algorithms

MDPI

Article

Detecting Image Forgery over Social Media Using U-NET with Grasshopper Optimization

Niousha Ghannad and Kalpdrum Passi *

School of Engineering and Computer Science, Laurentian University, Sudbury, ON P3E 2C6, Canada; nghannad@laurentian.ca

* Correspondence: kpassi@laurentian.ca

Abstract: Currently, video and digital images possess extensive utility, ranging from recreational and social media purposes to verification, military operations, legal proceedings, and penalization. The enhancement mechanisms of this medium have undergone significant advancements, rendering them more accessible and widely available to a larger population. Consequently, this has facilitated the ease with which counterfeiters can manipulate images. Convolutional neural network (CNN)-based feature extraction and detection techniques were used to carry out this task, which aims to identify the variations in image features between modified and non-manipulated areas. However, the effectiveness of the existing detection methods could be more efficient. The contributions of this paper include the introduction of a segmentation method to identify the forgery region in images with the U-Net model's improved structure. The suggested model connects the encoder and decoder pipeline by improving the convolution module and increasing the set of weights in the U-Net contraction and expansion path. In addition, the parameters of the U-Net network are optimized by using the grasshopper optimization algorithm (GOA). Experiments were carried out on the publicly accessible image tempering detection evaluation dataset from the Chinese Academy of Sciences Institute of Automation (CASIA) to assess the efficacy of the suggested strategy. The results show that the U-Net modifications significantly improve the overall segmentation results compared to other models. The effectiveness of this method was evaluated on CASIA, and the quantitative results obtained based on accuracy, precision, recall, and the F1 score demonstrate the superiority of the U-Net modifications over other models.

Keywords: U-Net; convolutional neural network; forged images; social media; grasshopper algorithm

check for updates

Citation: Ghannad, N.; Passi, K. Detecting Image Forgery over Social Media Using U-NET with Grasshopper Optimization. *Algorithms* **2023**, *16*, 399. https://doi.org/10.3390/a16090399

Academic Editor: Yi Chang

Received: 9 June 2023
Revised: 4 August 2023
Accepted: 21 August 2023
Published: 23 August 2023

1. Introduction

Nowadays, manipulating digital images has become very simple due to powerful computers, advanced image editing software, and high-resolution imaging tools. The development of image processing software, tools, and techniques has increased the number of fake images that cannot be easily recognized without special tools [1]. As digital images become more widespread daily, tools for distorting and falsifying digital images also become more available. Therefore, there is a need for methods that can detect the forgery of images. Some methods have been proposed to achieve this goal [2]. In some methods, information is embedded in the image from the beginning so that it can be examined when necessary and the image's authenticity can be determined. These methods are called active detection methods. Nevertheless, recognizing the originality of the image without the need for previous embedded information or the primary content of the image has been an important research topic in recent years. These methods are known as passive or blind methods [1].

With the growth of image processing software, many fake images have been created. In most cases, these fake images cannot be recognized by the human eye and therefore require image forgery detection algorithms. An image forgery detection algorithm should be able to make a correct decision regarding its authenticity without needing basic information about the image and its content. The need to detect image forgery is felt in many fields, such as legal-judicial investigations, photography of publications, news agencies, criminal investigations, insurance companies, medical images, scientific societies, etc. [3].

Researchers have proposed methods to detect different types of image forgery; however, most are designed to detect one type of forgery. Copy-move forgery is very difficult to detect because the forgery region is part of the same image, and the statistical distribution of the image pixels is similar. Splice detection methods look for changes in the statistical distribution of pixels, differences in the compression level, and various features related to the camera for detecting forgery. There are several methods for detecting copy-move forgeries, such as keypoint-based methods, block methods, segmentation-based methods, texture-based methods, and a combination of keypoint and block methods. Numerous methods, such as camera-based methods, Markov feature-based methods, compressed image-based methods, and transform-based methods, have been introduced by researchers to detect splicing forgery. Detecting a type of forgery is inefficient because it is impossible to identify the type of forgery quickly by looking at the fake image. A few methods have been presented to detect various types of image forgery [4].

With the emergence of machine learning and deep learning, these techniques have been applied to classify fake images and locate affected regions. Deep learning aims to learn high-level abstracts from data using hierarchical architectures. Convolutional neural networks (CNNs) have a great learning capacity due to thin connections and weight sharing and have received much attention in recent years. The U-Net model has become a popular component identification algorithm in recent years due to its precise response, high accuracy, high speed of processing and learning, no need for large data sets for learning, and no need for complex and expensive hardware.

This research proposes image segmentation using the modified U-Net deep learning model to classify and extract features. Therefore, in this paper, our contributions include:

(1) We are proposing a new method based on a U-net CNN to improve the fine details of images, which aims to increase the efficiency of image forgery detection.

(2) Inspired by the proposed deep learning model for complex medical image segmentation [5], we intend to conduct a similar task to exploit the unique features of U-Net for image enhancement in the current work.

(3) The proposed method's meta-parameters of the U-Net network are optimized using the grasshopper optimization algorithm (GOA). These parameters are the initial learning rate, the learning rate drop rate, the learning rate of cuts with the drop rate, and the mini-batch size.

U-Net integrates local information through the downsampling and upsampling of the path, thereby extracting texture information (patterns). Because the kernel used in U-networks is independent of the input image size, this model does not require a dense layer and works well with input images of any size.

1.1. Related Work

Existing methods are divided into two essential categories: block methods and keypoint methods for detecting copy-move.

1.1.1. Detecting Copy-Move Forgery Using Block Methods

Warif et al., 2016 [6], reviewed the latest techniques in the field of copy-move forgery detection. They described copy-move forgery detection using block and keypoint methods. Zhao et al., 2017 [7], introduced a technique based on image segmentation and an algorithm called swarm intelligence (SI). This technique divided the image into small blocks without overlapping (common parts). The SI algorithm was implemented to find the best possible parameters in each layer. With the scale-invariant feature transform (SIFT), these parameters were used to identify each layer. This method had a high false-positive rate. Jalab et al., 2019 [8], proposed a method in which the image blocks are transformed into discrete wavelet transforms and used to extract features from a new fractional texture descriptor based on Machado fractional entropy.

Deep learning-based techniques have outperformed advanced machine learning-based methods in many computer vision applications. In detecting the forgery of digital images, Bunk et al., 2017 [9], introduced two methods to detect image forgery. In the initial technique, an end-to-end system was presented for detection. Then, localization in manipulated digital images was performed based on Radon transformation and deep learning methods. The Radon transform is an integral transformation with a practical inverse application in rebuilding images from medical CT scans. Additionally, a method has been created to utilize the Radon transform in constructing a map of the polar regions of a planet by a spacecraft orbiting around its poles. The feature resampling combination was used in the second technique, and these features were obtained based on maps. In order to classify, the LSTM model was used to find manipulated areas. Zhang et al., 2016 [10], introduced a two-stage deep learning method for feature learning. In the first stage, stack automatic encoders were used, by which the model learns the characteristics of each piece of the image. In the second step, the texture information of the whole image is integrated.

Zhou et al., 2017 [11], introduced a technique based on the block method. The processing unit of each block is a powerful CNN. This method detected splicing forgery, and its effectiveness was observed in JPEG compression. Kim and Lee, 2017 [12], introduced an image manipulation detection algorithm using a deep CNN model. This neural network consists of four steps: a high-pass filter, two convolutional layers, two fully connected layers, and an output layer. In the experiments, the images were resized to 256 × 256 dimensions using the median filter method, Gaussian filtering, and collective white Gaussian noise.

Finding an ink mismatch is an essential step in detecting image forgery. Khan et al., 2018 [13], introduced a deep learning method to detect ink gaps in hyperspectral document images. These researchers extracted the spectral responses from the ink pixels of the hyperspectral document image, converted them to an image format suitable for CNNs, and fed them to a CNN for classification. The proposed technique successfully recognized different types of ink in super-sized document images for forgery detection.

Liu et al., 2017 [14], introduced an effective copy-transfer manipulation detection technique based on a convolutional kernel network (CKN), which is a combination of matrix calculations and convolutional operations. Cozzolino et al., 2018 [15], introduced a weakly supervised domain adaptation method for distinguishing between real and fake images. These researchers trained the auto-anchor-based method on the source domain and separated fake and authentic images in the hidden space. Khalid and Woo, 2020 [16], introduced a single-class classification model based on a one-class variational auto-encoder to detect fake images of human faces.

Marra et al., 2020 [17], introduced a CNN-based image forgery detection and localization method that makes decisions based on the entire image without changing its size. This framework consists of three blocks: piecewise feature extraction, image feature combination with several fusion strategies, and global classification. Meena and Tyagi, 2019 [18], and Walia and Saluja, 2018 [19], discussed all types of image forgery detection techniques: image link detection, copy-move forgery detection, image resampling detection and image manipulation detection.

Li et al., 2020 [20], introduced a face R-ray method to detect forgery in face images and manipulated boundaries in fake images using CNN HRNet in their framework. Abbas et al., 2021 [21], performed experiments using two deep learning models for copy-transfer image forgery: small VGGNet and Mobile Net V2. Saber et al., 2020 [22], investigated all types of forgery (digital watermarks, digital signatures, image linking, image manipulation, and copy-transfer forgery).

Zhang and Ni, 2020 [23], introduced a U-Net composed of dense convolutional and de-convolutional networks. The first network is a deductive sampling method for feature recovery, and the second network is an incremental sampling approach for feature map size recovery. Liu et al., 2018 [24], designed a CNN segmentation approach to find regions in digital images. First, a unified CNN architecture was developed to handle sliding windows with different input scales and colors. Then, high-accuracy CNN training processes were built by sampling the training areas. Bi et al., 2020 [25], used a non-fixed and fixed encoder to build a U-net with a dual encoder (D-Undet). The unfixed encoder captures the fingerprint of the image, which cleans the original and fake regions. In contrast, a fixed encoder provides direction data to facilitate network training and detection.

Marra et al., 2018 [26], tested the effectiveness of several image forgery detectors on image-to-image translation, including both ideal settings and even compression modes. Compression is generally used when uploading images to social media sites. Kadam et al., 2021 [27], introduced a method based on multi-image splicing using MobileNet V1. Jaiswal and Srivastava, 2019 [28], proposed a framework in which images are fed to a CNN and then processed by several layers for feature extraction and was finally used as a training vector for a recognition model. For feature extraction, ResNet-50 pre-trained deep learning was used.

Stehouwer et al., 2020 [29], proposed the attention method to analyze and refine feature maps for recognition tasks. Learned attention maps emphasize functional regions and identify altered regions to improve binary classification (fake face vs. real face). Nguyen et al., 2019 [30], built a convolutional neural network that uses a multi-task learning strategy to detect and segment manipulated facial images and videos and locates forged regions. Information received from one task is shared with the second task, thus improving the efficiency of both activities. A semi-supervised learning strategy has been used to strengthen the generality of the network. An encoder and a Y-shaped decoder were placed within this network.

Li and Lyu, 2018 [31], introduced exposing deepfake videos by detecting face-warping artifacts. Deepfake techniques create fixed-size images of the face that must be carefully warped to match the original facial makeup. Due to the dispersion of resolution between the warped face area and surrounding texture, this warping operation produces different side images. As a result, deepfake videos are detected using these side images.

Gidaris et al., 2018 [32], introduced a method for learning image features by training a CNN to detect the 2D rotation applied to the received input image. They used unsupervised representation learning by predicting image rotations. Wang et al., 2020 [33], proposed a method that consists of three parts: single-image hyper definition, semantic segmentation hyper definition, and a feature affinity module for semantic segmentation. Yu et al., 2022 [34], also used dual attention in pyramidal visual feature maps to fully investigate visual semantic relations and improve the level of generated sentences.

