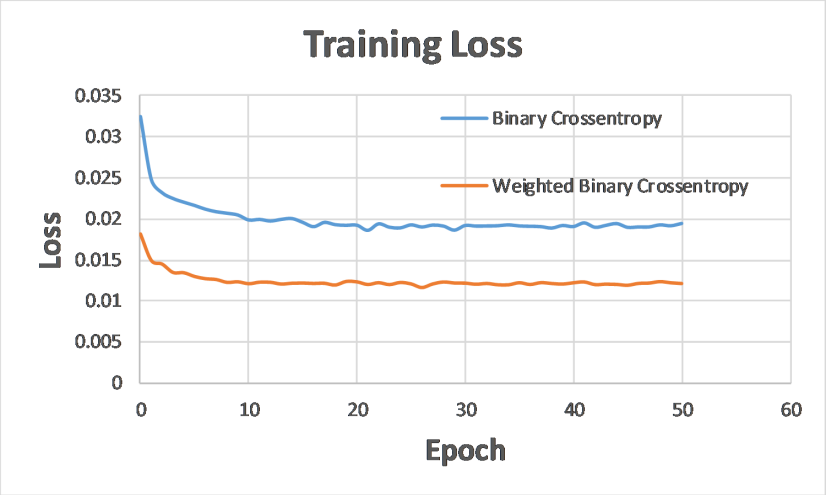
5.3 Implementation

The goal of any deep learning method is to reduce the difference between the predicted output and truth which is known as the loss function. This can be done using an optimizer which guides the model to minimize the loss function. Stochastic gradient descent (SGD) optimization with momentum 0.9 and weight decay 0.001 is used here. SGD with momentum accelerates moving the gradient vectors in the correct direction to improve convergence [92]. Further, a small weight decay is used after each update to prevent the weights from growing too large. An initial learning rate of 0.01 is used to update the weights based on the convergence speed. This learning rate is reduced by 10% if the validation loss does not decrease for three consecutive epochs where an epoch is an iteration over all training data. A reducing learning rate is employed to help the model reach the minimum loss. The proposed method was implemented using Keras [93] and evaluated on an NVIDIA GeForce GTX 1080 Ti GPU with a memory bandwidth of 11 Gbps.

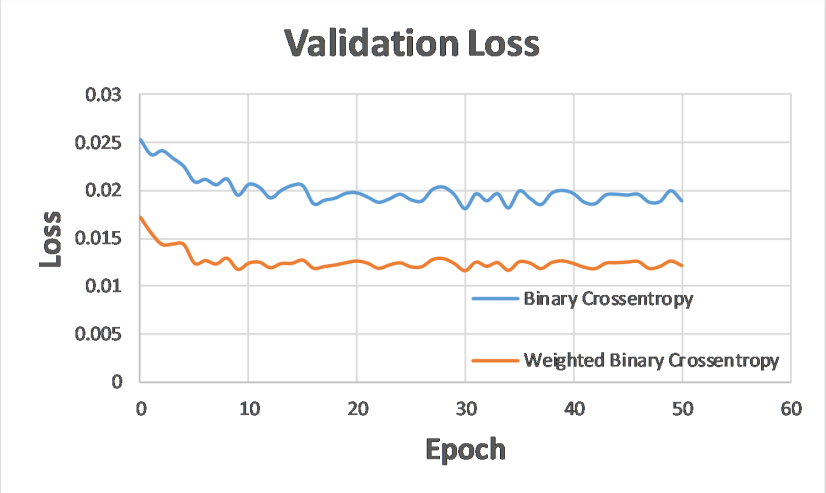
In the proposed method, binary cross-entropy is used as the loss function during training and is given by [113]

$$L = -y\log(p(y)) - (1-y)\log(1-p(y)), \tag{5.1}$$

where y is the ground truth label (0 or 1) and $p(y)$ is the prediction probability. For copy-move forgery detection, the forged region is copied from the same image. Thus, features such as color, brightness, and contrast are consistent with the remainder of the image. As a result, PADNET is trained on only the boundaries of the forged area. However, this results in a class imbalance



(a)



(b)

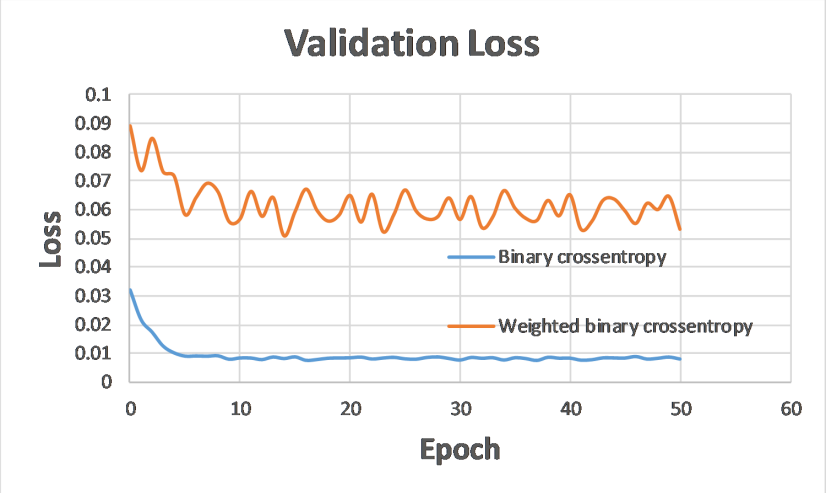
Figure 5.2: PADNET convergence with the binary cross-entropy and weighted binary cross-entropy loss functions for copy-move forgery detection, (a) training and (b) validation.

when detecting forged and original classes. Class imbalance occurs when there is a significant difference in the number of samples among classes in a training set. This results in a model that has learned features from one class more than the others. In this case, the model is more likely to predict all images as original (the majority). To solve this problem, a modified binary cross-entropy function is employed called weighted binary cross-entropy. This function gives higher weight to the forged class and less weight to the original class and is given by [114]

$$L_{\omega} = -\omega_0 y \log(p(y)) - \omega_1 (1-y) \log(1-p(y)), \tag{5.2}$$

where y is the ground truth label (0 or 1), $p(y)$ is the prediction probability, ω_0 is the weight for the forgery class and ω_1 is the weight for the original class. The weights used for copy-move forgery detection are 0.7 for the forgery class and 0.3 for the original class. This ensures that the model provides a low penalty for misclassifying original pixels and a high penalty for misclas-

(a)



(b)

Figure 5.3: PADNET convergence with the binary cross-entropy and weighted binary cross-entropy loss functions for image splicing detection, (a) training and (b) validation.

sifying forged pixels. PADNET convergence with the binary cross-entropy and weighted binary cross-entropy loss functions for copy-move forgery detection is given in Fig. 5.2. These results show that PADNET converges to a lower minimum with weighted binary cross-entropy for both training and validation. In both cases, the loss is 0.012 with weighted binary cross-entropy compared to 0.019 with binary cross-entropy. As a result, weighted binary cross-entropy is more efficient as a loss function for copy-move forgery.

For image splicing detection, the forged region is copied from a different image. This results in inconsistencies in the forged region features such as color, brightness, and contrast. Therefore, PADNET is trained on the entire forged region so there is no class imbalance. PADNET convergence with the binary cross-entropy and weighted binary cross-entropy loss functions for image splicing detection is given in Fig. 5.3. These results show that PADNET converges to a lower minimum with binary cross-entropy for both training and validation. In both cases,

the loss is below 0.01 with binary cross-entropy compared to 0.058 with weighted binary cross-entropy. As a result, binary cross-entropy is more efficient as a loss function for image splicing.

5.4 Image dataset

The datasets employed play a key role in creating robust deep learning models. Thus, it is important to have a wide variety of input data to ensure the generalization ability of the model. In this section, the algorithm used to create the datasets is presented.

5.4.1 Training dataset

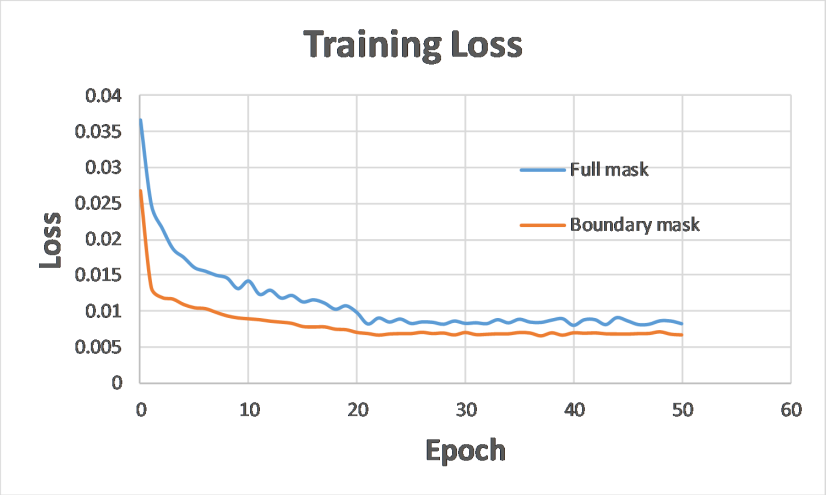
A common problem with many deep learning techniques is insufficient data. This can result in high variability in model predictions, which means that the model fits the training data well but performs poorly with new data. Most existing image forgery datasets focus on a specific kind of forgery. In addition, these datasets are small and do not cover a wide range of forgeries. For this reason, an algorithm is developed to create a computer generated dataset. This dataset contains both copy-move and image splicing forgeries and is used to train the proposed model.

