Hybrid-Optimized Model for Deepfake Detection

H. Mancy[1], Marwa Elpeltagy[2], Kamal Eldahshan[3], Aya Ismail[4]
Department of Computer Science-College of Engineering and Computer Sciences, Prince Sattam Bin Abdulaziz University, Alkharj 11942 Saudi Arabia[1]
Department of Mathematics-Faculty of Science (Girls) Al-Azhar University, Cairo, Egypt[1]
Systems and Computers Department, Al-Azhar University, Egypt[2]
Mathematics Department, Faculty of Science Al-Azhar University, Egypt[3]
Mathematics Department, Tanta University, Egypt[4]

***Abstract*—The advancement of deep learning models has led to the creation of novel techniques for image and video synthesis. One such technique is the deepfake, which swaps faces among persons and then produces hyper-realistic videos of individuals saying or doing things that they never said or done. These deepfake videos pose a serious risk to everyone and countries if they are exploited for extortion, scamming, political disinformation, or identity theft. This work presents a new methodology based on a hybrid-optimized model for detecting deepfake videos. A Mask Region-based Convolutional Neural Network (Mask R-CNN) is employed to detect human faces from video frames. Then, the optimal bounding box representing the face region per frame is selected, which could help to discover many artifacts. An improved Xception-Network is proposed to extract informative and deep hierarchical representations of the produced face frames. The Bayesian optimization (BO) algorithm is employed to search for the optimal hyperparameters' values in the extreme gradient boosting (XGBoost) classifier model to properly discriminate the deepfake videos from the genuine ones. The proposed method is trained and validated on two different datasets; CelebDF-FaceForencics++ (c23) and FakeAVCeleb, and tested also on various datasets; CelebDF, DeepfakeTIMIT, and FakeAVCeleb. The experimental study proves the superiority of the proposed method over the state-of-the-art methods. The proposed method yielded %97.88 accuracy and %97.65 AUROC on the trained CelebDF-FaceForencics++ (c23) and tested CelebDF datasets. Additionally, it achieved %98.44 accuracy and %98.44 AUROC on the trained CelebDF-FaceForencics++ (c23) and tested DeepfakeTIMIT datasets. Moreover, it yielded %99.50 accuracy and %99.21 AUROC on the FakeAVCeleb visual dataset.**

***Keywords*—Bayesian optimization; deepfake detection; deepfake videos; Mask R-CNN; Xception network; XGBoost**

I. Introduction

The recent developments of autoencoder [1] and generative adversarial networks (GANs) [2], [3] have raised the generation of realistic images and videos. Deepfake techniques can manipulate a human's identity, attributes, or expressions and produce high-quality forged still images and videos. FaceSwap and DeepFaceLab are now the two most often used public open-source software tools for creating deepfakes. They are supported by thousands of users who contribute to developing and enhancing the software and models. Although the technology is used for amusing purposes as in movies or smartphone apps, it also has an evil side when it is employed to create realistic porn videos, spread falsified news, or create fake evidence.

Deciding the video's authenticity can become a top priority when a video pertains to national security concerns. Rapid advancements in video creation methods enable low-budget opponents to utilize commercial machine learning (ML) tools to produce realistic phony content. Therefore, there is a need for a deepfake detection methodology that can keep up with the development of deepfake creation methods, and properly discriminate deepfake videos against genuine ones.

This research presents a new methodology for detecting deepfake videos. It attempts to explore artifacts and visual discrepancies within video frames and decides if a certain video is authentic or deepfake. The Mask R-CNN has achieved effective and accurate performance on several object detection and segmentation benchmarks; the Cityscapes dataset COCO challenges [4], [5], and the WiderFace dataset. It has been demonstrated to be more precise than popular detectors; single-shot multi-box detector (SSD) and You Only Look Once (YOLO) in COCO [6]. It produces fewer false positives compared to YOLO. Additionally, it is more accurate in identifying the object and also offers segmentation information [7]. Consequently, the Mask R-CNN is suggested to be utilized as a detector to extract human faces from frames. This is followed by selecting the optimal bounding box representing the face area for each frame attempting to find a variety of artifacts. The convolutional neural network (CNN) is known to learn and extract discriminative local features effectively. It has been proven to be efficient in recognizing synthesized images and videos. Thus, an improved Xception-Network is employed to generate a deep useful spatial representation of the detected face frames. It assists in discriminating between authentic and deepfake videos. A single-layer classifier built using CNN's activation function may not always be the ideal option for classification. Instead, the sophisticated XGBoost model can overcome the single classifier's shortcomings in feature classification and provide strong predictive performance [8], [9], [10]. The XGBoost is a tree-based boosting ensemble method. Its basic goal is to iteratively combine several weak classifiers into a stronger and more precise classifier [11]. Thus, XGboost is applied here on the extracted features from the improved Xception-Network to check the authenticity of videos. The majority of ML algorithms rely on a variety of hyperparameters. Selecting effective hyperparameters; hyperparameter optimization, is crucial in ML since these parameters have a significant impact on the model performance. Hence, the BO algorithm is utilized to time-efficiently search for good hyperparameters of the XGBoost model. This helps to prevent overfitting and improves the deepfake detection model performance. The contributions of this research can be summarized as follows:

- The Mask R-CNN is employed to detect human faces from video frames. The optimal bounding box to represent the facial area per frame is then chosen in an attempt to find more artifacts. This assists to enhance the effectiveness of determining videos' authenticity.

- A hybrid optimized model using an improved Xception-Network and XGBoost with the Bayesian optimization algorithm is presented. This extracts distinctive information from the detected human faces, prevents overfitting, provides more precise predictions, and helps to distinguish deepfake videos from authentic ones. This ensures the maximum performance of the detection method.