Singh and Sharma, 2021 [35], proposed a convolutional neural network to identify fake images shared on social media platforms. High-pass filters of image processing are used in the first layer to initialize the weights, which helps the neural network to converge faster and achieve better accuracy. Interpretability is a common concern in deep learning models. The proposed framework uses gradient-weighted class activation mapping to generate heat maps and localize the manipulated area of the image. The model has been validated with the publicly available CASIA dataset, and a 92.3% accuracy has been obtained, which is better than the other previous models.

1.1.2. Detecting Copy-Move Forgery Using Keypoint Methods

Wang et al. introduced a passive copy-move manipulation detection method using keypoint features [36]. This method has high computational complexity and lacks usability in real-time systems, which makes it inefficient. Another work [37] investigated moment detection methods. The most important feature of these methods is to be strong against blurring. However, it was unsuccessful in some cases, such as spinning. Wang et al. introduced a copy-move manipulation detection method [38]. They used quaternion exponent moments (QEMs). The results showed that this method is only efficient for copying and pasting in conditions such as scaling and rotation.

Kuznetsov and Myasnikov [39] introduced a hashing technique for copy-move detection, which can be used to detect duplicate regions transformed by a unique pre-processing technique. Niu et al., 2021 [40], introduced a fast and accurate forgery detection algorithm for copy-move based on complex-valued moment invariants.

Huang and Ciou, 2019 [41], introduced a keypoint-based image forgery method based on a superpixel segmentation algorithm and Helmert transform to detect copy-move forgery detection. At first, key points and descriptors were extracted using the SIFT algorithm. Then, using a descriptor, identical pairs were detected by calculating the similarities between key points. Based on spatial distance, geometric constraints, and Helmert transformation, these identical pairs were grouped to obtain approximate areas of forgery. Then, these approximate areas were improved, and the errors were corrected. Finally, the location of the fake areas was found more precisely.

Dixit and Bag, 2021 [42], introduced a complex framework using keypoint matching by the k-nearest neighbor algorithm and K-d tree to detect fake images. Yang et al., 2021 [43], introduced a method based on keypoint matching to detect copy-move forgery.

1.1.3. Issues and Motivation for Current Research

In current articles, most researchers have addressed an important issue: minimum accuracy. These techniques can detect splicing or copy-move fraud, but not both. Another essential issue with these techniques is that they detect forgery with low accuracy. The issue of minimum accuracy has been a significant concern for researchers. While several techniques have been developed to detect splicing or copy-move fraud, they are often limited in their accuracy and cannot detect both types of forgeries.

Therefore, we propose a novel approach that combines a deep learning-based method with an optimization algorithm. Specifically, we use a U-NET network with the grasshopper optimization algorithm to detect image forgeries. The U-NET network is a popular architecture for image segmentation tasks that has shown impressive results in various applications. In our approach, the U-NET network is used to segment the images into regions of interest, while the grasshopper optimization algorithm is used to optimize the segmentation process and improve the accuracy of the results.

One of the key advantages of our approach is its ability to detect various types of image forgeries, including splicing and copy-move forgeries, with high accuracy. This is made possible by the use of the U-NET network, which is capable of capturing complex features and patterns in the image. Additionally, our approach can be applied in real-time systems, making it more efficient than some existing methods that have high computational complexity and lack usability in real-time systems.

In comparison to existing methods, our approach offers several significant advantages. Firstly, it is capable of detecting a wide range of image forgeries with high accuracy, making it a versatile tool for image forensics. Secondly, our approach is efficient and can be applied in real-time systems, making it a practical solution for various applications, such as in forensic investigations or in the detection of fake images on social media. Finally, our approach combines deep learning with an optimization algorithm, which is a novel and promising approach in the field of image forensics. We believe that our approach has the potential to become a valuable tool in the field of image forensics, and we look forward to future research in this area.

2. Materials and Methods

2.1. Methodology

The deep learning network known as U-Net is utilized for picture segmentation and has the potential to be applied for counterfeit detection as well. Based on the behavior of grasshoppers, GOA is a metaheuristic optimization algorithm. The proposed method utilizes U-Net architecture to detect image forgery and forgery regions. Figure 1 provides an illustration of the format of this procedure. The proposed method is carried out in several steps:

1. The pre-processing of data, which includes the separation of data into a training and test set, as well as the scaling of photographs to a consistent size.
2. Develop the architecture of the U-Net network, with an encoder and a decoder, in addition to many convolutional layers and max-pooling/upsampling layers.
3. Train the U-Net network for a predetermined number of epochs utilizing the Dice coefficient loss function and the Adam optimizer.
4. Apply the GOA algorithm to optimize the U-Net network by utilizing the network parameters as decision variables and the mean squared error as the fitness function.
5. Evaluate the performance of the U-Net network based on measures such as precision, recall, and the F1 score by testing it with the test set.

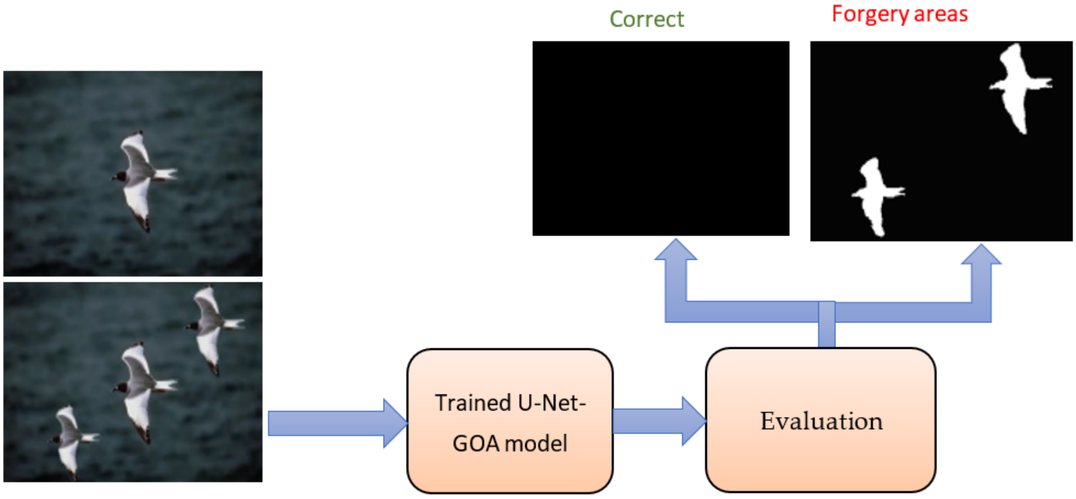


Figure 1. Structure of the proposed method.

The proposed method should predict the masks of the given image. Therefore, this method needs to have an encoder- and decoder-type of architecture because we are trying to highlight the manipulated regions in the given image. The method should split the image into two parts. One class is for the manipulated region (light color), and the other is for the entire region (dark color).

In the proposed method, we consider U-Net as a base model because it already has proven results for similar image segmentation types and satisfies the above conditions. The initial U-Net architecture for biomedical image segmentation consisted of two paths, one for the encoder, which comprised a stack of convolution, activation, and pooling layers, and one for the decoder. The other is for decoding, which is symmetrical with the encoder path. It should be mentioned that we have a mask for fake images, but we do not have a mask for intact images. Therefore, for all intact images, the default mask is considered, which only contains black pixels, and for all intact images, the mask generated by the U-Net is the same.

2.2. Models and Architecture

Various techniques have been proposed for semantic segmentation based on convolutional layers. One of the most popular architectures for this task is the U-Net, which was introduced by Ronneberger et al. [5]. The U-Net architecture consists of an encoder and a symmetrical decoder. The encoder path contains two convolutional layers in each downward step, followed by a 3 × 3 convolution operation. A pooling operation with a 2 × 2 size and stride size is also implemented to reduce the size of the feature maps. This process is repeated four times to collect the spatial characteristics of the image. After the UpSample operation, the feature maps are placed in the decoder for feature mapping using 2 × 2 transpose convolution. The number of channels is reduced by half, and two convolution layers with the dimensions of 3 × 3 are placed after the UpSample operation. The feature map of the last decoder block is subjected to the 1 × 1 convolution operation, which produces a segmentation map of the same size as the input image. The U-Net architecture uses the ReLU activation function in its convolution layer; however, the last convolution layer uses the sigmoid function. Figure 2 shows the architecture of the proposed method.

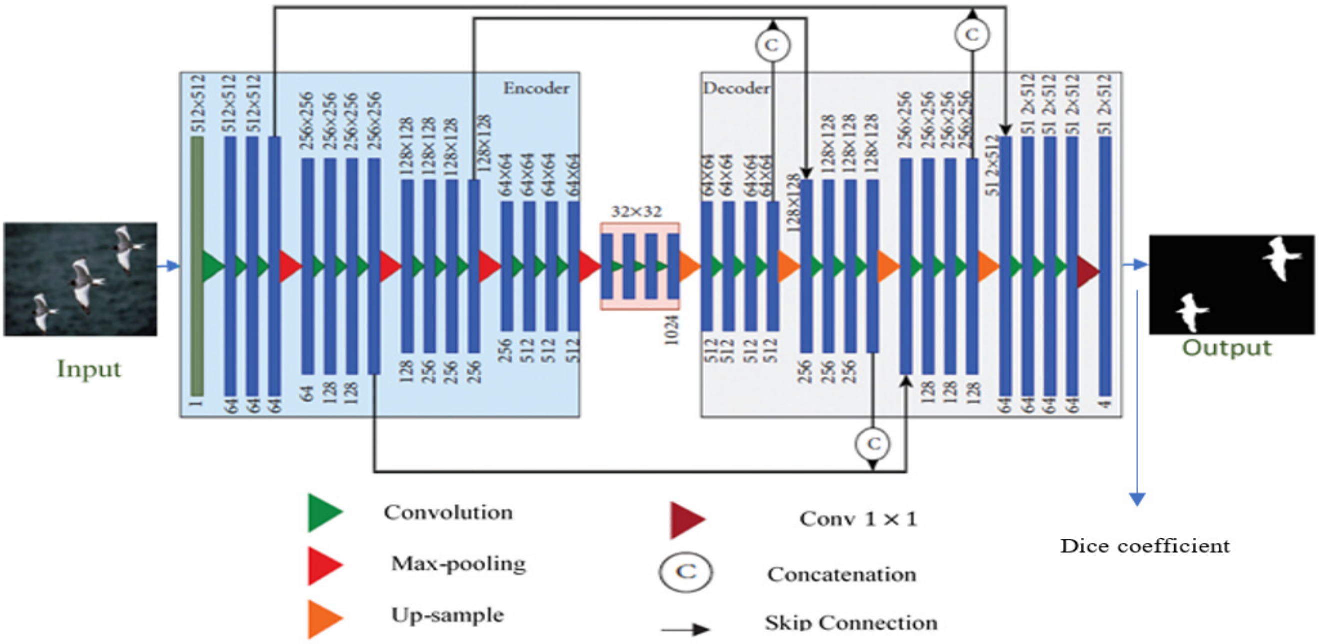


Figure 2. U-NET network architecture inspired by Ronneberger et al. [5].

The proposed model is based on the conventional U-Net architecture with several improvements. Firstly, convolution layers are added to each block of the encoder structure, and the same number of convolution layers are also added to each decoder block. Both the contraction and expansion paths have a more extensive set of weights than U-Net. Secondly, the weights in the encoder and decoder that connect the module have been increased from 2 to 3. Thirdly, the improved model uses batch normalization before any nonlinear function, whereas conventional U-Net does not use batch normalization in its network. Finally, the U-Net network's parameters are optimized using the grasshopper optimization algorithm.