The forged image dataset is created from the COCO dataset [86] and a set of random objects [90]. The COCO dataset consists of 80,000 training images and 40,000 validation images. The random objects have transparent backgrounds to avoid generating forged regions with specific boundaries such as a circle or rectangle. To create image splicing forgeries, an object was chosen randomly and pasted into a random image from the COCO dataset. To create copy-move forgeries, forged images were generated by randomly copying a region of random shape from an image and pasting it into a different part of that image. The COCO dataset was originally developed for object classification such as people, animals, and cars. However, the proposed method is designed to detect forged regions in images. Thus, the original COCO dataset labels are not used for training. Instead, binary masks were generated for each forged image with the forged region in white and the original region in black.

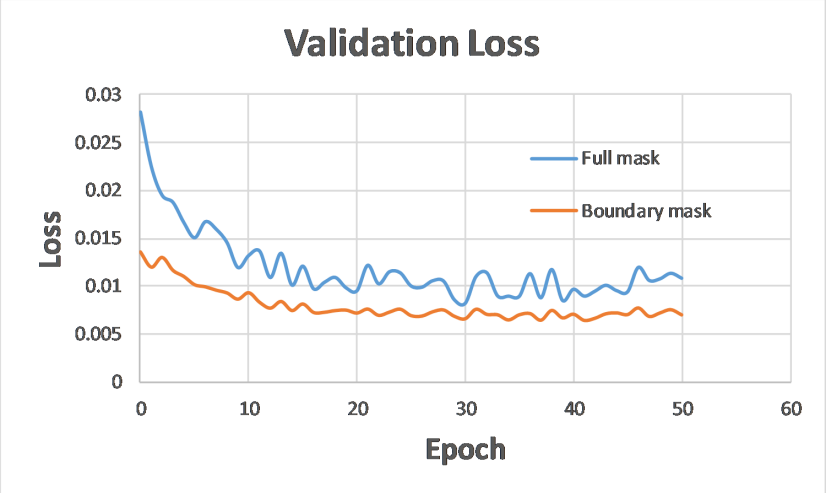
A subset of 21,000 images were randomly selected from the 80,000 training images for image splicing and an equal number for copy-move forgery for a total of 42,000 training images. Similarly, a subset of 905 images was randomly selected from the 40,000 validation images for image splicing and an equal number for copy-move forgery for a total of 1810 validation images. The images chosen for validation are different from those chosen for training. This avoids overfitting as the training data will only be statistically similar to what the model is tested on. Finally, a binary mask is created for each forged image with the forged pixels labeled as ones and the others labeled as zeros.

For copy-move forgeries, the forged region is copied from the same image. Thus, features such as color, brightness, and contrast are consistent with the rest of the image. The key features are the artifacts that appear at the boundaries of the forged regions such as sharp edges. Fig. 5.4 presents the PADNET convergence when training on boundary and full masks for copy-move forgery detection. This shows that convergence is better when training is done using boundary masks. Further, both the training and validation losses converge to a minimum.

With image splicing, the forged region is copied from a different image. Hence, features such as color, brightness, and contrast are different from the rest of the image. For this reason,



(a)

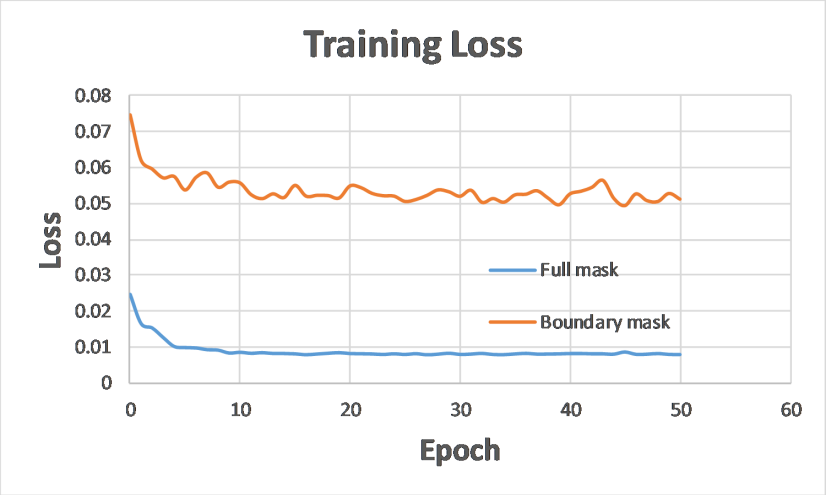


(b)

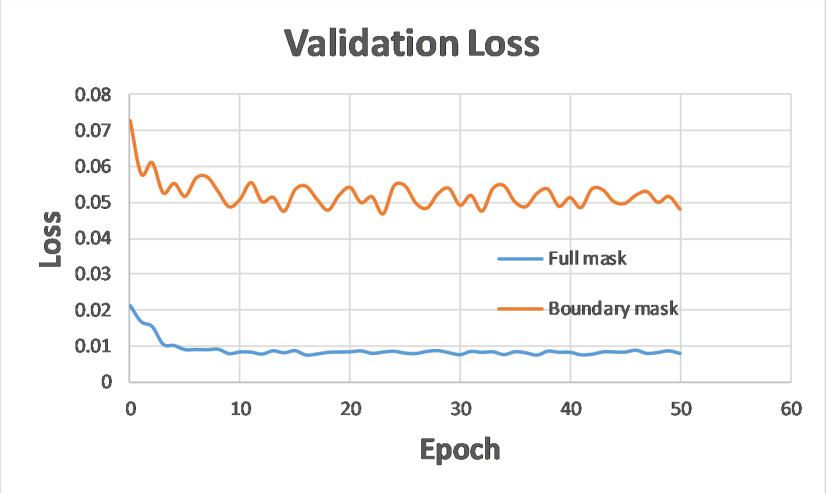
Figure 5.4: PADNET convergence when training on boundary labels and full mask labels for copy-move forgery detection, (a) training and (b) validation.

these are key features for forgery detection. Artifacts that appear at the boundaries of the forged regions are also important features to detect image splicing. Fig. 5.5 presents PADNET convergence when training on boundary and full masks for image splicing detection. This shows that convergence is better when training is done using full masks. The large gap between loss values indicates that features such as color, brightness, and contrast obtained by training using full masks is important.

The above results indicate that the choice of a binary mask should depend on the kind of forgery. For copy-move forgery, forgery labels are given to pixels that lie across the boundary of the forged region while for image splicing, these labels are given to the entire forged region. Fig. 5.6 shows examples of the generated copy-move and image splicing forgeries.



(a)



(b)

Figure 5.5: PADNET convergence when training on boundary labels and full mask labels for image splicing detection, (a) training and (b) validation.

5.4.1.1 Data augmentation

In forged images, artifacts such as sharp edges can be present prior to the forgery process, but inconsistencies in these artifacts can be evidence of tampering. To hide these artifacts, post-processing techniques such as brightness change, color reduction and JPEG compression can be applied to an image. Thus, a more robust model can be obtained by training on a dataset that has a variety of forged images with different post-processing. To achieve this goal, image augmentation techniques were used on the computer generated dataset. These techniques are shift, shear, zoom, and rotation. In shifting, the image is spatially shifted by a width and height chosen randomly between 0 and 0.4. Then, the image is cropped to its original dimensions. In shearing, a shear matrix is created by using a random transformation intensity between 40° and −40°. Then, an affine transformation is applied to the image using this matrix. This results

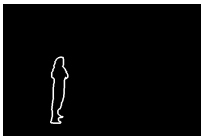
Type of Forgery	Original Image	Forged Image	Binary Mask
Copy-move			
Splicing			

Figure 5.6: Examples of the forged images generated.

in the upper part of the image shifted to the right and the lower part to the left. In zooming, a zoom matrix is created by selecting two zoom values from the range [1,10]. Then, an affine transformation is applied to the image using this matrix. Finally, the image is cropped to its original dimensions. In rotation, an angle between 0° and 360° is randomly chosen and used to rotate the image. These augmentation techniques are randomly selected during training so that each is applied to an image with probability 0.50. Thus, an image has no augmentation or all four techniques applied with probability 0.0625. Adding augmented images to the dataset increases the diversity of the data used for model training. This improves the generalization ability of the model and helps prevent overfitting.

5.4.2 Evaluation dataset

In this chapter, the CoMoFoD and CASIA1 datasets are used to evaluate the proposed method. These datasets are described below.

5.4.2.1 CoMoFoD dataset

The CoMoFoD image dataset is used to evaluate the copy-move forgery detection performance [36]. This dataset contains 10400 images of size 512×512 divided into five groups according to the changes done to the images, namely translation, rotation, scaling, distortion and a combination of the four. These images have been processed using six post-processing techniques. These techniques are brightness change (BC), contrast adjustment (CA), color reduction (CR), image blurring (IB), noise addition (NA) and JPEG compression (JC). For BC, the brightness of the image is changed by mapping the intensity values of the original image to a new range between lower and upper bounds. Any value that lies below the lower bound or above the upper bound is saturated to the corresponding bound. For CA, the contrast of the image is changed by mapping the intensity values of the original image to a new interval bounded by upper and lower bounds. The bounds used for BC and CA are $(0.1, 0.95)$, $(0.1, 0.9)$ and $(0.1, 0.8)$ denoted BC1, BC2 and BC3 for BC and CA1, CA2 and CA3, for CA, respectively. For CR, the color intensity values for each color channel of the original image are quantized from $(0, 256)$ to a smaller range. These ranges are $(0, 32)$, $(0, 64)$ and $(0, 128)$ denoted by CR1, CR2 and CR3, respectively. For IB, the original image is blurred by convolving the image with averaging filters of sizes 3×3,

5×5 and 7×7, denoted by IB1, IB2 and IB3, respectively. For NA, Gaussian noise with zero mean and different variances is added to the image. The variances are 0.009, 0.005 and 0.0005 denoted by NA1, NA2 and NA3, respectively.