- A comparative analysis with state-of-the-art deepfake video detection methods is conducted using several evaluation measures; accuracy, recall, precision, F measure, specificity, sensitivity, and Area Under Receiver Operating Characteristic (AUROC) curve metric.

The rest of the paper is structured as follows: Section II presents the related works for deepfake video detection methods. The proposed method and materials to detect deepfake videos are introduced in Section III. The experiment results and analysis are presented in Section IV. Section V is dedicated to the conclusion and future work.

II. Related Works

Recently, with the development of the internet over the world, the transmission of misleading information has increased significantly. The online media are seen to be tampered with to deceive the public. The progress of advanced artificial intelligence models in manipulating digital information has made it impossible to differentiate authentic media from the falsified with the naked eye. The deepfake technique uses deep-learning algorithms to swap faces or objects in digital content and videos, which convincingly generates realistic fake media. This has prompted the development of methods to detect deepfake media [12], [13]. Deepfake detection methods can be grouped into four types; physiological/physical-based methods [14, 15, 16, 17, 18], signal-based methods [10, 19, 20, 21, 22, 23, 24, 25], data-driven methods [26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38], or methods that are based on combining both signal and data-driven methods [39, 40, 41]. The physiological/ physical based detection methods reveal deepfake videos depending on the observation that synthesized videos lack direct knowledge of humans' physiological characteristics or the physics laws of the surroundings. The signal-based detection methods explore anomalies at the signal level caused by a deepfake generation process. They typically treat videos as a series of frames and a synced audio signal if the audio clip is available. The data-driven methods employ labelled videos; authentic or fake, to train deep learning models that can distinguish deepfake videos from authentic ones [42].

Gazi et al. [14] proposed a DeepVision algorithm for detecting deepfake videos based on analyzing the changes in eye blinking patterns. Blinking patterns fluctuate according to four factors: human gender, cognitive behavior, age, and time; AM, or PM. These factors are extracted per video. The Fast HyperFace algorithm is used to detect faces from video frames and localize landmarks. The eye-aspect-ratio algorithm is employed as an eye tracker to measure the eye blinking count, period, and elapsed blink time. The method finds and compares pattern information that matches the corresponding four factors in the pre-configured database of natural movements with the output of measured blinking per video to decide whether the blink is genuine or artificial. The four extracted factors are employed as search criteria for DeepVision's database. Elhassan et al. [17] presented a method to detect deepfake videos based on utilizing mouth and teeth movements per video frames as biological signal features. The dlib library is employed with the face detection algorithm to detect faces. The mouth-aspect-ratio technique is used to crop the opened mouth region from the detected face frames. The openness of the mouth is detected by determining 12 points representing the upper and lower lips. A CNN with six layers is used to extract features from mouth frames and then detect the deepfakes. Genuine videos are characterized by subtle motion signals that are not precisely replicated by different generative models. Consequently, the work in [18] leverages motion magnification to concentrate on differences in facial sub-muscular motions between genuine and fake videos. It combines traditional and deep motion magnification techniques to distinguish between genuine and fake video; as well, it also identifies the source generator of fake video based on generative artifacts.

The work in [20] employed two approaches to differentiate camera images from Generative Adversarial Network (GAN) based images. The first one is an Intensity Noise Histograms (INH)-based method using the rg chromaticity space. The second one is measuring the frequency of over exposed and under-saturated pixels as features in each image. Then, these features were fed into a linear Support Vector Machine classifier to detect the fake imagery. Zhang et al. [21] introduced a fake imagery detection method based on Spectrum Domain Features instead of the raw RGB image pixels. They employed the 2D Discrete Fourier Transform method on each channel of the RGB image to get a frequency spectrum image per channel. Then, the logarithm of the spectrum is computed and normalized to be fed into the fake imagery classifier to detect the artifacts and classify whether the image is fake or not. The Resnet-34 model with ImageNet is employed for the detection task. In addition, they presented AutoGAN, a GAN simulator that synthesizes GAN artifacts in the images and helps to train the classifier without requiring fake images for training or needing access to a pre-trained GAN model for creating fake images. In [25], a new deepfake video detection method was presented. It leveraged temporal phase variations across video frames using Complex Steerable Pyramid (CSP) decomposition. The output is then passed to a trainable spatiotemporal filter to detect motion cues suitable to distinguish deepfakes. After that, the ResNet-18 is employed to extract informative features, and the multi-scale temporal convolutional network is employed to capture facial temporal dynamics.

The work in [28], introduced a 3D CNN-based deepfake detection method. It used the RetinaFace to detect the faces from videos. It extracted the motion features from the adjacent video frames using the 3D CNN. The 3D CNN was composed of 3D residual blocks. It proved their efficiency in capturing spatial and temporal information. Agnihotri [29] employed the dlib to align and resize the face images. Then, three pre-trained CNNs were utilized for feature extraction; InceptionV3, EfficientNetB4, and InceptionResNetV2. This was followed by the Long Short-Term Memory (LSTM) network to classify fake and genuine images. Javed et al. [37] proposed to combine eye movement analysis with two deep learning models, MesoNet4 and ResNet101, to detect subtle and complex manipulations in deepfake videos. In [38], two deep-learning models, InceptionV3 and InceptionResNetV2, with the multilayer perceptron classifier, were presented to discern the authentic content from the deepfake one.