Adding more convolutional layers to each block of the encoder and decoder as well, increasing the set of weights in the encoder and decoder, and using batch normalization before nonlinear functions can improve the performance of the model. The grasshopper optimization algorithm is used to optimize the U-Net network parameters, which can further improve the performance of the model.

In this research, we increased the number of convolutional layers in each encoder and decoder block to handle the input image dimensions of 512 × 512. We used a modified U-Net structure, which adds a convolutional layer to each block of the encoder and decoder pipelines. The proposed model consists of four composite blocks forming the downsampling path, with the fourth block forming the upsampling path. In each encoder and decoder block, we applied two-dimensional (2D) convolutions with a kernel size of 3 × 3, followed by batch normalization and the ReLU activation function, repeated three times. The last block uses a 2D convolution with a kernel size of 1 × 1. To reduce the spatial dimensions of the feature maps after each block, a max-pooling operation was performed during the downsampling path. In the upsampling path, we used ConvTranspose2d to double the spatial dimensions of the feature maps. In the downsampling path, the number of feature channels increased from 1 to 64, 256, and 512, and then reduced from 512 channels to 1 in the upsampling path. Figure 3 illustrates the flow of the input images in the encoder pipeline, indicating which operation and how many convolutional layers are used in each block.

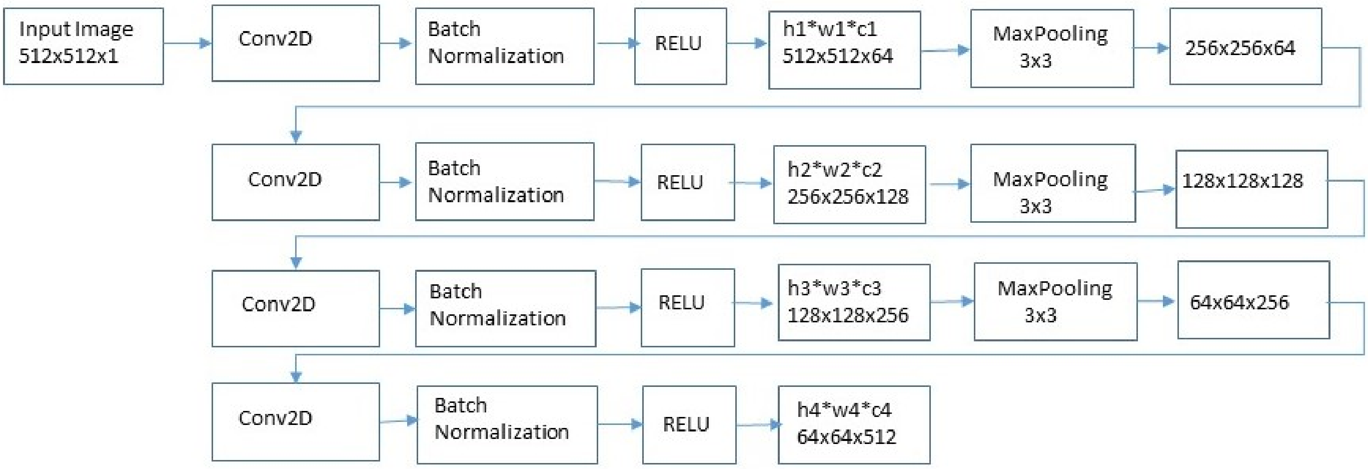


Figure 3. Encoder Flow Diagram.

2.2.1. Cross-Entropy (XE) Training

In the first step, two-dimensional (2D) convolution is applied to the input image along with batch normalization and the nonlinear ReLU function, which reduces the number of channels. Then, the max-pooling operation is performed to obtain the pooled (integrated) feature map. Downsampling reduces the size of the input, and this sequence of operations is repeated four times in the encoder path.

The output of the last encoder layer with smaller dimensions is considered as the input of the connected module with more weight. In this way, a local representation of the feature map generated from the last block of the proposed encoder path is obtained. The integrated module is placed in the bottleneck layer of U-Net because the input transmitted in the module reduces in size and dimension, which decreases the amount of time required for training and the complexity of the space in the feature maps. The bottleneck layer is a part of the encoder–decoder architecture proposed for image forgery detection. The bottleneck layer is placed after the last encoder layer and before the decoder layers in U-Net. The input to the bottleneck layer is the output of the last encoder layer with smaller dimensions. The proposed model improves the image forgery dataset's segmentation efficiency by increasing the convolutional layers and modifying the connected encoder–decoder added to the model. Figure 4 shows the encoder–decoder connection module. The last merged feature map enters the module, which is a convolutional layer with a filter size of 3 × 3.

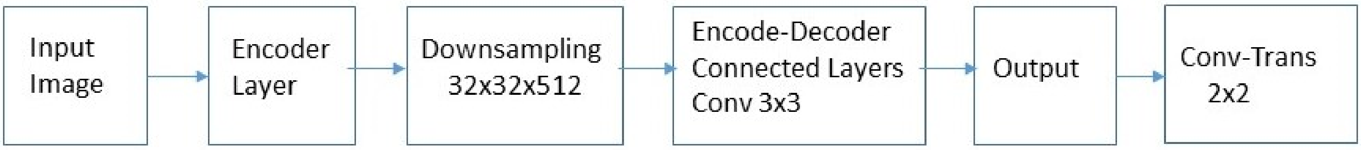


Figure 4. Encoder connection module.

The feature map of the last layer in the proposed model is in accordance with the SoftMax function, which produces the number of feature channels equal to the number of semantic segmentation label classes. In the last layer, an image with a size of 512 × 512 is obtained, which is equal to the image's original dimensions. Each convolutional layer comprises feature maps, and C is the convolutional layer. Equation (1) expresses the first layer's convolution with its constituents:

$$C_j^{(1)} = f\left(B_j^{(1)} + K_j^{(1)} \times I\right) \; where \; j = 1, \ldots, M^{(n)} \tag{1}$$

Here, j is the number of feature maps, and f(y) is the linear function applied to the filtered output before passing it to the convolution layer; I represents the input neuron; M denotes the feature map, and n denotes the number of layers. The feature map in the first layer $C^{(1)}$ is obtained by convolution of the input matrix in the kernel $K^{(1)}$ and adding the bias component $B^{(1)}$.

In the first convolution layer, the input is convoluted with the weight matrix to obtain a feature map. The feature map is obtained by sliding over different positions of the input matrix based on the value of the stride setting. Features, thus extracted, and the weight parameters are shared between all classes of the dataset. Therefore, this layer has a property equal to the variance and is unchangeable to image transformations.

The loss function based on the Dice coefficient uses every 200 cycles to express the Dice loss, which is also included in the evaluation matrix given in Equation (2).

$$Dice\ loss = 1 - dice(a, c) = 1 - 2 \times \frac{a \cap c}{a + c} \tag{2}$$

In Equation (2), a represents the ground truth or target binary mask, and c represents the predicted binary mask produced by the model. The intersection of a and c represents the number of pixels that are correctly classified as the object of interest in both the ground truth and predicted masks. The sum of a and c represents the total number of pixels classified as the object of interest in both masks. The Dice coefficient loss function is implemented after the final upsampling layer in the decoder part of the U-Net network. The final upsampling layer produces a predicted segmentation mask, which is compared to the ground truth mask using the Dice coefficient loss function. The gradients of this loss are then used to update the weights of the U-Net model during backpropagation. The last layer of the proposed model follows the U-NET method, which produces the number of feature channels equal to the number of semantic segmentation label classes. Our dataset contains classes of forged and intact images; therefore, we used the SoftMax function in the last layer of the U-Net, and it produces an output image with a size of 512 × 512, which is equal to the size of the input image A mathematical operation called SoftMax converts a vector of real values into a probability distribution. The U-Net convolutional network is constructed with all defined layers and finally trained. The parameters of this model are optimized by the grasshopper algorithm.

2.2.2. Hyperparameter Tuning Using Grasshopper Optimization Algorithm (GOA)

In this section, the U-NET network with the grasshopper optimization method is designed to detect image forgery.

Through a series of experiments and evaluations, the effectiveness of the proposed method will be demonstrated. Figure 5 shows the implementation of the grasshopper optimization algorithm to optimize the hyperparameters of the U-Net network. The algorithm involves several steps, including initializing the search agents, calculating the merit of each grasshopper, assigning optimal hyperparameters to U-NET networks, evaluating the U-NET networks using mean squared error (MSE), updating the position of grasshoppers, and checking whether the stop condition is met. If the stop condition is met, the best U-NET network with the minimum MSE is returned, and the results are processed and visually compared. If the stop condition is not met, the merit of each grasshopper is calculated again, and the algorithm continues until the optimal hyperparameters are found. The grasshopper optimization algorithm is used to explore the search space of hyperparameters and find the optimal combination that leads to the best U-NET network.

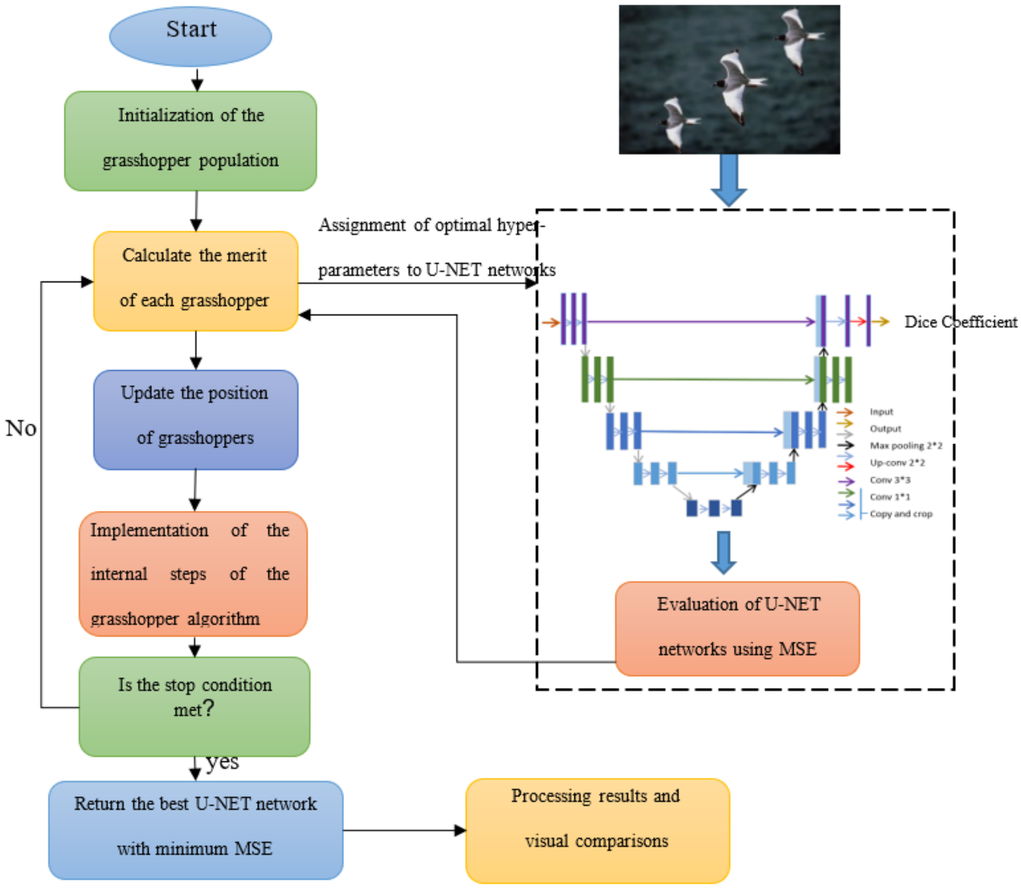

Start
Initialization of the grasshopper population
Calculate the merit of each grasshopper
Assignment of optimal hyper-parameters to U-NET networks
Update the position of grasshoppers
No
Implementation of the internal steps of the grasshopper algorithm
Is the stop condition met?
yes
Return the best U-NET network with minimum MSE
Processing results and visual comparisons
Dice Coefficient
Input
Output
Max pooling 2*2
Up-conv 2*2
Conv 3*3
Conv 1*1
Copy and crop
Evaluation of U-NET networks using MSE

Figure 5. The process of using the grasshopper algorithm in the proposed method.