5.4.2.2 CASIA1 dataset

The CASIA1 image dataset is used to evaluate the proposed method for image splicing detection [115]. This dataset consists of 1721 JPEG images of size 384×256. These images are divided into two subsets: authentic and forged. Only the forged set is considered here for evaluation and it contains 921 images.

In the CASIA1 dataset, spliced images are generated by copying part of a randomly chosen image from the authentic images and pasting this into another authentic image. These copied image regions may be processed with a post-processing technique such as scaling or rotation before pasting. The regions can be of any size and can be pasted into an image of the same or different texture.

5.5 Performance results

In this section, the copy-move and image splicing forgery detection performance is evaluated. The metrics employed are

$$\text{Precision} = \frac{\text{Forged region} \cap \text{Detected region}}{\text{Detected region}} = \frac{T_P}{T_P + F_P}, \tag{5.3}$$

$$\text{Recall} = \frac{\text{Forged region} \cap \text{Detected region}}{\text{Forged region}} = \frac{T_P}{T_P + F_N}, \tag{5.4}$$

$$\text{F1 score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{5.5}$$

where T_P is the true positive rate which is the number of forged pixels detected as forged, F_P is the false positive rate which is the number of pixels incorrectly detected as forged, and F_N is the false negative rate which is the number of forged pixels incorrectly detected as original. Precision is the ratio of correctly detected forged pixels to the total predicted forged pixels while recall is the ratio of correctly detected forged pixels to the total number of forged pixels. F1 score is the weighted average of precision and recall so it takes both false positives and false negatives into account.

The performance of PADNET is compared with several well-known techniques to evaluate its effectiveness in detecting forgeries. For copy-move forgery detection, we compare our results with the four recently proposed methods given in [63, 100, 104, 105]. In [100], a two branch deep neural network called BusterNet was used to detect and localize image forgeries. The first branch consists of an encoder-decoder architecture trained to detect copy-move forgeries. The second branch has a self-correlation module designed to extract similar features from image blocks and discriminate between real and forged regions. In [63], a dense-field based method was introduced to extract features from each pixel in an input image. The similarity between these features was then calculated and a set of heuristics used to improve the detection performance. The block-based method in [104] uses Zernike moments to extract features from each

Method	Precision (%)	Recall (%)	F1 score (%)
[100]	**57.3**	49.4	53.1
[63]	39.9	47.6	43.4
[104]	45.8	34.4	39.3
[105]	36.3	40.4	38.2
Proposed	50.4	**91.9**	**65.1**

Table 5.1: Performance results for the CoMoFoD dataset with no post-processing for copy-move forgery.

block in an input image. In [105], a deep matching and validation network (DMVN) was developed to detect image forgeries. It is a two-branch deep neural network that extracts features from donor and query images using a CNN feature extractor. The extracted feature maps are compared using a deep dense matching module. Then, the predicted masks are fed into a visual consistency validation module to focus on the segmented areas in both images. Finally, similar regions are extracted using a Siamese-like module.

For image splicing detection, we compare our results with the five recently proposed methods given in [116–120]. In [116], a passive approach was introduced to detect image splicing based on block artifacts resulting from JPEG compression. In [117], features were extracted based on demosaicking artifacts at a local level. A deep learning approach was presented in [118] to detect image splicing. This method employs a coarse CNN and a refined CNN to extract the differences between an image and splicing regions from patch descriptors at different scales. A fully convolutional network (FCN) was used in [119] for image splicing detection. Pretrained networks such as AlexNet, VGG, and GoogleNet were used to initialize the FCN and then it was fine-tuned for the segmentation task. DeepLabv3 was used in [120] to detect image splicing. This network uses parallel atrous (dilated) convolutions in parallel to capture multi-scale context using multiple atrous rates.

5.5.1 CoMoFoD dataset performance

The performance of the proposed method is evaluated using the well known CoMoFoD dataset for copy-move forgery detection. Table 5.1 gives the precision, recall and F1 score for the proposed and other methods without any post-processing. The best results are indicated in bold. The precision of the method in [100] is better than the proposed method by only 6.9%. However, the recall and F1 score of the proposed method is better than the method in [100] by 42.5% and 12% respectively. In copy-move forgery, the forgery labels are only given to the boundaries of the forged area. Thus, if few pixels are falsely detected as forged, it will have a big impact on the false positive ratio due to the low number of forged pixels compared to the whole image. This results in a high sensitivity of the precision and recall values. For this reason, an average of the precision and recall is better in evaluting the model performance which is represented by the value of the F1 score. The results show that the proposed method outperforms the other methods in terms of recall and F1 score.

To evaluate the effectiveness with post-processing, the F1 score was calculated for the entire dataset. Figure 5.7 presents the F1 scores for the proposed and four other methods. The pro-

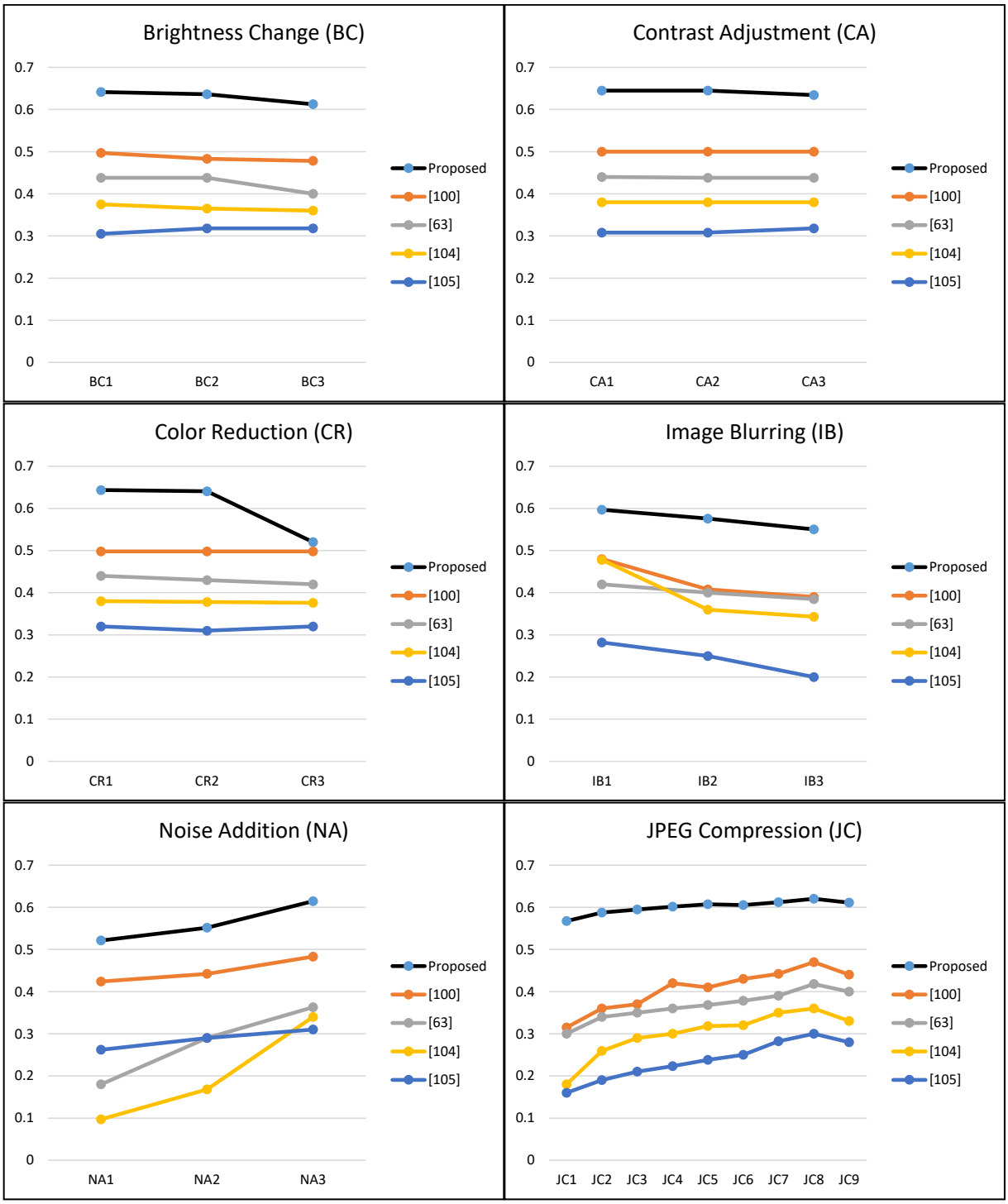


Figure 5.7: F1 scores for the CoMoFoD dataset with different post-processing techniques.