The work in [39], proposed to exploit the environmental artifacts to detect the deepfake videos via using texture feature-based method; local binary patterns. In addition, it employed the high-resolution network-based method to automatically learn the significant multi-resolution features from video frames. The features produced from both branches were combined and then fed into the capsule network for the final decision. Ismail et al. [40], proposed to use the Histogram of Oriented Gradient (HOG)-based CNN method to target some specific artifacts; visible splicing boundaries, for detecting the deepfakes. This discovered the distinction between the spatial HOG feature of the real and deepfake video frames. Additionally, an ameliorated XceptionNet was applied to video frames to automatically capture the hierarchical feature representations. The output features of both directions were merged to be fed into GRUs sequence and fully connected layers to detect the inconsistencies and temporal incoherence among the video frames, and then distinguish the deepfake videos from the real ones. In [41], three layers were introduced. The first layer was the RGB features extraction, which was employed to determine the potential forgery signs within the spatial video frames. It applied XceptionNet with an attention module on the cropped face regions resulting from using the dlib library. The second layer was the GAN features extraction, which was employed to detect forgery fingerprints in the high-frequency domain that were left by the GAN process. The final layer was utilized for feature extraction from the inner and outer areas of the manipulated part within a video frame.

Most deepfake detection methods suffer from the overfitting issue in the training data and lack to generalize well across various datasets and manipulation approaches. The proposed method aims to overcome these drawbacks. It uses the Mask R-CNN as a face detector, which is followed by selecting the optimal bounding box representing the facial region that could help to find more artifacts. It also presents a hybrid-optimized model. This model employs an improved Xception-Network to extract distinguished information. Additionally, it employs the XGBoost with the Bayesian optimization algorithm. The XGBoost is an ensemble model that could overcome the limitations of a single classifier. The BO can find the optimal hyperparameters of XGBoost which helps to avoid overfitting, produce more accurate predictions, and effectively distinguish deepfake videos from genuine ones. The proposed method is evaluated on two different datasets created using various manipulation methods. It surpassed previous methods and attained generalization.

III. Methods and Materials

The suggested deepfake video detection method is composed of the following stages: data pre-processing, deep feature extraction, and optimization-based classification. The three stages are depicted in Fig. 1 and will be described in detail hereafter.

A. *Data Pre-Processing Stage*

In this stage, the videos are converted into a sequence of frames. Then, the faces are detected and cropped from the frames if these frames are not face-centered. This is because most faking methods concentrate on creating forgery faces. The Mask R-CNN is employed here for face detection. It is a general and flexible framework for object instance segmentation. It is an improved version of the Faster R-CNN [43]. The mask R-CNN is characterized by locating objects from images while also producing a top-notch segmentation mask per object. It predicts a bounding box, a class label, and a mask for each instance in an image [44].

The Mask R-CNN is used here as follows. *First*, various levels of feature maps are extracted from the video frames using the last convolution layer of the ResNet-50 CNN's fourth phase. *Next*, the Feature Pyramid Network (FPN) [45] is utilized to ameliorate the feature extraction process by combining various scale features of the frame. The FPN consists of two paths: bottom-up, and top-down. The bottom up path is the usual CNN that is used for extracting four feature map sets. As going ascendingly, the spatial resolution declines. The semantic value of layers rises with more high-level structures discovered. The top-down path is used to build high-resolution layers out of a semantically rich layer. The lateral connections are added between these reconstructed layers that have more semantic properties and the corresponding feature maps to assist the detector in precisely predicting the objects' location. They serve as a skip connection to simplify training. Thus, the FPN produces multi-scale feature maps that have better information, and enhances the detection model performance. After that, the Region Proposal Lightweight Network (RPLN) employs the mechanism of a sliding window to scan these produced feature maps to find Regions of Interest (RoI) that contain the target object; human face. The sliding window consists of anchors that represent its center points. For each anchor, the RPLN produces two outcomes; foreground or background class, and a refined bounding box that perfectly fits the object. The foreground class indicates the box contains the object. To avoid overlapping multiple bounding boxes and ignore the redundant ones, a non-maximum suppression algorithm that is based on the intersection-over-union metric (1) is adopted to retain the bounding box with the highest target confidence score.

$$intersection - over - union = \frac{area_{P\_RoI} \cap area_{G\_Bb}}{area_{P\_RoI} \cup area_{G\_Bb}} \quad (1)$$

where, $area_{P\_RoI}$ represents the predicted RoI area, $area_{G\_Bb}$ refers to the ground truth bounding box area, and ∩ and ∪ indicate the overlap and union area of the two regions, respectively [46]. Thus, the RoI is positive, if the intersection-over-union metric is larger than 0.5. The region represents a negative bounding box if this metric is smaller than 0.5. This means the negative region is a background because it does not include the target foreground object. Following this, the final proposal regions are fed into a deep classifier and a regressor, and generate two outcomes. The first outcome is a specific object class, and the second one is a more refined bounding box that encapsulates the object better.

In addition, the RoI alignment layer is employed to correctly align the extracted feature maps with the input and preserve spatial locations. It is proposed in [44] to address the misalignment issue that results from using RoI pooling layer during the operations of the two-integer quantification in the Faster R-CNN. In detection and classification tasks, the RoI pooling layer is usually used after the convolutional layers. It can generate fixed-length feature map from each region and can then be forwarded to the next layers. However, the RoI pooling has a drawback. After a number of convolutional layers, the regions' position and size might be floating-point numbers, and there is a need to split the regions into fixed-size features. The RoI pooling rounds down these floating numbers into the nearest integer values. There are two rounding-down operations. The candidate region's coordinates are quantified to an integer. The quantified RoI is then split into k × k bins, and each bin is quantified once more. This may cause the localization precision to be lost. It generates misalignments between the region and the final extracted feature map. These misalignments have negative effects on the problem of detecting objects. The RoI alignment layer is another manner to obtain a fixed-size feature map from each region, but it retains the floating-point numbers in the operation. It eliminates all quantifications and uses bilinear interpolation resample method to generate accurate values. Thus, in the RoI alignment, the candidate region boundary coordinates are not rounded to retain the floating numbers. As well as, each RoI is split into $k \times k$ bins, and each bin is also not rounded. The bilinear interpolation is utilized to compute n sampled fixed points in each bin, and then the average or maximum pooling operation is performed to obtain alignment results representing this bin [47], [48], [49], [50], [51].