2.3. Datasets

The Chinese Academy of Sciences Institute of Automation (CASIA) image tempering detection evaluation dataset [44] has been used in our study to test the performance of the U-net model processed via our proposed methodology. This dataset contains a wide variety of images, including linking forgery and copy and transfer forgery, which makes it an excellent choice for evaluating the effectiveness of our proposed approach.

The images in this dataset are classified into various categories based on their content, including nature, animals, architecture, people, plants, objects, scenes, and textured images. The use of the CASIA dataset provides a comprehensive and challenging evaluation of our proposed method and allows us to determine its effectiveness in detecting forgeries across a range of different image types and content. This enables us to determine the method's effectiveness in detecting forgeries across different image categories.

This dataset is widely considered to be a more challenging and realistic dataset for manipulation detection, making it an ideal choice for evaluating the effectiveness of our method in real-world scenarios. Table 1 provides an overview of the CASIA v1.0 and CASIA v2.0 datasets that were used in the experiments conducted for the proposed method.

Table 1. Overview of CASIA v1.0 and CASIA v2.0 [44].

Dataset	Image Type	Image Size	Authentic	Spliced
CASIA v1.0	jpg	384 × 256	800	921
CASIA v2.0	jpg tif bmp	240 × 160 900 × 600	7491	5123

Table 2 provides statistical data on the types of spliced images that were created and how they were arranged in the CASIA V1.0 dataset. The table is organized into several categories, including the number of images in JPEG format, the types of manipulations used, the source of the tampered regions, and the shape of the tampered regions. Table 2 is helpful in understanding the characteristics of the spliced images in the CASIA V1.0 dataset and the way they were manipulated using different techniques.

Table 2. Some statistical information about the spliced images in CAISA ITDE V1.0.

Category		No. of Images
JPEG Format		921
Manipulation without pre-processing		562
Source of Tampered Region(s)	Same Image	451
	Different Images	470
Manipulation with pre-processing	Rotation	25
	Resize	206
	Distortion	53
	Rotation and Resize	45
	Resize and Distortion	27
	Rotation and Distortion	3
	Rotation, Distortion and Resize	0
Shape of Tampered Region	Circular boundary	114
	Rectangular boundary	169
	Triangular boundary	102
	Arbitrary boundary	536

The inclusion of more complex tampering techniques in CASIA v2.0 provides a more challenging and realistic evaluation of the proposed method's effectiveness in detecting forgeries in real-world scenarios. Overall, the use of these datasets provides a rigorous and comprehensive evaluation of the proposed method, enabling us to determine its effectiveness in detecting forgeries across a range of different tampering techniques and content categories. V2.0 considers the following criteria while generating tampered images:

- Illumination changes to spliced regions;
- Splicing with blurring;
- Copy-move;
- Text insertion;
- Image retouching.

Table 3 provides statistical information about the spliced images in the CAISA v2.0 database. The table shows the number of images in various categories, such as the format of the images, the type of manipulations applied to the images, the source of the tampered regions, the size of the tampered regions, and so on. Furthermore, the table provides information on the type of manipulations applied to the images, such as rotation, resizing, and distortion, either alone or in combination. It also displays the number of images with post-processing blurring, either along spliced edges or other regions.

Table 3. Some statistical information about the spliced images in CAISA ITDE V2.

Category		**No. of Images**
JPEG Format		2064
TIFF Format		3059
Manipulation without pre-processing		1843
Manipulation without post-processing (blurring)		4144
Source of Tampered Region(s)	Same Image	3274
	Different Images	1849
Manipulation with pre-processing	Rotation	568
	Resize	1648
	Distortion	196
	Rotation and Resize	532
	Resize and Distortion	211
	Rotation and Distortion	42
	Rotation, Distortion and Resize	83
Manipulation with post-processing	Blurring along spliced edges	848
	Blurring on other regions	131
Size of Tampered Region	Small	3358
	Medium	819
	Large	946

An example of one of the database photos utilized in the CASIA v1.0 dataset may be seen in Figure 6.

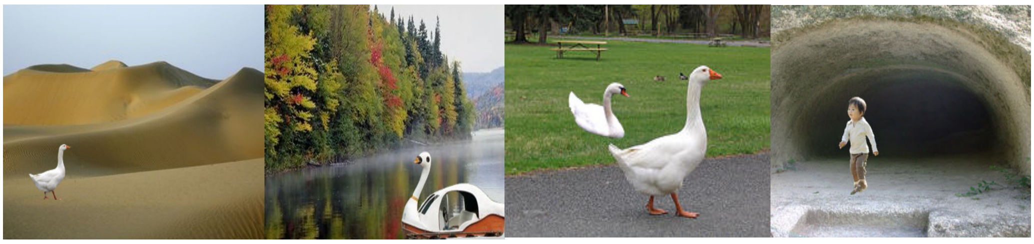


Figure 6. An example of database images [45].

2.4. Training Strategy

Figures 7 and 8 illustrate the generation of tampered images within the CASIA V1.0 dataset. The images in the dataset were tampered with using a number of tampering approaches, but only splicing was taken into account in the CASIA database version 1.0. Therefore, the altered images within the dataset are known as spliced images. Figure 8 demonstrates another instance of image splicing in the V1.0 dataset. Panel (a) of the figure depicts an authentic image of a landscape, while panel (b) depicts a composite image created by copying and pasting a portion of the landscape onto a different background. This technique is used to create altered images that are visually difficult to detect, as they may appear authentic at first glance.

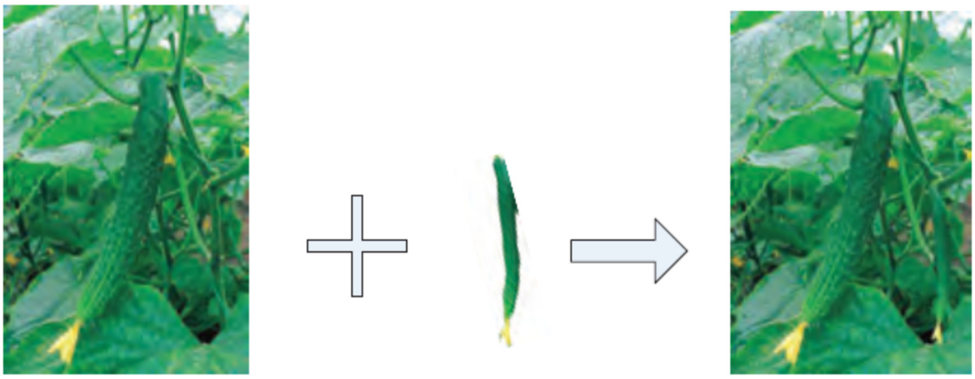
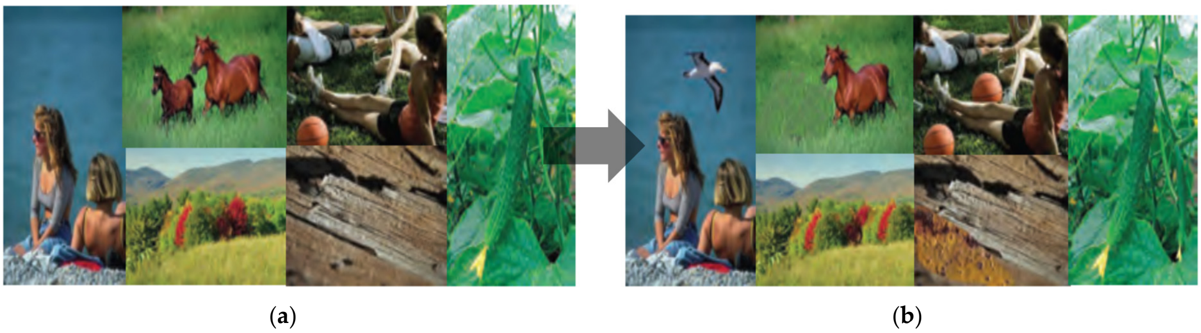


Figure 7. An illustration of producing manipulated images within the CASIA V1.0 dataset [46].



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

Figure 8. This is an example of how to generate tampered photos using CASIA V1.0: (**a**) authentic photos; (**b**) spliced images [47].

CASIA v2.0 is a supplement to CASIA v1.0, containing additional tampered images. This dataset includes images with more complex tampering techniques, such as local contrast enhancement and color modification. The images in this dataset are also classified into the same eight content categories as CASIA v1.0.

The use of these datasets allows for a comprehensive evaluation of the proposed method across a wide range of tampering techniques and content categories. The inclusion of more complex tampering techniques in CASIA v2.0 provides a more challenging and realistic evaluation of the proposed method's effectiveness in detecting forgeries in real-world scenarios.

Figure 9 shows how tampered images were generated in CASIA V2.0. In this example, a tampered image was created by splicing two different regions from different images. One region was taken from an image of a statue, while the other was taken from an image of a sky. The spliced region was then inserted into a new background image, resulting in a tampered image.

The most significant difference between the V1.0 and V2.0 tampered sets is the blurring operation usage after generating a spliced image. The database also includes several manually tampered images to make it more comprehensive. V2.0 considers the following criteria while generating tampered images:

- Illumination changes to spliced regions;
- Splicing with blurring;
- Copy-move;
- Text insertion;
- Image retouching.

Figure 10 illustrates an example of generating tampered photos in CASIA V2.0. In this example, (a) displays the legitimate photographs, and (b) displays the tampered images that match them.

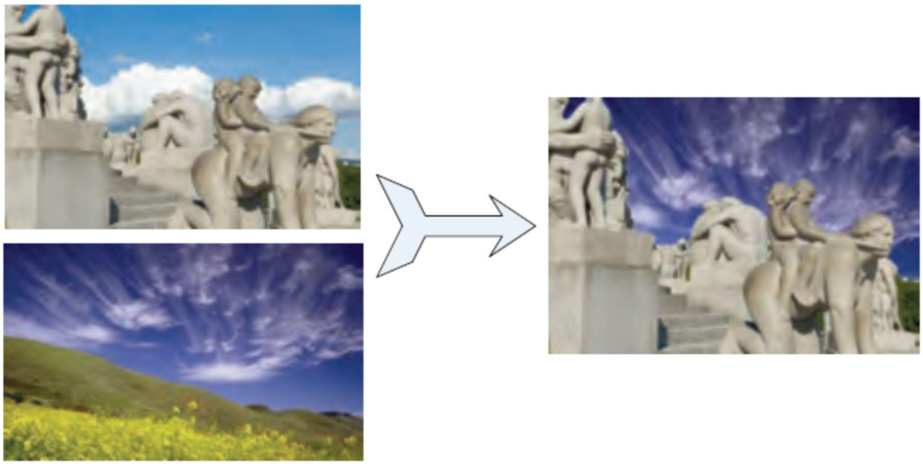
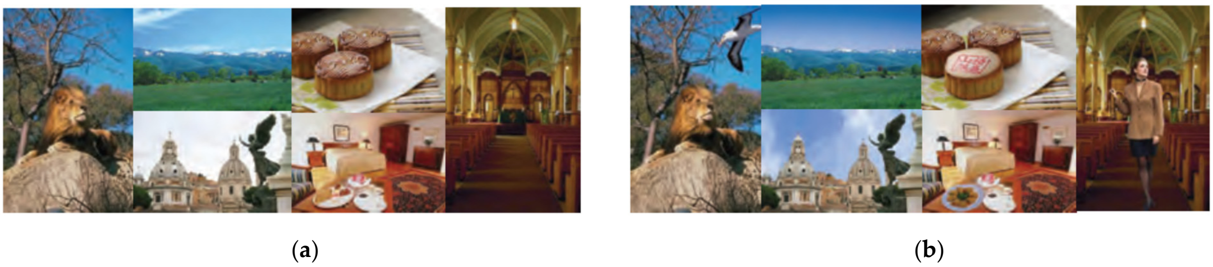


Figure 9. An illustration of generating tampered images with CASIA v2.0 [45].