Post-processing technique	[100]	[63]	[104]	[105]	Proposed
BC1	49.7	43.8	37.5	30.5	**64.1**
BC2	48.3	43.8	36.5	31.8	**63.6**
BC3	47.8	40.0	36.0	31.8	**61.3**
CA1	50.0	44.0	38.0	30.8	**64.5**
CA2	50.0	43.8	38.0	30.8	**64.5**
CA3	50.0	43.8	38.0	31.8	**63.4**
CR1	49.8	44.0	38.0	32.0	**64.4**
CR2	49.8	43.0	37.8	31.0	**64.1**
CA3	49.8	42.0	37.6	32.0	**52.0**
IB1	48.0	42.0	47.8	28.2	**59.7**
IB2	40.8	40.0	36.0	25.0	**57.6**
IB3	39.0	38.5	34.3	20.0	**55.0**
NA1	42.4	18.0	9.7	26.2	**52.1**
NA2	44.2	29.0	16.8	29.0	**55.2**
NA3	48.3	36.3	34.0	31.0	**61.5**
JC1	31.5	30.0	18.0	16.0	**56.7**
JC2	36.0	34.0	25.9	19.0	**58.7**
JC3	37.0	35.0	29.0	21.0	**59.5**
JC4	42.0	36.0	30.0	22.3	**60.1**
JC5	41.0	36.8	31.8	23.8	**60.7**
JC6	43.0	37.8	32.0	25.0	**60.5**
JC7	44.2	39.0	35.0	28.2	**61.2**
JC8	47.0	41.8	36.0	30.0	**62.0**
JC9	44.0	40.0	33.0	28.0	**61.1**

Table 5.2: F1 score results for the CoMoFoD dataset with different post-processing techniques for copy-move forgery.

posed method outperforms the other methods for all post-processing techniques. This figure consists of 6 sub-figures which give the F1 scores for each of the 6 post-processing techniques in the CoMoFoD dataset. For BC, increasing the level of post-processing from BC1 to BC3 only reduced the F1 score by 3%. For CA, the score was reduced by only 1% from CA1 to CA3. For CR, the score was reduced by 12% from CR1 to CR3. For IB, increasing the level of post-processing from IB1 to IB3 reduced the F1 score by 5%. For NA, increasing the level of post-processing from NA3 to NA1 reduced the F1 score by 9%. For JC, increasing the level of postprocessing from JC9 to JC1 reduced the F1 score by 4%.

The method in [100] achieved the highest F1 score among other techniques in all post-processing techniques. However, the proposed method achieved a higher F1 score by 14.4 %, 15.3 % and 13.5 % from BC1 to BC3. For CA, the proposed method achieved a higher F1 score by 14.5 %, 14.5 % and 13.4 % from CA1 to CA3. For CR, the proposed method achieved a higher F1 score by 14.6 %, 14.2 % and 2.2 % from CR1 to CR3. For IB, the proposed method achieved a higher F1 score by 11.7 %, 16.8 % and 16 % from IB1 to IB3. For NA, the proposed method achieved a higher F1 score by 9.7 %, 11.0 % and 13.2 % from NA1 to NA3. For JC, the proposed method achieved a higher F1 score by 25.2 %, 22.7 %, 22.5.4 %, 18.1 %, 19.7 %, 17.5 %, 17 %, 15 % and 17.1 % from JC1 to JC9. Table 5.2 shows the F1 scores for the proposed and other methods. This table contains the F1 scores for each of the 6 post-processing techniques in the dataset.

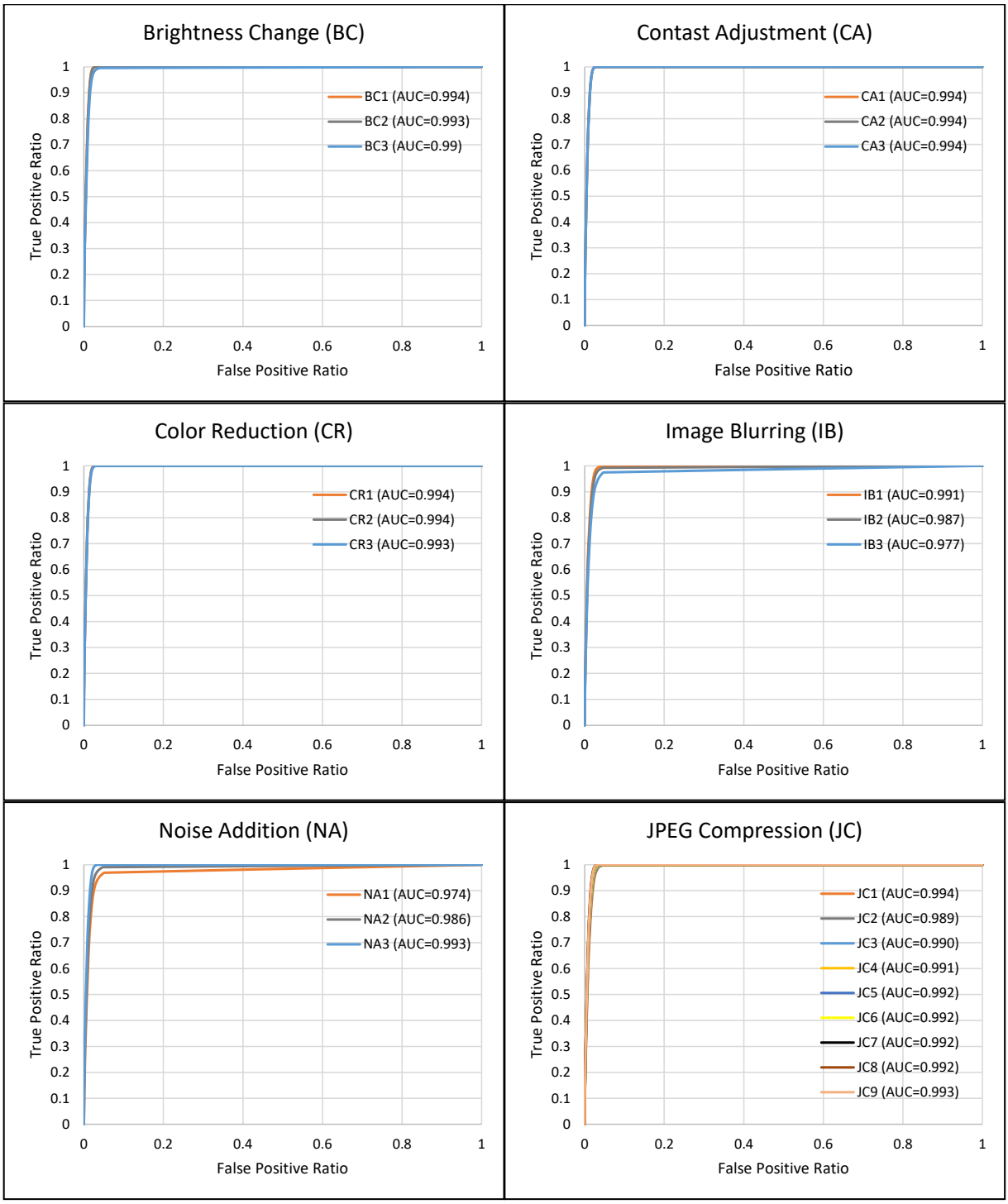


Figure 5.8: ROC curves for the CoMoFoD dataset with different post-processing techniques.

Method	Precision (%)	Recall (%)	F1 score (%)
[116]	35.9	**87.1**	50.8
[117]	5.7	84.6	10.7
[118]	41.7	42.4	42.0
[119]	50.9	17.3	25.8
[120]	48.1	63.6	54.8
Proposed	**51.3**	74.5	**60.8**

Table 5.3: Performance of the proposed and five other methods using the CASIA1 database for image splicing.

The receiver operating characteristic (ROC) curve is used to evaluate the discrimination performance in detecting copy-move forgeries. Figure 5.8 gives the ROC curves for the proposed method. The false positive rate (FPR) is given on the x-axis and the true positive rate (TPR) on the y-axis for different threshold values. This shows the tradeoff between TPR and FPR as the criterion for positivity is changed. The threshold ranges from 0 to 1 in steps of 0.01.

The area under the curve (AUC) indicates how well PADNET can discriminate between original and forged regions. The AUC lies between 0.5 and 1 where 0.5 denotes a poor classifier and 1 denotes a perfect classifier. For BC, increasing the level of post-processing from BC1 to BC3 only reduced the AUC by 0.4%. For CA, increasing the level of post-processing had no effect on the AUC which is 0.994. For CR, increasing the level of post-processing from CR1 to CR3 reduced the AUC by only 0.1%. For IB, increasing the level of post-processing from IB1 to IB3 reduced the AUC by 1.4%. For NA, increasing the level of postprocessing from NA3 to NA1 reduced the AUC by 1.9%. For JC, increasing the level of post-processing from JC1 to JC9 reduced the AUC by only 0.1%. Overall, the proposed method was able to detect copy-move forgeries in post-processed images with high discrimination performance.

5.5.2 CASIA1 dataset performance

The performance is now evaluated using the well known CASIA1 dataset for image splicing detection. Table 5.3 presents the precision, recall, and F1 score for the proposed and other methods. The best results are indicated in bold. These results show that the proposed method outperforms the others in terms of precision and F1 score. Only the method in [116] has a higher recall than the proposed method by only 12.6%. However, the proposed method has a higher precision and F1 score than the method in [116] by 15.4% and 6.7%, respectively. Figure 5.9 presents the ROC curve for PADNET. The AUC is 0.962 which indicates that PADNET is effective in discriminating between original and forged regions.

Overall, the proposed method was able to detect both kinds of forgery with high discrimination. Figure 5.10 shows examples of the results for images from the CoMoFoD and CASIA1 datasets. For copy-move forgery, the forgery mask is across the boundaries of the forged region. For image splicing, the forgery mask is over the entire forged region. This allows for detection of different kinds of forgery based on the shape of the generated mask.