The bilinear interpolation method consists of two steps [4]. In the first step, the linear interpolation is performed in the x-axis direction as follows, using Formula (2) and (3):

$$f(x, y_1) = \frac{x_2 - x}{x_2 - x_1} f(x_1, y_1) + \frac{x - x_1}{x_2 - x_1} f(x_2, y_1) \quad (2)$$

$$f(x, y_2) = \frac{x_2 - x}{x_2 - x_1} f(x_1, y_2) + \frac{x - x_1}{x_2 - x_1} f(x_2, y_2) \quad (3)$$

In the second step, the linear interpolation is performed on the y-direction as follows, using Formula (4):

$$f(x, y) = \frac{y_2 - y}{y_2 - y_1} f(x, y_1) + \frac{y - y_1}{y_2 - y_1} f(x, y_2) \quad (4)$$

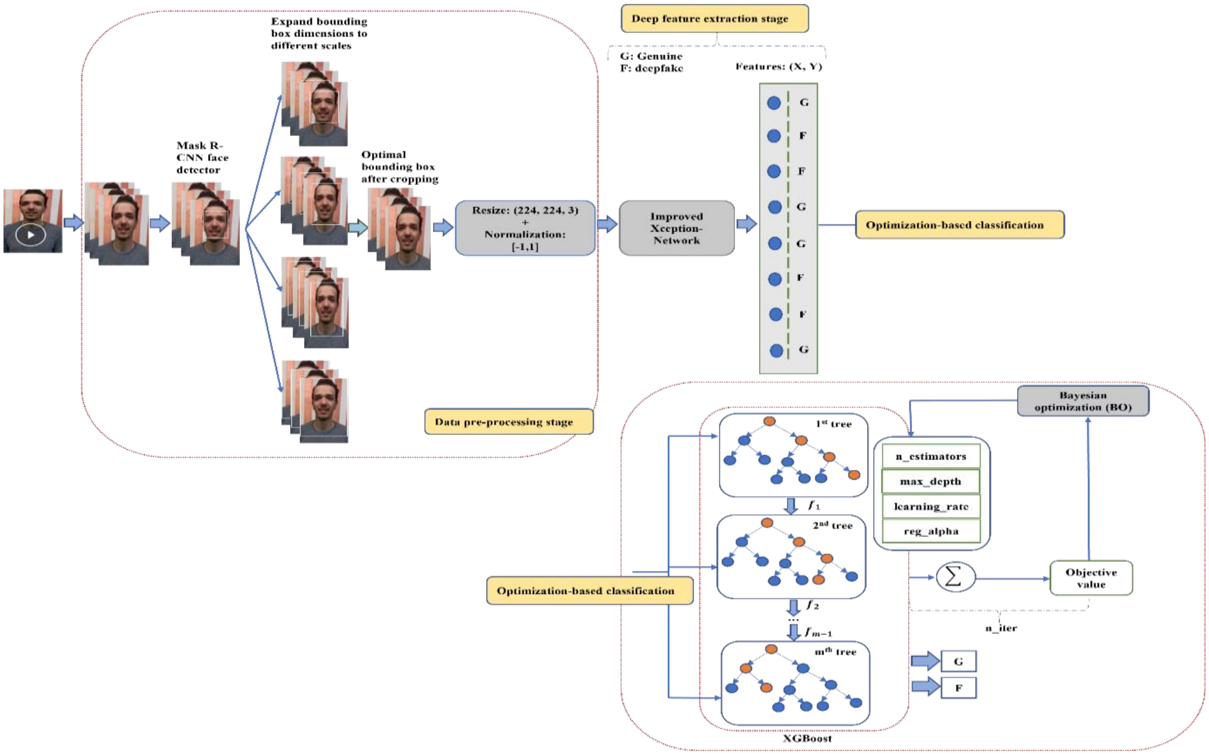

Expand bounding box dimensions to different scales
Mask R-CNN face detector
Optimal bounding box after cropping
Resize: (224, 224, 3) + Normalization: [-1,1]
Data pre-processing stage
Deep feature extraction stage
G: Genuine
F: deepfake
Features: (X, Y)
Improved Xception-Network
G
F
F
G
G
F
F
G
Optimization-based classification
Bayesian optimization (BO)
1st tree
f_1
2nd tree
f_2
...
f_{m-1}
mth tree
n_estimators
max_depth
learning_rate
reg_alpha
Optimization-based classification
∑
Objective value
n_iter
G
F
XGBoost

Fig. 1. The proposed deepfake video detection method architecture.

where, $f(x_1, y_1)$, $f(x_2, y_1)$, (x_1, y_2), and $f(x_2, y_2)$ indicate nearby grid points values, $f(x, y)$ represents the sampling point value, and $f(x, y_1)$ and $f(x, y_2)$ refer to the values produced from interpolating on the x-direction. Finally, the RoI alignment output is fed into fully connected layers for the operations of localization and classification. The architecture of the Mask R-CNN to detect faces from video frames is depicted in Fig. 2.

The resulting bounding box which localizes the face is very tight to the front. Thus, as shown in Fig. 1, the produced original bounding box's size is expanded by 7%, 14%, 21%, 28%, and 35% in proportion to its area to occupy a sizable portion of or all the head and neck that could potentially hold artifacts. These bounding boxes of different sizes are employed to crop and extract the face frames. This tries to find the optimal bounding box representing the face area, which helps to reveal as many artifacts as possible.