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

Figure 10. An illustration of the production of manipulated images in CASIA V2.0 is presented, where (**a**) denotes authentic images and (**b**) represents tampered images [47].

3. Results

As the usage of digital image editing tools has made it simpler to change photos, digital image fraud detection is an important topic of research. When it comes to legal and forensic investigations, this manipulation might have catastrophic repercussions. The usage of social media and internet platforms has increased the possibility of picture alteration and fraud, necessitating the development of effective forgery detection methods. Numerous techniques have been suggested for the identification of manipulated images, such as the implementation of D-Net [25], CNN–HPF [35], RRU-Net [48], Mask R-CNN + Sobel filter [35], a CNN based on the pre-trained AlexNet model [49], and Deep Neural Architecture-BusterNet [50]. These techniques employ a variety of strategies to identify faked images, including deep learning, a convolutional neural network and high-pass filter combo, and residual analysis.

In our study, we propose a modification of an approach that incorporates a grasshopper optimization algorithm (GOA) to overcome the limitations of the existing methods. The GOA is a metaheuristic algorithm that effectively solves optimization problems. By incorporating the GOA, we aimed to improve the accuracy and speed of the forgery detection model.

During the experimentation, we assigned different subsets of the dataset with different training–validation ratios, which led to an improvement in the model's performance. This modification enhances the model's ability to learn and adapt to different types of image forgeries, making it more efficient in detecting and localizing image manipulations. Additionally, the use of the GOA improves the overall accuracy and speed of the forgery detection model. This paper presents an approach for detecting image forgery using a U-NET network optimized with the grasshopper optimization algorithm (GOA). We split the dataset into training, validation, and test sets and used the ratio comparison method to compare the performance of our proposed method with existing methods.

After training, the model's accuracy on the test set was 100 percent. These results indicate that the model is exceptionally effective at detecting image manipulation. Our forgery detection model was based on the U-Net network architecture. We trained the network using the Dice coefficient as the loss function and the Adam optimizer. The model was trained for 200 epochs. During the training phase, the grasshopper optimization algorithm was utilized to optimize the model's hyperparameters, including the learning rate, momentum, and weight decay. The validation set was used to evaluate the performance of the model during training, and the model with the highest validation accuracy was chosen for testing.

Tables 4–6 present the validation accuracy of the CASIA dataset. Among all existing models, mostly CNNs based on the pre-trained Alex Net model had the best and the most stable performance by achieving 91.6%, 93.94%, and again 93.94% accuracy on the validation set with train/validation ratios of 7:3, 8:2, and 9:1, respectively. The best accuracy was achieved for the training/validation ratios 8:2 and 9:1.

Table 4. Validation accuracy of each epoch during training with train/validation ratio = 7:3.

	Methods	**Train/Valid = 7:3** **Accuracy of Validation Set for Each Epoch during the Training Process (%)**									
		E 1	**E 2**	**E 3**	**E 4**	**E 5**	**E 6**	**E 7**	**E 8**	**E 9**	**E 10**
	D-Net [33]	66.22	72.31	68.01	73.45	75.37	83.8	80.23	83.8	82.43	81.88
	CNN–HPF [43]	77.1	75.22	77.94	75.22	77.8	88.76	85.6	86.07	89.22	89.22
	RRU-Net [45]	62.71	67.8	73.41	75.4	73.4	76.13	73.4	76.01	75.5	75.5
	Mask R-CNN + Sobel filter [43]	65.07	77.91	76.33	80.53	76.49	77.97	78.62	75.58	77.78	80.53
	CNN based on pre-trained Alex Net model [46]	76.91	85.97	88.84	90.05	90.05	88.29	91.63	90.29	88.66	91.63
	Deep Neural Architecture-Buster Net [47]	63.21	66.8	73.97	70.92	73.97	70.92	70.38	71.34	73.97	71.73
Modified	D-Net	77.5	85.3	88.91	87.16	86.04	88.91	88.74	87.1	87.1	87.98
	CNN–HPF	80.11	91.24	89.58	90.28	90.92	92.43	90.82	90.32	90.03	91.71
	RRU-Net	66.77	78.19	75.89	75.14	75.47	77.13	74.91	78.19	76.13	77.13
	Mask R-CNN + Sobel filter	70.7	81.47	82.12	79.69	82.12	80.01	82.03	80.92	81.83	80.86
	CNN based on pre-trained Alex Net model	81.46	85.71	91.28	92.75	92.8	91.28	92.75	92.09	92.64	92.75
	Deep Neural Architecture-Buster Net.	67.86	75.77	82.46	80.69	80.78	82.28	78.17	79.81	79.56	82.83
	U-NET network optimized with Grasshopper	78.78	85.75	90.57	92.3	92.34	94.63	93.57	93.47	93.67	94.63

In the modified models, a CNN based on the pre-trained Alex Net model maintains its stability and further improves accuracy up to 92.75%, 94.49%, and 95.38% with train/validation ratios equal to 7:3, 8:2, and 9:1, respectively. The best accuracy was achieved with a training/validation ratio of 9:1. The U-Net model with GOA provides the best performance of 94.63% for a training/validation ratio of 7:3. The accuracies achieved by the modified models are higher than the original models; their general performance also varies with different train/validation ratios. During the whole training process, modified models tend to have a better performance with an obvious increase in accuracy for every epoch as compared to the raw models.

Table 5 shows that the best accuracy of 93.94% by a CNN pre-trained on AlexNet, and the modified version improves the accuracy of the CNN to 94.49% for a training/validation ratio of 8:2. U-Net with GOA outperforms all models with an accuracy of 95.31% for a training/validation ratio of 8:2.

Table 5. Validation accuracy of each epoch during training with train/validation ratio = 8:2.

	Methods	**Train/Valid = 8:2** **Accuracy of Validation Set for Each Epoch during the Training Process (%)**									
		E 1	**E 2**	**E 3**	**E 4**	**E 5**	**E 6**	**E 7**	**E 8**	**E 9**	**E 10**
	D-Net [33]	75.73	77.36	85.8	80.54	84.04	85.8	82.23	81.96	83.14	85.01
	CNN–HPF [43]	79.58	85.24	89.2	92.3	90.96	89.2	90.72	92.3	90.87	91.2
	RRU-Net [45]	63.64	74.73	72.18	75.4	74.19	76.01	76.01	74.83	73.5	75.4
	Mask R-CNN + Sobel filter [43]	66.61	77.13	76.98	78.16	77.81	78.09	76.5	77.56	76.5	78.16
	CNN based on pre-trained Alex Net model [46]	77.69	86.62	89.21	92.37	91.86	89.96	93.41	93.94	93.94	92.3
	Deep Neural Architecture-Buster Net [47]	63.86	75.12	76.29	72.04	77.49	71.69	74.03	70.97	76.68	77.49
Modified	D-Net	77.06	82.34	87.94	87.94	82.82	89.56	87.5	87.5	85.88	89.56
	CNN–HPF	80.5	85.8	90.36	92.06	91.57	92.1	93.14	92.1	90.98	93.08
	RRU-Net	68.44	75.6	77.67	78.95	78.95	77.67	77.5	78.23	77.5	78.8
	Mask R-CNN + Sobel filter	70.47	75.25	81.61	81.12	83.5	82.33	79.98	83.5	83.5	81.98
	CNN based on pre-trained Alex Net model	83.24	89.36	90.65	90.95	93.12	90.23	93.12	94.49	93.76	94.49
	Deep Neural Architecture-Buster Net.	68.51	82.12	80.09	83.01	81.73	80.91	83.01	78.69	78.69	81.46
	U-NET network optimized with Grasshopper	79.55	86.38	92.50	90.65	93.96	95.31	92.45	95.14	95.31	95.26

Table 6 shows that a CNN pre-trained on AlexNet provides the best accuracy of 93.94%, and the modified version improves the accuracy to 95.38% for a training/validation ratio of 9:1. U-Net with GOA outperforms all models with an accuracy of 97.98%.

Table 6. Validation accuracy of each epoch during training with train/validation ratio = 9:1.

	Methods	**Train/Valid = 9:1** **Accuracy of Validation Set for Each Epoch during the Training Process (%)**									
		E 1	**E 2**	**E 3**	**E 4**	**E 5**	**E 6**	**E 7**	**E 8**	**E 9**	**E 10**
	D-Net [33]	75.73	77.36	85.8	80.54	84.04	85.8	82.23	81.96	83.14	85.01
	CNN–HPF [43]	79.58	85.24	89.2	92.3	90.96	89.2	90.72	92.3	90.87	91.2
	RRU-Net [45]	63.64	74.73	72.18	75.4	74.19	76.01	76.01	74.83	73.5	75.4
	Mask R-CNN + Sobel filter [43]	66.61	77.13	76.98	78.16	77.81	78.09	76.5	77.56	76.5	78.16
	CNN based on pre-trained Alex Net model [46]	77.69	86.62	89.21	92.37	91.86	89.96	93.41	93.94	93.94	92.3
	Deep Neural Architecture-Buster Net [47]	63.86	75.12	76.29	72.04	77.49	71.69	74.03	70.97	76.68	77.49
Modified	D-Net	78.03	85.89	89.4	89.65	86.94	90.87	90.96	87.87	85.44	90.89
	CNN–HPF	82.46	90.81	91.33	92.72	92.82	95.81	93.31	93.59	95.81	92.34
	RRU-Net	67.02	78.39	80.58	80.11	80.62	81.56	78.43	80.21	81.56	81.56
	Mask R-CNN + Sobel filter	74.04	83.99	81.09	85.54	80.71	83.19	83.11	85.54	85.54	83.96
	CNN based on pre-trained Alex Net model	84.96	89.35	92.13	94.27	95.38	91.99	94.73	95.38	94.97	95.38
	Deep Neural Architecture-Buster Net.	71.65	81.89	81.65	86.33	82.2	83.19	79.39	83.19	82.24	86.33
	U-NET network optimized with Grasshopper	85.68	89.6	94.45	92.58	92.45	97.98	94.45	97.98	95.6	97.98

Table 7 shows the validation and test accuracy for the training/validation ratios 7:3, 8:2, and 9:1. Although the test accuracy shows an improvement over validation accuracy, there are some inconsistencies. For example, in D-Net, the validation accuracy for the training/validation ratio of 7:3 is 83.8%; it reduces to 76.3% for test accuracy. However, it can be observed that the highest accuracy of 95.4% is achieved by a CNN pre-trained on the AlexNet model with a training/validation ratio of 9:1, whereas, for the modified version, the highest accuracy of 95.7% is achieved with an 8:2 training/validation ratio. U-Net with GOA outperforms all models with 100% test accuracy for a training/validation ratio of 8:2.