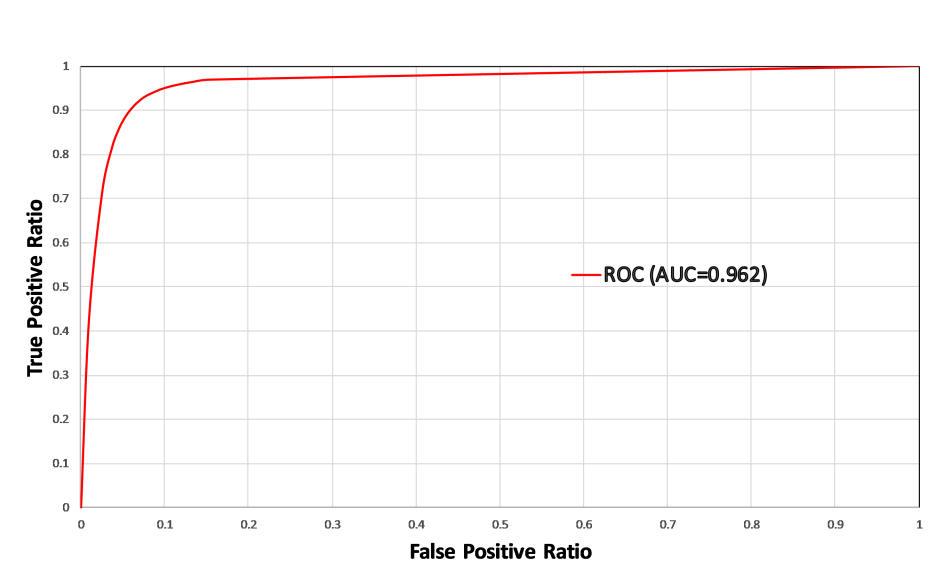


Figure 5.9: ROC curve for PADNET using the CASIA1 dataset.

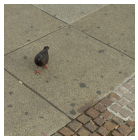
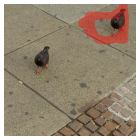
Type of Forgery	Original Image	Forged Image	Binary Mask	Result
Copy-Move				
Image Splicing				

Figure 5.10: Examples of the results obtained using the CoMoFoD and CASIA1 datasets.

5.6 Conclusion

This chapter introduced path aggregation DeepUNet (PADNET). This architecture is an end-to-end trainable deep neural network that employs a feature pyramid network (FPN) to aggregate features from multiscale levels of a ResNet-50 backbone. The extracted feature maps were used to train a DeepUNet architecture designed to learn discriminative features considering both high-level global features and low-level local features. Synthetic data was created to train PADNET. PADNET convergence during training was evaluated for both the boundaries of the forged regions and the entire forged regions. For copy-move forgeries, PADNET converged to a lower minimum when trained on the boundaries of the forged regions. This is because the key features related to forgeries are the artifacts that appear across the boundaries of the forged regions such as sharp edges. For image splicing, PADNET converged to a lower minimum when trained on the entire forged regions. This is because features in these regions such as color, brightness, and contrast differ from the rest of the image and so are key forgery features.

As there is a class imbalance problem when training PADNET on boundary labels for copy-move forgery detection, weighted binary cross-entropy was used as the loss function during training. Experiments were conducted to evaluate the convergence of PADNET with weighted binary cross-entropy and binary cross-entropy loss functions. The experimental results obtained indicate that convergence is improved using weighted binary cross-entropy as the loss function for copy-move forgery detection and binary cross-entropy as the loss function for image splicing detection.

PADNET was evaluated using the CoMoFoD dataset for copy-move forgery detection. Compared to six other methods in the literature, PADNET achieved the highest precision, recall, and F1 scores for all six post-processing techniques in the dataset. The ROC curves showed that PADNET is effective in discriminating between original and forged regions and the AUC was greater than 97% for all post-processing techniques. For image splicing detection, PADNET was evaluated using the CASIA1 dataset. Compared to four other methods in the literature, PADNET achieved the highest precision and F1 scores. In addition, the ROC curve showed that the AUC was 96.2%. Finally, this method addresses the two major limitations of state-of-the-art image forgery detection algorithms which are

(i) detecting multiple kinds of forgery, and

(ii) detecting forgeries in low resolution images and images that have been subjected to post-processing.

Chapter 6

Conclusion and Future Work

In this chapter, the contributions of this dissertation are summarized and suggestions for future research in the field of image forgery detection are given.

6.1 Conclusion

This dissertation considered the problem of detecting forgeries in images. The main contributions are the detection of forgeries in images that have been post-processed and methods that have low time complexity compared to other methods. Both copy-move forgery and image splicing were considered. First, a block-based method was introduced to detect copy-move forgeries in images using SVD for feature extraction followed by a KS-test for matching image blocks. Second, a deep learning method based on an encoder-decoder network was introduced. The encoder is used to transform the input image into a feature map and the decoder is employed to convert the feature map into a binary mask with the forged regions labeled. For image splicing detection, a deep learning method was introduced based on Mask RCNN with a new ResNet backbone design for faster convergence. Unlike other methods, a network was introduced to detect both kinds of forgery. This network is called path aggregation DeepUNet (PADNET). It consists of two networks: a feature pyramid network (FPN) to aggregate features from multiscale levels of a ResNet-50 backbone and a DeepUNet architecture to learn discriminative features considering both high-level global features and low-level local features. The proposed methods were tested on post-processed forged images and comparisons were made to other methods.

6.2 Future work

The work in this dissertation focused on detecting copy-move and splicing forgeries using block-based and deep learning approaches. In addition, post-processing techniques were used with the forged images to evaluate the effectiveness of the proposed methods. Inspired by the results obtained, the following topics can be investigated for future research.

(a) (b) (c)

Figure 6.1: Example of image retouching: (a) original image, (b) image with real makeup, and (c) retouched image.

6.2.1 Image retouching

Image forgery can be done in several ways such as copy-move forgery, image splicing, and image retouching. The forgery process changes the image feature statistics resulting in artifacts due to inconsistencies. These artifacts change according to the type of forgery. In copy-move forgery and image splicing, artifacts are the result of copying a region from one area to the other. On the other hand, image retouching does not employ a copy-paste process. Instead, an image is altered to improve the appearance of the image. For example, a retoucher can use editing tools to add makeup, smooth skin, or whiten teeth. Fig. 6.1 shows an example of a person with real makeup versus adding makeup by using image retouching. Artifacts resulting from image retouching can be found anywhere in the image and are consistent with the rest of the image. Thus, image retouching detection is a challenging task.

In the last decade several techniques have been introduced to detect image retouching which can be classified as: 1) before-and-after makeup [122], 2) disguise [123], plastic surgery [124], or 3) morphing based synthetic alterations [125]. In [122], a technique was introduced to reduce the impact of makeup on face recognition. First, features are extracted by capturing the shape and texture characteristics of the face. Then, SVM and Alligator classifiers are applied for comparison. In [123], a method was introduced to verify faces under disguise variations. This method automatically localizes feature descriptors to identify disguised face patches. In [124], the effect of plastic surgery on face recognition algorithms was examined. It was shown that current state-of-the-art face recognition algorithms cannot provide acceptable identification performance. Therefore, further research is required to address this problem.

In [125], a new technique was introduced to generate morphed images. The aim of this study is to show that printed photos pose serious concerns in terms of security. For example, an image can be visually very similar to the actual image but contain facial features of a different subject. This allows a criminal to exploit the passport of an accomplice with criminal records to overcome the security controls. Proper countermeasures must be taken to avoid storing digitally altered photos in electronic machine readable travel documents (eMRTDs).

Image retouching affects image pixels such that new features are generated. Deep learning networks such as UNet can be used to learn these features. This network will work as a classifier that can distinguish between retouched and original images. To train this network, a dataset

Figure 6.2: Fake images generated using the GAN model in [129].

is required such that retouched images are labeled as ones and original images are labeled as zeros. A robust model can be produced by providing images with and without real makeup. Thus, the network can learn to distinguish between real makeup and retouching.

6.2.2 GAN generated fake images

Recently, deep learning approaches such as generative adversarial network (GAN) [126] have been used to generate realistic fake images. A GAN consists of two networks: a generator and a discriminator. The generator learns to generate realistic fake images using random noise while the discriminator learns to discriminate between real and fake images. GAN models such as those in [127, 128] can be used to generate high resolution images that are indistinguishable from real ones by humans. Figure 6.2 shows examples of images generated using the GAN model in [129]. Several techniques have been developed to address the problem of detecting GAN generated fake images [130–132]. In [130], an approach was presented which is based on a combination of co-occurrence matrices and deep learning. A co-occurrence matrix is a matrix that is defined over an image to be the distribution of co-occuring pixel values at a given offset. First, co-occurrence matrices for the three color channels are obtained in the pixel domain. Then, a deep CNN framework is trained using these matrices to extract features. Detecting GAN generated images requires both real and fake images from the targeted GAN model. However, the model used by the attacker is often unavailable. To solve this problem, a GAN simulator (AutoGAN) was developed to simulate the artifacts produced by several well known GAN models [131]. This method can identify artifacts caused by up-sampling in the GAN pipeline. These artifacts are manifested as replications of spectra in the frequency domain and can be used as inputs to the classifier model. In [132], a method was introduced for discriminating between GAN-generated and camera images. This exploits the fact that the frequencies of saturated and under-exposed pixels will be suppressed by the generator normalization steps. In particular, a measure based on the frequency of over-exposed pixels provides good discrimination between GAN-generated and camera images.