After the face frames are extracted from videos, they are resized to 224 × 224 × 3 and normalized in the range [-1, 1] to fit the next stage. The output of this will be the input to the coming stage to extract deep video features.

B. Deep Feature Extraction Stage

The detected videos' face frames that have the shape (224 × 224 × 3) are taken as input to the suggested improved Xception-Network where 224 refers to height and width values and 3 indicates the RGB channels for each frame. The architecture of the traditional Xception-Network depends on depth-wise separable convolutional layers. These layers not only allow for a considerable reduction in the parameters' number but also allow for the independent learning of spatial and channel correlation. It consists of three main phases. The first phase comprises one convolutional block and three separable convolutional blocks with skip connections. The second one consists of eight separable convolutional blocks that have also linear shortcut connections. The final phase comprises two separable convolutional blocks; one of them with residual connection around it and the other does not include it. These linear skip connections seek to stop the gradient from vanishing while the network is being trained [52]. The traditional Xception achieved good performance for facial image forgery detection [53].

The architecture of the suggested improved Xception-Network is shown in Fig. 3. Three convolutional blocks are added to the original architecture before the final rectified linear unit (relu) that followed the final separable convolutional block. These convolutional blocks include convolution layers with filters 1536, 1024, and 1536, respectively, and kernel of size (3, 3), batch normalization layers, and relu activation layers except for the last block. All the convolution layers' inputs are padded with a value of 0 to maintain the grid size. The convolution layers' filters provide feature maps with connections to the local area of the preceding layer. Thus, the convolution output is calculated by convolving the input (inp) with the filters as expressed in the following Formula (5) [54]:

$$x_i = inp * w_i + b_i, i = 1,2, \dots, n \tag{5}$$

where, n represents the number of convolution filters, and x_i indicates the feature map output corresponding to the i^{th} filter. The w_i denotes the learnable parameters of the i^{th} convolution filter, and b_i represents the i^{th} bias. The convolution layers provide effective spatial hierarchal representations of the input. The batch normalization normalizes the input via the entire batch by subtracting its mean and dividing by its standard deviation. Then, these normalized values $(\widehat{x_i})$ are scaled and shifted per channel using the following Formula (6) [54]:

$$y_i = \gamma \widehat{x_i} + \beta \tag{6}$$

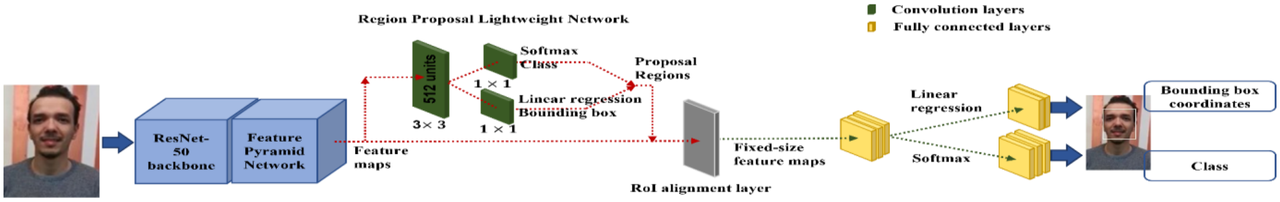


Fig. 2. The Mask R-CNN architecture for face detection.

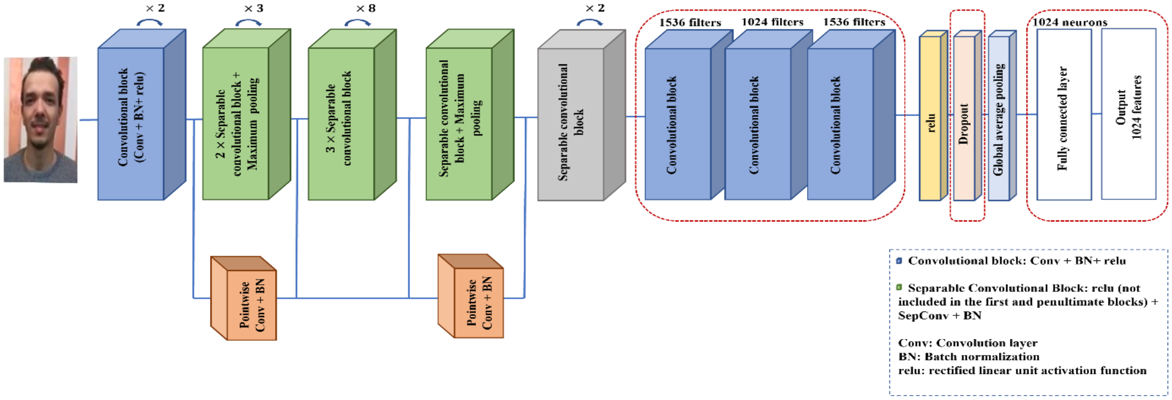

× 2
× 3
× 8
× 2
Convolutional block (Conv + BN+ relu)
2 × Separable convolutional block + Maximum pooling
3 × Separable convolutional block
Separable convolutional block + Maximum pooling
Separable convolutional block
1536 filters
1024 filters
1536 filters
Convolutional block
Convolutional block
Convolutional block
relu
Dropout
Global average pooling
1024 neurons
Fully connected layer
Output 1024 features
Pointwise Conv + BN
Pointwise Conv + BN
Convolutional block: Conv + BN+ relu
Separable Convolutional Block: relu (not included in the first and penultimate blocks) + SepConv + BN
Conv: Convolution layer
BN: Batch normalization
relu: rectified linear unit activation function

Fig. 3. The architecture of the suggested improved Xception-Network.