Table 7. Validation and test accuracy results.

	Methods	**Train/Valid Ratio = 7:3**		**Train/Valid Ratio = 8:2**		**Train/Valid Ratio = 9:1**	
		Valid Accuracy	**Test Accuracy**	**Valid Accuracy**	**Test Accuracy**	**Valid Accuracy**	**Test Accuracy**
	D-Net [33]	83.80%	76.30%	85.80%	87.5%	86.81%	87.64%
	CNN–HPF [43]	89.22%	89.22%	92.30%	91.80%	93.73%	75.40%
	RRU-Net [45]	76.13%	78.63%	76.01%	79.51%	78.75%	79.58%
	Mask R-CNN + Sobel filter [43]	80.53%	80.53%	78.16%	64.50%	81.28%	75.98%
	CNN based on pre-trained Alex Net model [46]	91.63%	89.13%	93.94%	80.94%	94.70%	95.40%
	Deep Neural Architecture-Buster Net [47]	73.97%	61.47%	77.49%	64.49%	84.37%	78.54%
Modified	D-Net	88.91%	95.50%	89.56%	95.06%	90.96%	93.85%
	CNN–HPF	92.43%	96.58%	93.14%	94.64%	95.81%	95.95%
	RRU-Net	78.19%	88.19%	78.95%	88.95%	81.56%	85.73%
	Mask R-CNN + Sobel filter	82.12%	85.62%	83.50%	93.50%	85.54%	87.21%
	CNN based on pre-trained Alex Net model	92.75%	93.25%	94.49%	95.70%	95.38%	95.38%
	Deep Neural Architecture-Buster Net.	82.46%	94.96%	83.01%	96.51%	86.33%	93.63%
	U-NET network optimized with Grasshopper	94.63%	98.13%	95.31%	100.00%	97.98%	97.85%

Figures 11–13 show the bar graphs to compare the accuracy for the validation and test ratios of 7:3, 8:2, and 9:1, respectively.

Table 8 shows the specificity, recall, precision, F1 score, and AUC for all the models and the modified models for a training/validation ratio of 7:3. The CNN pre-trained with AlexNet provides the best F1-score and AUC values. However, in the modified versions, the Mask R-CNN with Sobel filter provides the best F1-score and AUC values. U-Net with GOA outperforms all the models.

Table 9 shows the specificity, recall, precision, F1 score, and AUC for all the models and the modified models for a training/validation ratio of 8:2. A CNN pre-trained with AlexNet provides the best AUC of 1.0 and CNN–HPF provides the best F1 score of 0.926. However, in the modified versions, the Mask RCNN with Sobel filter provides the best F1-score and AUC values. U-Net with GOA outperforms all the models with an F1 score of 0.991.

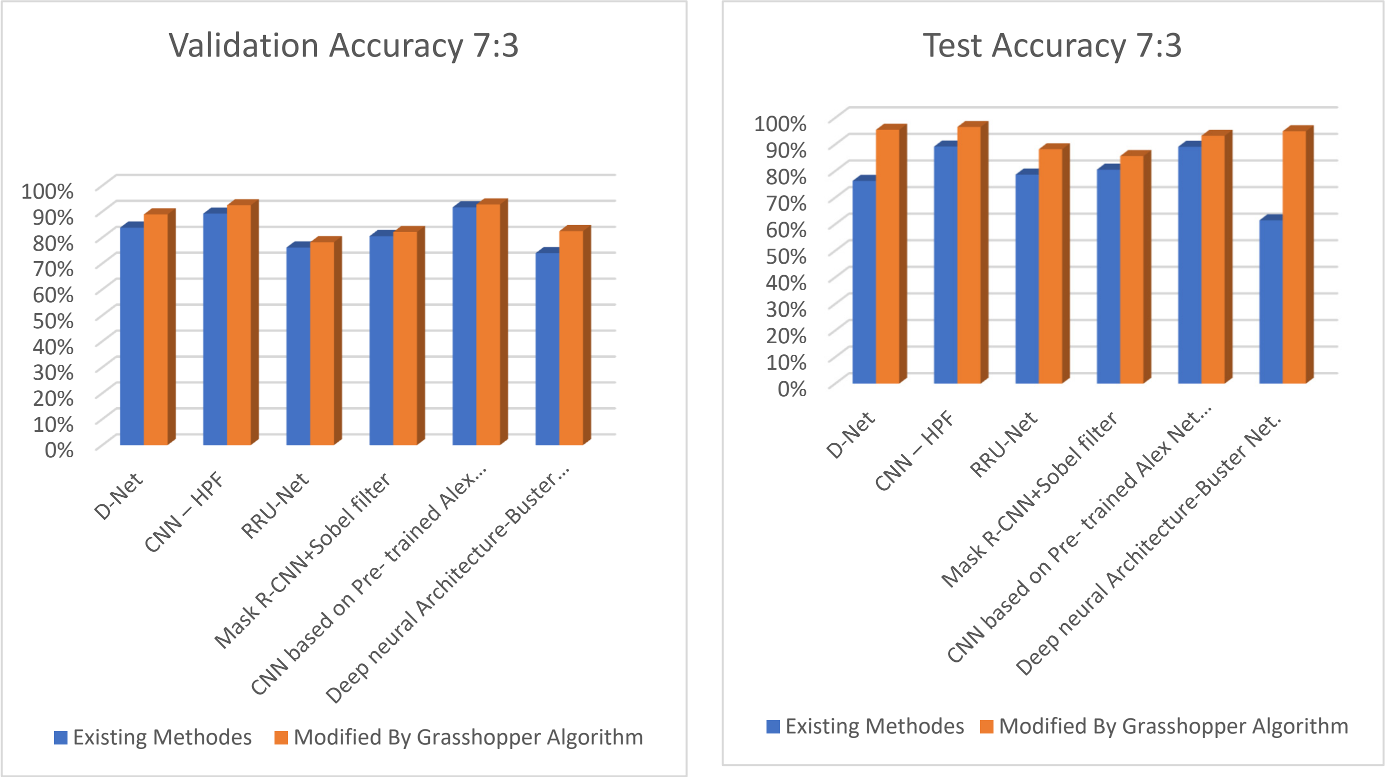


Figure 11. Validation and test accuracy results with a 7:3 validation/test ratio.

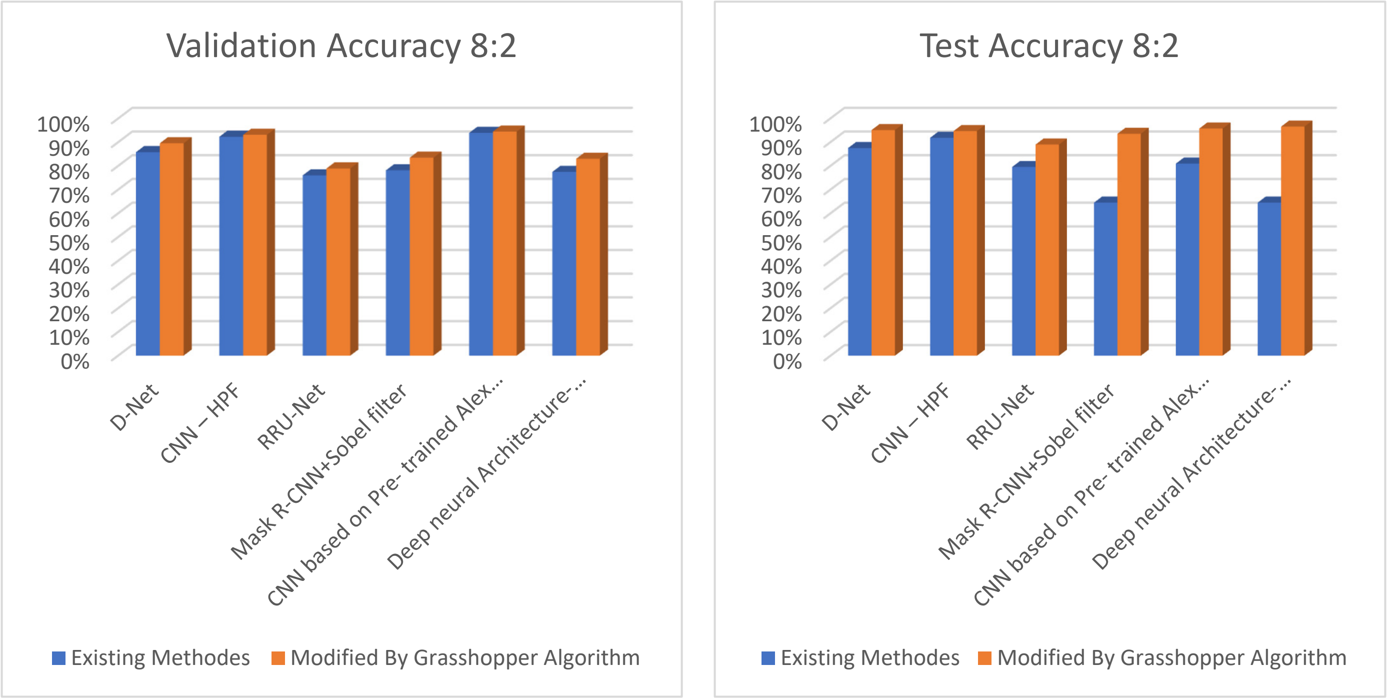


Figure 12. Validation and test accuracy results with an 8:2 validation/test ratio.

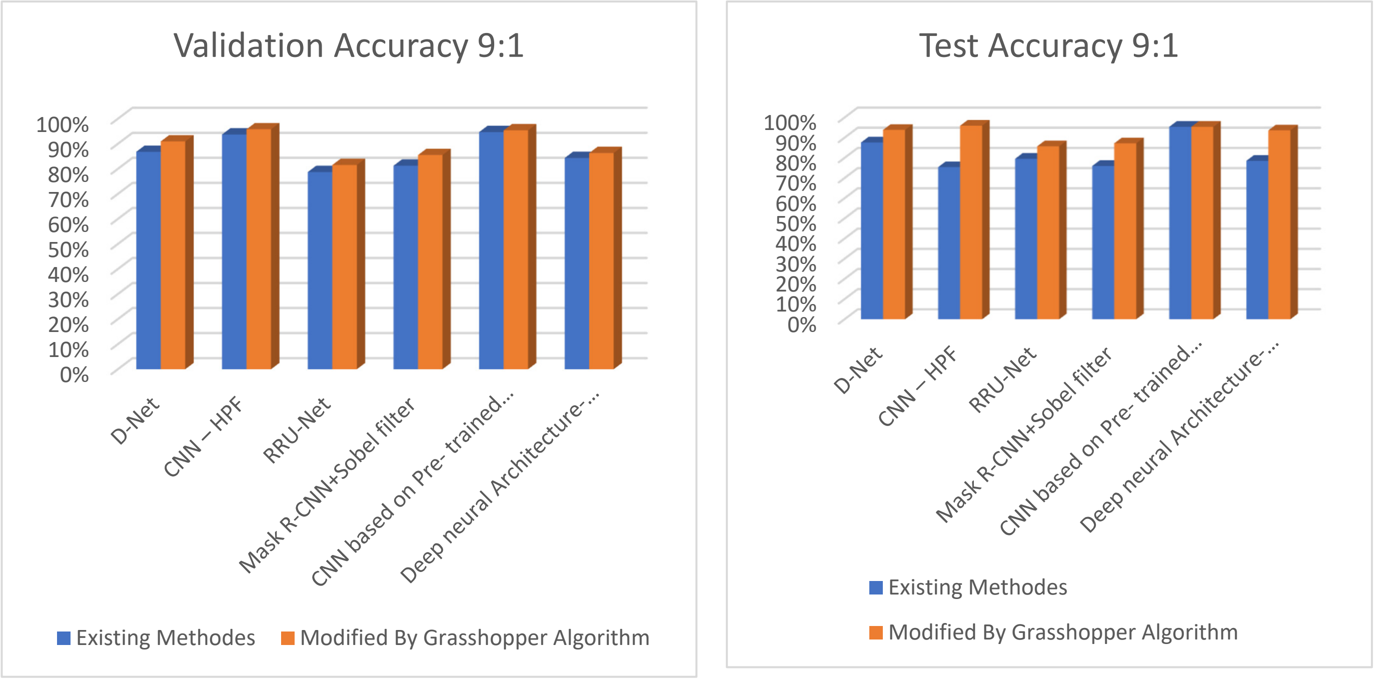


Figure 13. Validation and test accuracy results with a 9:1 validation/test ratio.