Further research can be conducted by training a GAN model to regenerate the generator

used to create fake images. Unlike existing GAN-based methods, this method has three inputs: 1) real images, 2) fake images, and 3) realistic fake images. Fake images are generated from random noise by the generator and realistic fake images are generated by an unknown generator. The training process has two stages. In the first stage, the GAN model is trained with two inputs which are the real and fake images. The discriminator is trained to classify the real and fake images while the generator is trained to fool the discriminator by generating more realistic fake images. The training process continues until both networks cannot be improved any further. In the second stage, the GAN model is trained with three inputs: 1) real images, 2) realistic fake images from the trained generator, and 3) realistic fake images from the unknown generator. The realistic fake images are generated by two different generators. Thus, the discriminator is trained to extract generic features that are not related to a specific generator. These features can be used to classify real and fake images.

Bibliography

[1] Qazi, T. et al.: Survey on blind image forgery detection. IET Image Processing. 7(7), pp. 660–670 (2013)

[2] Shivakumar, B. L., Baboo, S. S.: Detecting copy-move forgery in digital images: A survey and analysis of current methods. Global Journal of Computer Science and Technology. 10(7), pp. 61–65 (2010)

[3] Mahdian, B., Saic, S.: Blind methods for detecting image fakery. In: Proceedings of the IEEE International Carnahan Conference on Security Technology, Prague, Czech Republic, Oct. 13–16, pp. 18–24 (2008)

[4] Ardizzone, E., Bruno, A., Mazzola, G.: Copy-move forgery detection by matching triangles of keypoints. IEEE Transactions on Information Forensics and Security. 10(10), pp. 2084–2094 (2015)

[5] Bharati, A. et al.: Detecting facial retouching using supervised deep learning. IEEE Transactions on Information Forensics and Security. 11(9), pp. 1903–1913 (2016)

[6] Kang, X., Wei, S.: Identifying tampered regions using singular value decomposition in digital image forensics. In: Proceedings of the IEEE International Conference on Computer Science and Software Engineering. Hubei, China, Dec. 12–14, pp. 926–930 (2008)

[7] Chien-Chang, C., Wang, H., Lin, C.: An efficiency enhanced cluster expanding block algorithm for copy-move forgery detection. In: Proceedings of the IEEE International Conference on Intelligent Networking and Collaborative Systems. Taipei, Taiwan, Sep. 2–4, pp. 26503–26522 (2015)

[8] Manu, V. T., Mehtre, B. M.: Detection of copy-move forgery in images using segmentation and SURF. In: Proceedings of International Symposium on Signal Processing and Intelligent Recognition Systems. Trivandrum, India, Dec. 16–19, pp. 645–654 (2014)

[9] Prakash, C. S., Om, H., Maheshkar, S., Maheshkar, V.: Keypoint-based passive method for image manipulation detection. Cogent Engineering. 5(1), (2018)

[10] alZahir, S., Hammad, R.: Blind copula based copy-move forgery detection algorithm. In: Proceedings of the IEEE International Conference on Consumer Electronics. Las Vegas, NV, USA, Jan. 8–10, pp. 436–437 (2017)

[11] Khan, S., Kulkarni, E. A.: An efficient method for detection of copy-move forgery using discrete wavelet transform. International Journal of Advanced Trends in Computer Science and Engineering. 2(5), pp. 1801–1806 (2010)

[12] Ryu, S., Kirchner, M., Lee, M., Lee, H.: Rotation invariant localization of duplicated image regions based on Zernike moments. IEEE Transactions on Information Forensics and Security. 8(8), pp. 1355–1370 (2013)

[13] Popescu, A. C., Farid, H.: Exposing digital forgeries by detecting duplicated image regions. Technical Report TR2004-515. Department of Computer Science, Dartmouth College, Hanover, NH, (2004)

[14] Sharma, K., Abrol, P.: Non-overlapping block-based parametric forgery detection model. Journal of Computer Applications. 133(3), pp. 17–24 (2016)

[15] Das, T., Hasan, R., Azam, Md, R.: A robust method for detecting copy-move image forgery using stationary wavelet transform and scale invariant feature transform. In: Proceedings of the International Conference on Computer, Communication, Chemical, Material and Electronic Engineering. Rajshahi, Bangladesh, Feb. 8–9, pp. 1–4 (2018)

[16] Wang, J., Liu, G., Li, H., Dai, Y., Wang, Z.: Detection of image region duplication forgery using model with circle block. In: Proceedings of the IEEE International Conference on Multimedia Information Networking and Security. Hubei, China, Nov. 18–20, pp. 25–29 (2009)

[17] Bravo-Solorio, S., Nandi, A. K.: Automated detection and localization of duplicated regions affected by reflection, rotation and scaling in image forensics. Signal Processing. 91(8), pp. 1759–1770 (2011)

[18] Langille, A., Gong, M.: An efficient match-based duplication detection algorithm. In: Proceedings of the IEEE Canadian Conference on Computer and Robot Vision. Quebec, QC, Canada, Jun. 7–9, pp. 64–64 (2006)

[19] Fridrich, J., Soukal, D., Lukàš, J.: Detection of copy-move forgery in digital images. In: Proceedings of the Digital Forensics Research Workshop. Cleveland, OH, USA, Aug. 6–8 (2003)

[20] Ahmed, B., Gulliver, T. A., alZahir, S.: Blind copy-move forgery detection using SVD and KS test. SN Applied Sciences. 2(1377), pp. 1–12 (2020)

[21] Rosenblatt, F.: The Perceptron: A Perceiving And Recognizing Automaton (Project PARA), 1st Ed. Cornell Aeronautical Laboratory, New York, NY, (1985)

[22] Yanling, Z., Bimin, D., Zhanrong, W.: Analysis and study of perceptron to solve XOR problem. In: Proceedings of the IEEE International Workshop on Autonomous Decentralized System. Beijing, China, Nov. 7, pp. 168–173 (2002)

[23] Deng, L., Yu, D.: Deep learning: Methods and applications. Foundations and Trends in Signal Processing. 7(3-4), pp. 197–387 (2014)

[24] He, K. et al.: Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In: Proceedings of the IEEE International Conference on Computer Vision. Santiago, Chile, Dec. 13–16, pp. 1026–1034 (2015)

[25] Bhandare, A., Bhide, M., Gokhale, P., Chandavarkar, R.: Applications of convolutional neural networks. Journal of Computer Science and Information Technologies. 7(5), pp. 2206–2215 (2016)

[26] Nair, V., Hinton, G. E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the International Conference on Machine Learning. Haifa, Israel, Jun. 21–24 (2010)

[27] Han, J., Morag, C.: The influence of the sigmoid function parameters on the speed of backpropagation learning. In: Proceedings of the International Workshop on Artificial Neural Networks. Torremolinos, Spain, Jun. 7–9, pp. 195 – 201 (1995)

[28] Krizhevsky, A., Sutskever, I., Hinton, G. E.: Imagenet classification with deep convolutional neural networks. In: Proceedings of the Conference on Advances in Neural Information Processing Systems. Lake Tahoe, NV, Dec. 3–8, pp. 1097–1105 (2012)

[29] Ranzato, M. A., Boureau, Y., LeCun, Y.: Sparse feature learning for deep belief networks. In: Proceedings of the Conference on Advances in Neural Information Processing Systems. pp. 1185–1192 (2008)

[30] LeCun, Y. et al.: Handwritten digit recognition with a back-propagation network. In: Proceedings of the Conference on Advances in Neural Information Processing Systems. pp. 396–404 (1990)

[31] Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge, MA, (2017)

[32] Rao, Y., Ni, J.: A deep learning approach to detection of splicing and copy-move forgeries in images. In: Proceedings of the IEEE International Workshop on Information Forensics and Security. Abu Dhabi, UAE, Dec. 4–7, pp. 1–6 (2016)

[33] Salomon, D.: Transformations and Projections in Computer Graphics. Springer-Verlag, Berlin, (2007)

[34] Joglekar, N. P., Chatur, P. N.: A compressive survey on active and passive methods for image forgery detection. Journal of Engineering and Computer Science. 4(1), pp. 10187–10190 (2015)

[35] Ahmed, B., Gulliver, T. A., alZahir, S.: Image splicing detection using mask-RCNN. Signal, Image and Video Processing. 14(5), pp. 1035–1042 (2020)

[36] Tralic, D., Zupancic, I., Grgic, S., Grgic, M.: CoMoFoD - New database for copy-move forgery detection. In: Proceedings of the International Symposium on Electronics in Marine. Zadar, Croatia, Sep. 25–27, pp. 49–54 (2013)

[37] Liu, G., Wang, J., Lian, S., Wang, Z.: A passive image authentication scheme for detecting region-duplication forgery with rotation. Journal of Network and Computer Applications. 34(5), pp. 1557–1565 (2011)

[38] Yao, H., Qiao, T., Tang, Z., Zhao, Y., Mao, H.: Detecting copy-move forgery using non-negative matrix factorization. In: Proceedings of the IEEE International Conference on Multimedia Information Networking and Security. Shanghai, China, Nov. 4–6, pp. 591–594 (2011)

[39] Chaitawittanun, N., Munlin, M.: An efficient clustering technique for copy-paste attack detection. Journal of Computer, Electrical, Automation, Control and Information Engineering. 8(2), pp. 394–402 (2015)

[40] Singh, V. K., Tripathi, R. C.: Fast and efficient region duplication detection in digital images using sub-blocking method. Journal of Advanced Science and Technology. 35, pp. 93–102 (2011)