where, y_i represents the output value, and γ and β represent the scale and offset factors that can be learned during the training. The batch normalization speeds up the training, assists in minimizing the diminishing gradient problem, and improves the model generalization. The relu activation function (f) outputs a zero value for negative features to boost the network's nonlinear properties. It is defined as follows, using Formula (7):

$$f = \text{maximum}(0, y) \tag{7}$$

where, y denotes the relu's input value. It helps to speed up training and causes sparsity in the hidden units by squeezing values between zero and maximum [55]. Additionally, a dropout layer is added between the relu that followed the final separable convolutional block and the global average pooling layer to drop out randomly selected nodes with a probability of 20% per weight update. It has been adopted to diminish the effect of overfitting. After that, a fully connected layer with 1024 neurons and relu activation function is added. It is defined as follows, using Formula (8):

$$y_i \; = \; f\,(\, w_1 x_1 + \; \cdots + \; w_k x_k \,) \tag{8}$$

where, x_k denotes the k^{th} input to the fully connected layer, and y_i represents the i^{th} output from this layer. The $f\,(.)$ indicates the relu activation function, and w_* represents learnable weights in the network. The fully connected layer provides learning capabilities from all features' combinations of the preceding layer.

By applying the proposed improved Xception-Network on face frames, the output per frame is 1024 features constituting a vector representation. The proposed improvements on the Xception-network assists to produce a more valuable spatial hierarchical representation of face frames. This enhances the effectiveness of the deepfake detection method in real settings.

C. Optimization-Based Classification

After the improved Xception-network effectively extracted valuable spatial features per video, the XGBoost model optimized by the BO algorithm is adopted to distinguish the deepfake videos from the genuine ones. This contributes to overcoming the limitation of a single-layer classifier, preventing overfitting, and ameliorating the overall deepfake detection model's performance.

The XGBoost is employed for classification and regression tasks. It is based on the gradient-boosting approach. The input to the XGBoost can be expressed using Formula (9):

$$S = \{(x_i, y_i)\}, i = 1,2, \dots, n \tag{9}$$

where, x_i represents the i^{th} sample's features, y_i denotes the truth label, and n represents the samples' number.

The XGBoost model continuously adds a decision tree to learn a new function each time; $f(x)$, to fit the residual of the prior tree. After the model is trained, M trees are produced where the leaf node of each tree corresponds to a prediction score. The sample's final predicted value can be obtained by adding these scores corresponding to every tree. This can be defined as follows, using Formula (10):

$$\widehat{y_\iota} = \sum_{m=1}^{M} f_m(x_i), f_m \in F = \{f(x) = w\} \tag{10}$$

where, F represents the set space of all trees, and $\widehat{y_\iota}$ refers to the predicted value. The $f(x)$ refers to a single tree, and the w denotes the leaf nodes' weight score per tree. Since the XGBoost aims to learn these M trees, the following objective function should be minimized, using Formula (11):

$$F(y) = \sum_{i=1}^{n} l(y_i, \widehat{y_\iota}) + \sum_{m=1}^{M} \Omega\,(f_m) \tag{11}$$

where, $l(y_i, \widehat{y_\iota})$ represents the training loss function that measures the difference between the estimated and target scores. The $\Omega\,(f_m)$ denotes the penalty term which can help prevent overfitting, and it is expressed as follows, using Formula (12):

$$\Omega\,(f_m) = \gamma K + 0.5\lambda \sum_{j=1}^{K} w_j{}^2 \tag{12}$$

where, K represents the leaves' number, and γ refers to a hyper-parameter employed to control the model's complexity by controlling the leaves' number. The w denotes the leaves' weight score, and λ is used to make sure the leaves' score is not excessively high.

Since a new decision tree is iteratively added during the training, the XGBoost model at each iteration step (t) updates the objective function as follows, using Formula (13):

$$F(y)^{(t)} = \sum_{i=1}^{n} l(y_i, \widehat{y_\iota}^{(t-1)} + f_t(x_i)) + \Omega(f_t) \tag{13}$$

This objective function is minimized by applying the Taylor method. The first three terms of the Taylor expansion are taken by the XGBoost, and the extremely small high-order terms are ignored. Thus, the objective function is transformed into the following, using Formula (14):

$$F(y)^{(t)} \approx \sum_{i=1}^{n} [l(y_i, \widehat{y_\iota}^{(t-1)}) + g_i f_t(x_i) + 0.5 h_i f_t{}^2(x_i)] + \Omega(f_t) \tag{14}$$

where, g_i represents the first derivative of the objective loss function and h_i denotes its second derivative. These derivatives help to fit the residual error. Since the $l(y_i, \widehat{y_\iota}^{(t-1)})$ term has no impact on the objective function's optimization, it is eliminated. Thus, the objective function is rewritten as follows, using Formula (15):

$$\begin{aligned}\tilde{F}(y)^{(t)} &\approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + 0.5 h_i f_t{}^2(x_i)\right] + \gamma K + 0.5\lambda \sum_{j=1}^{K} w_j{}^2 \\ &= \sum_{j=1}^{K} [(\sum_{i \in I_j} g_i) w_j + 0.5(\sum_{i \in I_j} h_i + \lambda) w_j{}^2] + \gamma K \\ &\sum_{j=1}^{K} [G_j w_j + 0.5(H_j + \lambda) w_j{}^2] + \gamma K\end{aligned} \tag{15}$$

where, $I_j = \{i\}$ represents the data points indices set assigned to the j^{th} leaf node. The tree model iteration process can be considered as the leaf nodes iteration. The score of the optimal leaf node can be computed as follows, using Formula (16):

$$w_j = -\frac{G_j}{H_j + \lambda} \tag{16}$$

Finally, the objective loss function can be calculated as follows, using Formula (17):

$$\tilde{F}(y) = -0.5\sum_{j=1}^{K}\frac{{G_j}^2}{H_j+\lambda}+\gamma K \quad (17)$$

One of the most important concepts in machine learning is a parameter, and in the training, the model attempts to discover the appropriate parameters that help to obtain better performance. Hyperparameters are examples of such parameters. A hyperparameter controls the model's complexity or its learnability. Since having appropriate hyperparameters ameliorate the learning models' performance, optimizing them is significant.