Table 8. Statistical model analyses with train/validation ratio = 7:3.

		Train/Valid Ratio = 7:3				
		Specificity	**Recall**	**Precision**	**F1 Score**	**AUC**
D-Net		0.913	0.706	0.857	0.774	0.926
CNN–HPF		0.913	0.824	0.875	0.848	0.941
RRU-Net		0.733	1	0.714	0.833	0.81
Mask R-CNN + Sobel filter		0.933	0.8	0.889	0.842	0.927
CNN based on pre-trained Alex Net model		0.913	0.941	0.889	0.914	0.972
Deep Neural Architecture-Buster Net.		0.933	0.811	0.928	0.865	0.932
Modified	D-Net	0.939	0.789	0.893	0.968	0.968
	CNN–HPF	0.939	0.787	0.893	0.923	0.923
	RRU-Net	0.892	0.687	0.803	0.855	0.855
	Mask R-CNN + Sobel filter	0.9	0.937	0.908	0.976	0.976
	CNN based on pre-trained Alex Net model	0.956	0.766	0.918	0.923	0.923
	Deep Neural Architecture-Buster Net.	0.94	0.662	0.875	0.878	0.878
	U-Net with GOA	0.956	0.937	0.957	0.986	0.986

Table 10 shows the specificity, recall, F1 score, precision, and AUC for all the models and the modified models for a training/validation ratio of 9:1. RRU-Net provides the best AUC of 0.98, and the CNN pre-trained on AlexNet provides the best F1 score of 0.924. However, in the modified versions, the Mask RCNN with Sobel filter provides the best F1-score and AUC values. U-Net with GOA outperforms all the models with an F1 score of 0.98.

Table 9. Statistical model analyses with train/validation ratio = 8:2.

		Train/Valid Ratio = 8:2				
		Specificity	**Recall**	**Precision**	**F1 Score**	**AUC**
D-Net		0.94	0.8	0.88	0.82	0.92
CNN–HPF		0.95	0.9	0.955	0.926	0.96
RRU-Net		0.783	0.941	0.762	0.842	0.882
Mask R-CNN + Sobel filter		0.783	1	0.773	0.872	0.923
CNN based on pre-trained Alex Net model		0.9	1	0.833	0.909	1
Deep Neural Architecture-Buster Net.		0.967	0.829	0.967	0.892	0.938
Modified	D-Net	0.95	0.844	0.916	0.904	0.904
	CNN–HPF	0.969	0.82	0.944	0.951	0.951
	RRU-Net	0.941	0.649	0.875	0.922	0.922
	Mask R-CNN + Sobel filter	0.95	0.9	0.955	0.982	0.982
	CNN based on pre-trained Alex Net model	0.944	0.865	0.908	0.965	0.965
	Deep Neural Architecture-Buster Net.	0.946	0.718	0.896	0.946	0.946
	U-Net with GOA	0.9	1	0.833	0.991	0.991

Table 10. Statistical model analyses with train/validation ratio = 9:1.

		Train/Valid Ratio = 9:1				
		Specificity	**Recall**	**Precision**	**F1 Score**	**AUC**
D-Net		0.9	0.8	0.8	0.8	0.928
CNN–HPF		0.867	0.9	0.818	0.857	0.978
RRU-Net		0.8	1	0.714	0.833	0.88
Mask R-CNN + Sobel filter		0.87	0.941	0.842	0.889	0.928
CNN based on pre-trained Alex Net model		0.961	0.911	0.937	0.924	1
Deep Neural Architecture-Buster Net.		0.967	0.857	0.968	0.909	0.938
Modified	D-Net	0.977	0.887	0.961	0.95	0.924
	CNN–HPF	0.965	0.89	0.943	0.951	0.923
	RRU-Net	0.925	0.793	0.872	0.875	0.875
	Mask R-CNN + Sobel filter	0.933	0.943	0.943	0.978	0.978
	CNN based on pre-trained Alex Net model	0.974	0.883	0.956	0.954	0.954
	Deep Neural Architecture-Buster Net.	0.962	0.818	0.933	0.905	0.905
	U-Net with GOA	0.967	0.986	0.972	0.98	0.98

Table 11 shows the resulting analyses of detecting image forgery on the CASIA dataset using different models. The table includes the train/valid ratio, highest accuracy, and highest AUC for each model. The CNN–HPF model obtained the highest accuracy of 93.73% and the highest AUC of 0.978 for the 9:1 train/valid ratio. The CNN based on the pre-trained Alex Net model achieved the highest accuracy of 95.40% and the highest AUC of 1.00. The U-Net with GOA achieved 100.00% accuracy and an AUC of 0.991.

Table 11. Result analysis of detecting image forgery on CASIA dataset.

		Train/Valid Ratio	**Highest Accuracy**	**Highest AUC**
D-Net		9:1	87.64%	0.928
CNN–HPF		9:1	93.73%	0.978
RRU-Net		9:1	79.58%	0.882
Mask R-CNN + Sobel filter		9:1	81.28%	0.928
CNN based on pre-trained Alex Net model		9:1	95.40%	1
Deep Neural Architecture-Buster Net.		9:1	84.37%	0.938
Modified	D-Net	7:3	95.50%	0.968
	CNN–HPF	7:3	96.58%	0.951
	RRU-Net	8:2	88.95%	0.922
	Mask R-CNN + Sobel filter	8:2	93.50%	0.982
	CNN based on pre-trained Alex Net model	8:2	95.70%	0.965
	Deep Neural Architecture-Buster Net.	8:2	96.51%	0.946
	U-Net with GOA	8:2	100.00%	0.991

The modified D-Net obtained the highest accuracy for the 7:3 train/valid ratio with 95.50% and an AUC of 0.968. The CNN–HPF model obtained the highest accuracy with a 96.58%. The Mask R-CNN + Sobel filter attained the highest AUC of 0.982, while the Deep Neural Architecture-Buster Net attained the highest accuracy of 96.51% and the highest AUC of 0.946. In terms of accuracy and area under the curve (AUC), the CNN-based models outperformed other models. The U-Net with GOA attained perfect precision.

Discussion

As can be seen in Table 7, we evaluated six existing methods for image forgery detection, including D-Net, CNN–HPF, RRU-Net, Mask R-CNN+Sobel filter, a CNN based on the pre-trained Alex Net model, and Deep Neural Architecture-Buster Net. We tested these methods on a dataset with different levels of manipulation. The dataset is divided into three sets with different train/valid ratios: 7:3, 8:2, and 9:1. The train/valid ratio refers to the proportion of images used for training and validation. We evaluated the performance of these methods based on their validation accuracy and test accuracy.

The results show that the existing methods have varying degrees of accuracy and efficiency in detecting image forgery. Among them, a CNN based on the pre-trained Alex Net model performs the best in terms of accuracy, with a test accuracy of 95.40% at a train/valid ratio = 9:1. However, its efficiency is not satisfactory, with a valid accuracy of only 94.7%. On the other hand, D-Net and RRU-Net have relatively high efficiencies; however, their accuracy is not as good as that of the CNN based on the pre-trained Alex Net model.

We propose a new method for detecting image forgery using a U-NET network optimized with grasshopper optimization to resolve the limitations of existing methods. U-NET is a prominent deep neural network architecture for image segmentation tasks. It was chosen due to its ability to capture both high-level and low-level image characteristics. Grasshopper optimization was used to optimize the U-NET network by altering hyperparameters, such as the number of filters in each layer, learning rate, and batch size.

4. Conclusions

Nowadays, due to the importance and attractiveness of social media and the increasing expansion of image editing and processing software tools, as well as the significant reduction in the costs of using these tools, forging and manipulating digital images is not only as simple as possible, it is acceptable. It has become the most common type of manipulation and forgery, from the most straightforward and most accessible image editing software used by regular users to the latest methods based on artificial intelligence, such as deep forgery technology that uses the architecture of cross-generative networks. The field of document forgery has also been equipped with deep learning tools. This progress in the tools and intelligence of methods of forgery detection and the authentication of images has continuously faced new challenges and problems.

As mentioned earlier, copy-transfer forgery is the most common and one of the most accessible types of image manipulation. With this method, one or more areas of the image are copied, and after some changes, they are placed in another section of the same image. Counterfeiters perform many pre-processing or post-processing attacks, such as rotation, filtering, compression, rescaling, etc., on these areas to make the act of detecting forgery even more difficult. Therefore, the detection method should resist such changes and geometric transformations. Typically, to detect this type of forgery, small blocks of the image are compared to each other. For this purpose, several methods have been proposed in this field, which, despite the diverse quantity, follow two general approaches: (1) block-based methods that examine all the blocks of the image and (2) methods based on key points, focusing more on the information-rich areas with unique characteristics.

This research introduced a solution based on deep learning to detect a forgery in digital images. The method of designing and implementing the network architecture and the details of the layers of convolutional neural network architecture used were explained. This paper introduced a modified model based on U-NET architecture for correct image segmentation. The proposed model adds complementary convolutional layers in the encoder and decoder pipelines and improves the convolution module to establish a connection between the encoder and decoder, which is a great advantage for extracting features from the last layer of the encoder path and performing segmentation. Adding more sets of weights to a U-NET improves its performance. The results obtained from the proposed model have proved the proposed model's efficiency in segmenting undetectable and interconnected areas, and the performance evaluation criteria have reached a better state than the basic U-NET method. Also, the AUC and accuracy were obtained as 0.99 and 100%, respectively. Hence, the proposed U-NET method has improved image forgery detection.

Author Contributions: Conceptualization, N.G. and K.P.; methodology, N.G. and K.P.; software, N.G.; validation, N.G. and K.P.; formal analysis, N.G.; investigation, N.G.; resources, N.G. and K.P.; data curation, N.G.; writing—original draft preparation, N.G.; writing—review and editing, K.P.; visualization, N.G. and K.P.; supervision, K.P.; project administration, K.P.; funding acquisition, K.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: CASIA dataset available in [44].