[41] Wang, T., Tang, J., Luo, B.: Blind detection of region duplication forgery by merging blur and affine moment invariants. In: Proceedings of the IEEE International Conference on Image and Graphics. Qingdao, China, Jul. 26–28, pp. 258–264 (2013)

[42] Nguyen, H. C., Katzenbeisser, S.: Detection of copy-move forgery in digital images using radon transformation and phase correlation. In: Proceedings of the IEEE International Conference on Intelligent Information Hiding and Multimedia Signal Processing. Piraeus, Greece, Jul. 18–20, pp. 134–137 (2012)

[43] Li, Y.: Image copy-move forgery detection based on polar cosine transform and approximate nearest neighbor searching. Forensic Science International. 224(1-3), pp. 59–67 (2013)

[44] Warbhe, A. D., Dharaskar, R. V., Thakare, V. M.: A survey on keypoint based copy-paste forgery detection techniques. Procedia Computer Science. 78, pp. 61–67 (2016)

[45] Ulutas, G., Muzaffer, G.: A new copy move forgery detection method resistant to object removal with uniform background forgery. Mathematical Problems in Engineering. 2016, Art. ID 3215162 (2016)

[46] Otsu, N.: A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man, and Cybernetics. 9(1), pp. 62–66 (1979)

[47] Cao, Y., Gao, T., Fan, L., Yang, Q.: A robust detection algorithm for copy-move forgery in digital images. Forensic Science International. 214(1), pp. 33–43 (2012)

[48] Diane, W. N. N., Xingming, S., Moise, F. K.: A survey of partition-based techniques for copy-move forgery detection. The Scientific World Journal. 2014, Art. ID 975456 (2014)

[49] Cao, Y., Gao, T., Fan, L., Yang, Q.: A robust detection algorithm for region duplication in digital images. International Journal of Digital Content Technology and its Applications. 5(6), pp. 95-103 (2011)

[50] Bravo-Solorio, S., Nandi, A. K.: Passive forensic method for detecting duplicated regions affected by reflection, rotation, and scaling. In: Proceedings of the IEEE European Signal Processing Conference. Glasgow, UK, Aug. 24–28, pp. 824–828 (2009)

[51] Al-Qershi, O. M., Khoo, B. E.: Passive detection of copy-move forgery in digital images: State-of-the-art. Forensic Science International. 231(1-3), pp. 284–295 (2013)

[52] Li, G., Wu, Q., Tu, D., Sun, S.: A sorted neighborhood approach for detecting duplicated regions in image forgeries based on DWT and SVD. In: Proceedings of the IEEE International Conference on Multimedia and Expo. Beijing, China, Jul. 2–5, pp. 1750–1753 (2007)

[53] Zhang, T., Wang, R.: Copy-move forgery detection based on SVD in digital image. In: Proceedings of the International IEEE Congress on Image and Signal Processing. Tianjin, China, Oct. 17–19, pp. 1–5 (2009)

[54] Myna, A. N., Venkateshmurthy, M. G., Patil, C. G.: Detection of region duplication forgery in digital images using wavelets and log-polar mapping. In: Proceedings of the IEEE International Conference on Computational Intelligence and Multimedia Applications. Sivakasi, Tamil Nadu, India, Dec. 13–15, pp. 371-377 (2007)

[55] Kang, L., Cheng, X.: Copy-move forgery detection in digital image. In: Proceedings of the International IEEE Congress on Image and Signal Processing. Yantai, China, Oct. 16–18, pp. 2419–2421 (2010)

[56] Chakravarti, I. M., Laha, R. G., Roy, J.: Handbook of Methods of Applied Statistics. Wiley, New York, NY, USA (1967)

[57] Justel, A., Peña, D., Zamar, R.: A multivariate Kolmogorov-Smirnov test of goodness of fit. Statistics and Probability Letters. 35(3), pp. 251–259 (1997)

[58] Dash, S., Jena, U. R.: Texture classification using steerable pyramid based Laws' masks. Journal of Electrical Systems and Information Technology. 4(1), pp. 185–97 (2017)

[59] Unser, M., Chenouard, N., Ville, D. V. D.: Steerable pyramids and tight wavelet frames in $L_2(R^d)$. IEEE Transactions on Image Processing. 20(10), pp. 2705–2721 (2011)

[60] Freeman, W. T., Adelson, E. H.: The design and use of steerable filters. IEEE Transactions on Pattern Analysis and Machine Intelligence. 13(9), pp. 891–906 (1991)

[61] Klema, V., Laub, A.: The singular value decomposition: Its computation and some applications. IEEE Transactions on Automatic Control. 25(2), pp. 164–176 (1980)

[62] Murali, S., Chittapur, G. B., Prabhakara, H. S., Anami, B. S.: Comparision and analysis of photo image forgery detection techniques. Journal of Computational Sciences and Applications. 2(6), pp. 45–56 (2013)

[63] Cozzolino, D., Gragnaniello, D., Verdoliva, L.: Image forgery detection based on the fusion of machine learning and block-matching methods. arXiv preprint arXiv:1311.6934. (2013)

[64] Shi, Y. Q., Chen, C., Chen, W.: A natural image model approach to splicing detection. In: Proceedings of the Workshop on Multimedia and Security. Dallas, TX, USA, Sep. 20–21, pp. 51–62 (2007)

[65] He, Z., Wei, L., Wei, S., Huang, J.: Digital image splicing detection based on Markov features in DCT and DWT domain. Pattern Recognition. 45(12), pp. 4292–4299 (2012)

[66] Mushtaq, S., Mir, A. H.: Novel method for image splicing detection. In: Proceedings of the IEEE International Conference on Advances in Computing, Communications and Informatics. New Delhi, India, Sep. 24–27, pp. 2398–2403 (2014)

[67] Dong, J., Wang, W., Tan, T., Shi, Y. Q.: Run-length and edge statistics based approach for image splicing detection. In: Proceedings of the Workshop on Digital Watermarking. Busan, Korea, Nov. 10–12, pp. 76–87 (2008)

[68] He, Z., Lu, W., Sun, W.: Improved run length based detection of digital image splicing. In: Proceedings of the International Workshop on Digital Watermarking. Atlantic City, NJ, USA, Oct. 23–26, pp. 349–360 (2011)

[69] Lee, H., Ekanadham, C., Ng, A. Y.: Sparse deep belief net model for visual area V2. In: Proceedings of the Conference on Advances in Neural Information Processing Systems. Vancouver, BC, Canada, Dec. 3–5, pp. 873–880 (2007)

[70] Hugo, L., Bengio, Y., Louradour, J., Lamblin, P.: Exploring strategies for training deep neural networks. Journal of Machine Learning Research. 10(1), pp. 1–40 (2009)

[71] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE. 86(11), pp. 2278–2324 (1998)

[72] Swietojanski, P., Ghoshal, A., Renals, S.: Convolutional neural networks for distant speech recognition. IEEE Signal Processing Letters. 21(9), pp. 1120–1124 (2014)

[73] Tuama, A., Comby, F., Chaumont, M.: Camera model identification with the use of deep convolutional neural networks. In: Proceedings of the IEEE International Workshop on Information Forensics and Security. Abu Dhabi, UAE, Dec. 4–7, pp. 1–6 (2016)

[74] Baroffio, L., Bondi, L., Bestagini, P.: Camera identification with deep convolutional networks. arXiv preprint arXiv:1603.01068. (2016)

[75] Rota, P., Sangineto, E., Conotter, V.: Bad teacher or unruly student: Can deep learning say something in image forensics analysis? In: Proceedings of the IEEE International Conference on Pattern Recognition. Cancun, Mexico, Dec. 4–8, pp. 2503–2508 (2016)

[76] Bayar, B., Stamm, M. C.: A deep learning approach to universal image manipulation detection using a new convolutional layer. In: Proceedings of the ACM Workshop on Information Hiding and Multimedia Security. Galicia, Spain, Jun. 20–22, pp. 5–10 (2016)

[77] Wang, Q., Zhang, R.: Double JPEG compression forensics based on a convolutional neural network. EURASIP Journal on Information Security. 2016, Art. ID 23 (2016)

[78] Ying, Z., Goh, J., Win, L. L., Thing, V. L. L.: Image region forgery detection: A deep learning approach. In: Proceedings of the Singapore Cyber-Security Conference. Singapore, Jan. 14–15, pp. 1–11 (2016)

[79] Zhang, Z., Zhang, Y., Zhou, Z., Luo, J.: Boundary-based image forgery detection by fast shallow CNN. In: Proceedings of the IEEE International Conference on Pattern Recognition. Beijing, China, Aug. 20–24, pp. 2658–2663 (2018)

[80] Ouyang, J., Liu, Y., Liao, M.: Copy-move forgery detection based on deep learning. In: Proceedings of the IEEE International Congress on Image and Signal Processing, Biomedical Engineering and Informatics. Shanghai, China, Oct. 14–16, pp. 1–5 (2017)

[81] Lin, T. Y. et al.: Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, HI, USA, Jul. 22–25, pp. 2117–2125 (2017)

[82] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, NV, USA, Jun. 26–Jul. 1, pp. 770–778 (2016)

[83] Deng, J. et al.: Imagenet: A large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Miami, FL, USA, Jun. 20–25, pp. 248–255 (2009)

[84] Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Proceedings of the International Conference on Artificial Intelligence and Statistics. Sardinia, Italy, May 13–15, pp. 249–256 (2010)

[85] He, K., Gkioxari, G., Dollar, P., Girshick, R.: Mask R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision. Honolulu, HI, USA, Jul. 22–25 (2017)

[86] Lin, T. Y. et al.: Microsoft COCO: Common objects in context. In: Proceedings of the European Conference on Computer Vision. Zurich, Switzerland, Sep. 6–12, pp. 740–755 (2014)

[87] Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence. 39(6), pp. 1137–1149 (2016)

[88] Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision. Santiago, Chile, Dec. 13–16, pp. 1440–1448 (2015)

[89] Meena, K. B., Tyagi, V.: Image forgery detection: Survey and future directions. Data, Engineering and Applications. 2, pp. 163–194 (2019)

[90] Braxmeier, J.: Stunning free images and royalty free stock. Available via https://pixabay.com. Cited 09-Jun-2019.