Traditionally, hyperparameters optimization mainly depends on a trial-and-error manner and practical experience. Recently, optimization algorithms are employed to find satisfactory optimal hyperparameters. Random search, and grid search are popular examples of such optimization algorithms. The random search algorithm is slightly faster compared to the grid search algorithm, but it does not produce optimal results after optimizing the hyperparameters. The grid search optimization algorithm is very slow. On the other hand, Bayesian optimization [56], is a probabilistic-based optimization algorithm that globally seeks to maximize or minimize the objective function; $\max|\underset{x\epsilon H}{min} f(x)$ where, H represents the search space. It is flexible and powerful due to its probabilistic model [57, 58, 59, 60, 61, 62]. Therefore, the BO algorithm is employed here to search for the optimal hyperparameters' values of the XGBoost model. These values minimize the objective loss function of the XGBoost and improve the overall performance of the proposed method.

First, hyperparameter space; H, is defined by exploring the range of input values specified for each hyperparameter. The hyperparameter values could be continuous, categorical, or integers. The BO algorithm builds a probabilistic model of the objective function, utilizes this model to choose the next sample point to acquire, and updates the model based on this new sample point and its true objective function assessment [63]. It mainly consists of three steps: probabilistic model, acquisition function, and update process.

The probabilistic model; $p(f(x))$, can be defined as a distribution over the objective function for approximation. It gives an estimation of the objective function. Here, the probabilistic model is the Gaussian Process (GP) due to its analytic tractability and descriptive power [64, 65]. A GP is formally defined as a group of random variables where, each finite subset follows a multivariate normal distribution. Thus, the distribution over $f(x)$ in the GP is defined as follows, using Formula (18):

$$f(x) \sim N(\mu(x), \mathrm{c}(x) = k(x_n, x_m)) \quad (18)$$

where, the function $\mu(x)$ represents the mean and $c = k(x_n, x_m)$ represents the covariance. The k denotes the positive-definite kernel that specifies how points in the input space are correlated. Here, the Matern kernel [66, 67] is employed. The covariance function controls how observations affect the prediction.

The acquisition function is a metric that determines which hyperparameter value can cause the function to return the optimal value. It is employed to measure the evaluation effectiveness at any x . The acquisition function can be considered a guide to searching for the optimum. Its role is a trade-off between exploration and exploitation. The GP model's mean indicates the exploitation of the model's knowledge. The GP model's uncertainty indicates exploration due to the model doesn't have enough observations. Thus, the acquisition function uses the mean and the standard deviation of the function $f(x)$ at every x to calculate a value that represents how desirable it is to sample again at this location. The Upper Confidence Bound (UCB) is one such simple acquisition function that aims to weigh the importance between the mean and the uncertainty of the GP [68]. Its formula is defined as follows, using Formula (19) [69]:

$$\mathrm{UCB} = \mu(\mathrm{x}) + \beta\,\sigma(x) \quad (19)$$

where, β >0 is the learning rate hyperparameter that manages the preference between exploitation and exploration.

D. Dataset

The proposed method has been trained and validated on two datasets: CelebDF-FaceForencics++ (c23) [10],] 26], [40] and FakeAVCeleb [70], while it has been tested on CelebDF, DeepfakeTIMIT [71], and FakeAVCeleb. The CelebDF-FaceForencics++ (c23) dataset was created based on combining two popular datasets: the CelebDF and the FaceForencics++ (c23). 2848 genuine and deepfake visual videos of the CelebDF-FaceForencics++ (c23) are used to train and validate the proposed method. 518 genuine and fake visual videos of the CelebDF are used to test the proposed method. This mimics real-world situations due to CelebDF has high-quality visual deepfake videos that closely match those shared online. In addition, to confirm the robustness of the proposed method, 640 genuine and high-quality fake videos of the DeepfakeTIMIT dataset are also used to test the proposed method. Its fake videos are created using GAN-based face swapping techniques. Moreover, 1215 genuine and deepfake visual videos of the FakeAVCeleb are used to train, validate, and test the proposed method. Its genuine videos are varied in gender, age, and ethnic groups, and its fake videos are generated using different manipulation methods. This makes this dataset more realistic. All these datasets help to ameliorate the generalization of the proposed method in real scenarios.

IV. Experimental Results and Analysis

The proposed method to detect deepfake visual videos is trained and validated using CelebDF-FaceForencics++ (c23), and FakeAVCeleb datasets. It is tested using CelebDF, DeepfakeTIMIT, and FakeAVCeleb datasets. Evaluation metrics [72], [73]: accuracy, recall, precision, F-measure, specificity, sensitivity, and AUROC curve metric, are employed to assess the proposed method's performance. The following experiments are conducted:

Experiment 1: In this experiment, the proposed method is applied to the CelebDF-FaceForencics++ (c23) visual videos dataset. Since the frames of this dataset are not face-centered, the Mask R-CNN is used here for face detection. Different scales of bounding boxes representing faces are produced. Then, the proposed method's performance is evaluated per scale and the best result is recorded in Table II. It confirmed that expanding the original tight bounding box representing the face by 28% in proportion to its area to occupy a large portion of the head and neck helps to reveal more artifacts and improves performance.