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

References

1. Jain, I.; Gooel, N. Advancements in Image Splicing and Copy-move Forgery Detection Techniques: A Survey. In Proceedings of the 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 28–29 January 2021. [CrossRef]
2. Nabi, S.T.; Kumar, M.; Singh, P.; Aggarwal, N.; Kumar, K. A Comprehensive Survey of Image and Video Forgery Techniques: Variants, Challenges, and Future Directions. *Multimed. Syst.* **2022**, *28*, 939–992. [CrossRef]
3. Barad, Z.; Goswami, M.M. Image Forgery Detection using Deep Learning: A Survey. In Proceedings of the 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6–7 March 2020; pp. 571–576.
4. Hebbar, N.K.; Kunte, A.S. Image Forgery Localization Using U-Net based Architecture and Error Level Analysis. In Proceedings of the 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, 17–18 December 2021; pp. 1992–1996. [CrossRef]

5. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Lecture Notes in Computer Science*; Springer International Publishing: Cham, Switzerland, 2015; pp. 234–241. [CrossRef]
6. Abd Warif, N.B.; Wahab, A.W.A.; Idris, M.Y.I.; Ramli, R.; Salleh, R.; Shamshirband, S.; Choo, K.K.R. Copy-move forgery detection: Survey, challenges and future directions. *J. Netw. Comput. Appl.* **2016**, *75*, 259–278. [CrossRef]
7. Zhao, F.; Shi, W.; Qin, B.; Liang, B. Image forgery detection using segmentation and swarm intelligent algorithm. *Wuhan Univ. J. Nat. Sci.* **2017**, *22*, 141–148. [CrossRef]
8. Jalab, H.; Subramaniam, T.; Ibrahim, R.; Kahtan, H.; Noor, N. New Texture Descriptor Based on Modified Fractional Entropy for Digital Image Splicing Forgery Detection. *Entropy* **2019**, *21*, 371. [CrossRef]
9. Bunk, J.; Bappy, J.H.; Mohammed, T.M.; Nataraj, L.; Flenner, A.; Manjunath, B.S.; Peterson, L. Detection and Localization of Image Forgeries Using Resampling Features and Deep Learning. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017. [CrossRef]
10. Zhang, Y.; Goh, J.; Win, L.L.; Thing, V.L.L. Image Region Forgery Detection: A Deep Learning Approach. In *EBook: Volume 14: Proceedings of the Singapore Cyber-Security Conference (SG-CRC) 2016, Singapore, 14–15 January 2016*; IOS Press: Singapore, 2016. [CrossRef]
11. Zhou, J.; Ni, J.; Rao, Y. Block-Based Convolutional Neural Network for Image Forgery Detection. In *Digital Forensics and Watermarking*; Kraetzer, C., Shi, Y.-Q., Dittmann, J., Kim, H.J., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 65–76.
12. Kim, D.-H.; Lee, H.-Y. Image manipulation detection using convolutional neural network. *Int. J. Appl. Eng. Res.* **2017**, *12*, 11640–11646.
13. Khan, M.; Yousaf, A.; Abbas, A.; Khurshid, K. Deep Learning for Automated Forgery Detection in Hyperspectral Document Images. *J. Electron. Imaging* **2018**, *27*, 053001. [CrossRef]
14. Liu, Y.; Guan, Q.; Zhao, X. Copy-move forgery detection based on convolutional kernel network. *Multimed. Tools Appl.* **2017**, *77*, 18269–18293. [CrossRef]
15. Cozzolino, D.; Thies, J.; Rössler, A.; Riess, C.; Nießner, M.; Verdoliva, L. ForensicTransfer: Weakly-supervised Domain Adaptation for Forgery Detection. *arXiv* **2018**, arXiv:1812.02510.
16. Khalid, H.; Woo, S.S. OC-FakeDect: Classifying Deepfakes Using One-class Variational Autoencoder. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 2794–2803. [CrossRef]
17. Marra, F.; Gragnaniello, D.; Verdoliva, L.; Poggi, G. A Full-Image Full-Resolution End-to-End-Trainable CNN Framework for Image Forgery Detection. *IEEE Access* **2020**, *8*, 133488–133502. [CrossRef]
18. Meena, K.B.; Tyagi, V. Image Forgery Detection: Survey and Future Directions. In *Data, Engineering and Applications*; Shukla, R.K., Agrawal, J., Sharma, S., Singh Tomer, G., Eds.; Springer: Singapore, 2019; pp. 163–194. [CrossRef]
19. Walia, S.; Saluja, K. Digital image forgery detection: A systematic scrutiny. *Aust. J. Forensic Sci.* **2019**, *51*, 488–526. [CrossRef]
20. Li, L.; Bao, J.; Zhang, T.; Yang, H.; Chen, D.; Wen, F.; Guo, B. Face X-Ray for More General Face Forgery Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 5000–5009. [CrossRef]
21. Abbas, M.N.; Ansari, M.S.; Asghar, M.N.; Kanwal, N.; O'Neill, T.; Lee, B. Lightweight Deep Learning Model for Detection of Copy-Move Image Forgery with Post-Processed Attacks. In Proceedings of the 2021 IEEE 19th World Symposium on Applied Machine Intelligence and Informatics (SAMI), Herl'any, Slovakia, 21–23 January 2021; pp. 125–130. [CrossRef]
22. Saber, A.H.; Khan, M.; Mejbel, B. A Survey on Image Forgery Detection Using Different Forensic Approaches. *Adv. Sci. Technol. Eng. Syst. J.* **2020**, *5*, 361–370. [CrossRef]
23. Zhang, R.; Ni, J. A Dense U-Net with Cross-Layer Intersection for Detection and Localization of Image Forgery. In Proceedings of the ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 4–8 May 2020. [CrossRef]
24. Liu, Y.; Guan, Q.; Zhao, X.; Cao, Y. Image Forgery Localization based on Multi-Scale Convolutional Neural Networks. In Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security, Innsbruck, Austria, 20–22 June 2018. [CrossRef]
25. Liu, B.; Wu, R.; Bi, X.; Xiao, B.; Li, W.; Wang, G.; Gao, X. D-Unet: A Dual-encoder U-Net for Image Splicing Forgery Detection and Localization. *arXiv* **2020**, arXiv:2012.01821.
26. Marra, F.; Gragnaniello, D.; Cozzolino, D.; Verdoliva, L. Detection of GAN-Generated Fake Images over Social Networks. In Proceedings of the 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), Miami, FL, USA, 10–12 April 2018. [CrossRef]
27. Kadam, K.; Ahirrao, S.; Kotecha, K.; Sahu, S. Detection and Localization of Multiple Image Splicing Using MobileNet V1. *IEEE Access* **2021**, *9*, 162499–162519. [CrossRef]
28. Jaiswal, A.; Srivastava, R. Image Splicing Detection using Deep Residual Network. In Proceedings of the 2nd International Conference on Advanced Computing and Software Engineering (ICACSE), Sultanpur, India, 8–9 February 2019; pp. 99–102. [CrossRef]
29. Stehouwer, J.; Dang, H.; Liu, F.; Liu, X.; Jain, A.K. On the Detection of Digital Face Manipulation. In Proceedings of the 2020 IEEECVF Conference on Computer Vision and Pattern Recognition CVPR, Seattle, WA, USA, 13-19 June 2020; pp. 5780–5789.

30. Nguyen, H.H.; Fang, F.; Yamagishi, J.; Echizen, I. Multi-task Learning for Detecting and Segmenting Manipulated Facial Images and Videos. In Proceedings of the 2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS), Tampa, FL, USA, 23–26 September 2019; pp. 1–8.
31. Li, Y.; Lyu, S. Exposing DeepFake Videos by Detecting Face Warping Artifacts. *arXiv* **2018**, arXiv:1811.00656.
32. Gidaris, S.; Singh, P.; Komodakis, N. Unsupervised Representation Learning by Predicting Image Rotations. *arXiv* **2018**, arXiv:1803.07728.
33. Wang, L.; Li, D.; Zhu, Y.; Tian, L.; Shan, Y. Dual Super-Resolution Learning for Semantic Segmentation. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 3773–3782. [CrossRef]
34. Yu, L.; Zhang, J.; Wu, Q. Dual Attention on Pyramid Feature Maps for Image Captioning. *IEEE Trans. Multimed.* **2022**, *24*, 1775–1786. [CrossRef]
35. Singh, B.; Sharma, D.K. Image Forgery over Social Media Platforms–A Deep Learning Approach for its Detection and Localization. In Proceedings of the 2021 8th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India, 17–19 March 2021; pp. 705–709.
36. Wang, X.Y.; Li, S.; Liu, Y.; Niu, Y.; Yang, H.; Zhou, Z. A new keypoint-based copy-move forgery detection for small smooth regions. *Multimed. Tools Appl.* **2016**, *76*, 23353–23382. [CrossRef]
37. Mahmoud, K.; Husien, A. Moment Based Copy Move Forgery Detection Methods. *Int. J. Comput. Sci. Inf. Secur.* **2016**, *14*, 28–35.
38. Wang, X.-Y.; Liu, Y.-N.; Xu, H.; Wang, P.; Yang, H.-Y. Robust Copy—Move Forgery Detection Using Quaternion Exponent Moments. *Pattern Anal. Appl.* **2018**, *21*, 451–467. [CrossRef]
39. Kuznetsov, A.; Myasnikov, V. A new copy-move forgery detection algorithm using image preprocessing procedure. In Proceedings of the 3rd International Conference "Information Technol. Nanotechnol. ITNT-2017, Samara, Russia, 25–27 April 2017; Volume 201, pp. 436–444. [CrossRef]
40. Niu, P.; Wang, C.; Chen, W.; Yang, H.; Wang, X. Fast and Effective Keypoint-Based Image Copy-Move Forgery Detection using Complex-Valued Moment Invariants. *J. Vis. Commun. Image Represent.* **2021**, *77*, 103068. [CrossRef]
41. Huang, H.-Y.; Ciou, A.-J. Copy-move forgery detection for image forensics using the superpixel segmentation and the Helmert transformation. *EURASIP J. Image Video Process.* **2019**, *2019*, 68. [CrossRef]
42. Dixit, A.; Bag, S. A fast technique to detect copy-move image forgery with reflection and non-affine transformation attacks. *Expert Syst. Appl.* **2021**, *182*, 115282. [CrossRef]
43. Yang, J.; Liang, Z.; Gan, Y.; Zhong, J. A novel copy-move forgery detection algorithm via two-stage filtering. *Digit. Signal Process.* **2021**, *113*, 103032. [CrossRef]
44. Dong, J.; Wang, W.; Tan, T. CASIA Image Tampering Detection Evaluation Database. In Proceedings of the 2013 IEEE China Summit and International Conference on Signal and Information Processing, Beijing, China, 6–10 July 2013; pp. 422–426.
45. Lei, Z.; Pietikainen, M.; Li, S.Z. Learning discriminative face descriptor. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 289–302.
46. Huang, J.; Li, Z.; Wang, J.; Wu, Y. *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*; Technical Report; University of Massachusetts: Amherst, MA, USA, 2007.
47. Abd El-latif, E.; Taha, A.; Zayed, H. A Passive Approach for Detecting Image Splicing using Deep Learning and Haar Wavelet Transform. *Int. J. Comput. Netw. Inf. Secur.* **2019**, *11*, 28–35. [CrossRef]
48. Bi, X.; Wei, Y.; Xiao, B.; Li, W. RRU-Net: The Ringed Residual U-Net for Image Splicing Forgery Detection. In Proceedings of the 2019 IEEECVF Conference on Computer Vision and Pattern Recognition. Workshop CVPRW, Long Beach, CA, USA, June 16–17 June 2019; pp. 30–39.
49. Doegar, A.; Dutta, M.; Gaurav, K. CNN Based Image Forgery Detection Using Pre-trained AlexNet Model. *Int. J. Comput. Intell. IoT* **2019**, *2*, 1.
50. 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, 8–14 September 2018.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.