[91] Matterport Inc.: Mask-RCNN. Available via https://github.com/matterport/Mask_RCNN. Cited 09-Jun-2019.

[92] Qian, N.: On the momentum term in gradient descent learning algorithm. Neural Networks. 12(1), pp. 145–151 (1999)

[93] Keras-team (2015) Keras. Available via https://github.com/fchollet/keras. Cited 09-Jun-2019.

[94] Casella, G., Berger, R. L.: Statistical Inference, 2nd Ed. Duxbury Press, Belmont, CA, (2017)

[95] Sakamoto, H.: On the distributions of the product and the quotient of the independent and uniformly distributed random variables. Tohoku Mathematical Journal, First Stage. 49, pp. 243–260 (1943)

[96] Zhang, T., Wang, R.: Copy-move forgery detection based on SVD in digital image. In: Proceedings of the IEEE International Congress on Image and Signal Processing. Tianjin, China, Oct. 17–19, pp. 1–5 (2009)

[97] Hegazi, A., Taha, A., Selim, M. M.: An improved copy-move forgery detection based on density-based clustering and guaranteed outlier removal. Journal of King Saud University-Computer and Information Sciences. (2019)

[98] Bi, X., Wei, Y., Xiao, B., Li, W.: RRU-Net: The ringed residual U-Net for image splicing forgery detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. Long Beach, CA, USA, Jun. 16–17 (2019)

[99] Zhang, R., Ni, J.: A dense U-Net with cross-layer intersection for detection and localization of image forgery. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing. Barcelona, Spain, May 4–8, pp. 2982–2986 (2020)

[100] Wu, Y., Abd-Almageed, W., Natarajan, P.: BusterNet: Detecting copy-move image forgery with source/target localization. In: Proceedings of the European Conference on Computer Vision. Munich, Germany, Sep. 8–14, pp. 168–184 (2018)

[101] Abdalla, Y., Iqbal, M. T., Shehata, M.: Copy-move forgery detection and localization using a generative adversarial network and convolutional neural-network. Information. 10(9), pp. 286–312 (2019)

[102] Nixon, M., Aguado, A.: Feature Extraction and Image Processing for Computer Vision, 3rd Ed. Academic Press, New York, NY, (2012)

[103] Mannor, S., Peleg, D., Rubinstein, R.: The cross entropy method for classification. In: Proceedings of the International Conference on Machine Learning. Bonn, Germany, Aug. 7–11, pp. 561–568 (2005)

[104] Ryu, S. J., Lee, M. J., Lee, H. K.: Detection of copy-rotate-move forgery using Zernike moments. In: Proceedings of the International Workshop on Information Hiding. Calgary, AB, Canada, Jun. 28–30, pp. 51–65 (2010)

[105] Wu, Y., Abd-Almageed, W., Natarajan, P.: Deep matching and validation network: An end-to-end solution to constrained image splicing localization and detection. In: Proceedings of the ACM International Conference on Multimedia. Mountain View, CA, USA, Oct. 23–27, pp. 1480–1502 (2017)

[106] Kumar, A., Bhavsar, A., Verma, R.: Syn2Real: Forgery classification via unsupervised domain adaptation. In: Proceedings of the IEEE Winter Conference on Applications of Computer Vision Workshops. Snowmass Village, CO, USA, Mar. 1–5, pp. 63–70 (2020)

[107] Lee, H., Ekanadham, C., Ng, A. Y.: Sparse deep belief net model for visual area V2. In: Proceedings of the Conference on Advances in Neural Information Processing Systems. pp. 873–880 (2008)

[108] Chen, J., Kang, X., Liu, Y., Wang, Z. J.: Median filtering forensics based on convolutional neural networks. IEEE Signal Processing Letters. 22(11), pp. 1849–1853 (2015)

[109] Xu, G., Wu, H. Z., Shi, Y. Q.: Structural design of convolutional neural networks for steganalysis. IEEE Signal Processing Letters. 23(5), pp. 708–712 (2016)

[110] Abdalla, Y., Iqbal, M. T., Shehata, M.: Copy-move forgery detection and localization using a generative adversarial network and convolutional neural-network. Information. 10(9), Art. ID 286 (2019)

[111] He, Z., Lu, W., Sun, W.: Improved run length based detection of digital image splicing. In: Proceedings of the International Workshop on Digital Forensics and Watermarking. Atlantic City, NY, USA, Oct. 23–26, pp. 349–360 (2011)

[112] Li, R. et al.: Deepunet: A deep fully convolutional network for pixel-level sea-land segmentation. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. 11(11), pp. 3954–3962 (2018)

[113] Mannor, S., Peleg, D., Rubinstein, R.: The cross entropy method for classification. In: Proceedings of the International Conference on Machine Learning. Bonn, Germany, Aug. 7–11, pp. 561–568 (2005)

[114] Xie, S., Tu, Z.: Holistically-nested edge detection. In: Proceedings of the IEEE International Conference on Computer Vision. Santiago, Chile, Dec. 7–13, pp. 1395–1403 (2015)

[115] Dong, J., Wang, W., Tan, T.: Casia image tampering detection evaluation database. In: Proceedings of the IEEE China Summit and International Conference on Signal and Information Processing. Beijing, China, Jul. 6–10, pp. 422–426 (2013)

[116] Ye, S., Sun, Q., Chang, E. C.: Detecting digital image forgeries by measuring inconsistencies of blocking artifact. In: Proceedings of the IEEE International Conference on Multimedia and Expo. Beijing, China, Jul. 2–5, pp. 12–15 (2007)

[117] Ferrara, P., Bianchi, T., De Rosa, A., Piva, A.: Image forgery localization via fine-grained analysis of CFA artifacts. IEEE Transactions on Information Forensics and Security. 7(5), pp. 1566–1577 (2012)

[118] Wei, Y., Bi, X., Xiao, B.: C2R-Net: The coarse to refined network for image forgery detection. In: Proceedings of the IEEE International Conference on Trust, Security and Privacy in Computing and Communications. New York, NY, USA, Aug. 1–3, pp. 1656–1659 (2018)

[119] Long, J., Shelhamer, E., Darrell T,.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, MA, USA, Jun. 7–12, pp. 3431–3440 (2015)

[120] Chen, L. C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)

[121] Eckert, M. L., Kose, N., Dugelay, J. L.: Facial cosmetics database and impact analysis on automatic face recognition. In: Proceedings of the International IEEE Workshop on Multimedia Signal Processing. Pula, Italy, Sep. 30–Oct. 2, pp. 434–439 (2013)

[122] Kose, N., Aprville, L., Dugelay, J. L.: Facial makeup detection technique based on texture and shape analysis. In: Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition. Ljubljana, Slovenia, May 4–8, pp. 1–7 (2015)

[123] Dhamecha, T. I., Singh, R., Vatsa, M., Kumar, A.: Recognizing disguised faces: Human and machine evaluation. PLOS ONE. 9(7), pp. 1–16 (2014)

[124] Singh, R., Vatsa, R., Bhat, H. S.: Plastic surgery: A new dimension to face recognition. IEEE Transactions on Information Forensics and Security. 5(3), pp. 441–448 (2010)

[125] Ferrara, M., Franco, A., Maltoni, D.: The magic passport. In: Proceedings of the IEE International Joint Conference on Biometrics. Clearwater, FL, USA, Sep. 29–Oct. 2, pp. 1–7 (2014)

[126] Goodfellow, I., Pouget-Abadie, J., Mirza, M.: Generative adversarial nets. In: Proceedings of the International Conference on Neural Information Processing Systems. Cambridge, MA, USA, Dec. 8–13, pp. 2672–2680 (2014)

[127] Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Long Beach, CA, USA, Jun. 15–20, pp. 4401–4410 (2019)

[128] Brock, A., Donahue, J., Simonyan, K.: Large scale GAN training for high fidelity natural image synthesis. In: Proceedings of the International Conference on Learning Representations. New Orleans, LA, USA, May 6–9 (2019)

[129] Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of GANs for improved quality, stability, and variation. In: Proceedings of the International Conference on Learning Representations. Vancouver, BC, Canada, Apr. 30–May 3 (2018)

[130] Nataraj, L., Mohamed, T. M., Manjunath, B. S.: Detecting GAN generated fake images using co-occurrence matrices. Computer Vision and Pattern Recognition. pp. 532-1–532-7 (2019)

[131] Zhang, X., Karaman, S., Chang, S. F.: Detecting and simulating artifacts in GAN fake images. In: Proceedings of the IEEE International Workshop on Information Forensics and Security. Delft, Netherlands, Dec. 9–12, pp. 1–6 (2019)

[132] McCloskey, S., Albright, M.: Detecting GAN-generated imagery using saturation cues. In: Proceedings of the IEEE International Conference on Image Processing. Taipei, Taiwan, Sep. 22–25, pp. 4584–4588 (2019)