The XGBoost model contains the following hyperparameters: n_estimators, max_depth, learning_rate, and reg_alpha. The n_estimators hyperparameter represents the number of the model's iterations which expresses the number of decision trees that will be generated. The max_depth represents the maximum depth of the decision tree. This constrains the maximum number of children that each tree's branch can have. The learning_rate represents the amount by which the weights are changed each time a tree is constructed. It manages the weighting of newly added trees to the model and prevents overfitting. The reg_alpha represents the L1 regularization term on weights. These hyperparameters constitute the search space that is used by Bayesian optimization to search for the optimal hyperparameters' values of the XGBoost. The range value adopted for each hyperparameter is shown in Table I.

TABLE I RANGE VALUES FOR THE XGBOOST HYPERPARAMETERS DURING BAYESIAN OPTIMIZATION

HYPERPARAMETER	Range
n_estimators	(10, 300)
max_depth	(5, 35)
learning_rate	(0, 1.0)
reg_alpha	(0, 1)

The validation AUROC score is utilized here during the Bayesian optimizer as the objective to be maximized. The number of model iteration times; n_iter, is selected as 70. This number refers to the number of hyperparameter combinations that are drawn from the search space. The result of each iteration on the CelebDF-FaceForencics++ (c23) dataset is recorded in Table II. The optimal set of hyperparameters is obtained at the forty-sixth iteration. Its value is: 0.023807687602778738 for learning_rate, 6.919040104018402 for max_depth, 299.5191634881166 for n_estimators, and 0.9287421690707279 for reg_alpha.

Finally, the XGBoost model is trained with these obtained optimal hyperparameters' values on the deep extracted features of the CelebDF-FaceForencics++ (c23) to minimize the objective loss function. The proposed method performance is evaluated on the CelebDF and DeepfakeTIMIT testing sets, and recorded in Table III. It achieves %97.88 accuracy, %97.68 recall, %99.12 precision, %98.39 F-measure, %98.27 specificity, %97.68 sensitivity, and %97.65 AUROC on the CelebDF test set. It yields %98.44 accuracy, %98.12 recall, %98.74 precision, %98.43 F-measure, %98.75 specificity, %98.12 sensitivity, and %98.44 AUROC on the DeepfakeTIMIT test dataset.

Experiment 2: In this experiment, the proposed method is applied to the FakeAVCeleb visual videos dataset. Its frames are face-centred and cropped. The number of model iteration times; n_iter, is selected as 10. The result of each iteration on the FakeAVCeleb dataset is recorded in Table IV. The optimal set of hyperparameters is obtained at the fourth iteration. Its value is: 0.25030643979197587 for learning_rate, 5.004759697203255 for max_depth, 151.11064359914357 for n_estimators, and 0.12147201179549794 for reg_alpha. The XGBoost model is then trained with these final optimal hyperparameters' values on the extracted features of the FakeAVCeleb visual videos dataset. The proposed method performance is evaluated on the FakeAVCeleb test set and recorded in Table V. It yields %99.50 accuracy, %100 recall, %98.97 precision, %99.48 F-measure, %99.06 specificity, %100 sensitivity, and %99.21 AUROC.

The confusion matrix visualization of the proposed method on CelebDF-FaceForencics++ (c23) training set with CelebDF and DeepfakeTIMIT testing sets, and FakeAVCeleb visual videos datasets is shown in Fig. 4. The ROC curve and the AUROC curve metric of the proposed method on CelebDF-FaceForencics++ (c23) training set with CelebDF and DeepfakeTIMIT testing sets, and FakeAVCeleb datasets are seen in Fig. 5. The ROC curve is very close to the upper left corner confirming the maximum performance of the proposed method. In addition, the high value of the AUROC curve metric also indicates better model performance.

TABLE II THE AUROC VALIDATION SCORE FOR EACH HYPERPARAMETER COMBINATION ON THE CELEBDF-FACEFORENCICS++ (C23) DATASET

ITERATION	learning_rate	max_depth	n_estimators	reg_alpha	AUROC score
1	0.4085	24.05	160.5	0.3467	0.9708
2	0.2247	21.98	220.9	0.6408	0.9735
3	0.9311	29.74	23.34	0.8362	0.9687
4	0.1335	34.4	11.02	0.9807	0.9587
5	0.09204	34.81	227.9	0.5098	0.9706
6	0.5368	5.32	11.9	0.9079	0.9675
7	0.9837	33.53	240.2	0.09379	0.9675
8	0.1178	5.799	83.88	0.9184	0.9712
9	0.1696	5.077	182.3	0.8891	0.9714
10	0.01696	34.84	85.08	0.8856	0.9716
11	0.1569	33.13	190.3	0.9958	0.9716
12	0.03718	5.464	248.7	0.9449	0.9702
13	0.01949	15.39	49.39	0.1021	0.9593
14	0.6047	11.07	106.3	0.1091	0.9693
15	0.1576	5.25	297.3	0.1108	0.9696

16	0.2196	5.658	164.4	0.9158	0.9741
17	0.01259	5.12	233.7	0.6258	0.9768
18	0.599	8.632	231.4	0.4695	0.9696
19	0.03539	5.213	84.0	0.7726	0.9744
20	0.2311	5.181	238.0	0.4045	0.9721
21	0.001677	34.75	57.33	0.1706	0.9423
22	0.07261	34.78	144.2	0.4977	0.9711
23	0.05849	32.9	279.4	0.9094	0.9752
24	0.000612	5.583	209.8	0.8951	0.942
25	0.9515	34.84	248.8	0.6843	0.9669
26	0.9829	19.12	28.74	0.8571	0.9689
27	0.886	32.38	275.3	0.9044	0.9683
28	0.6127	5.323	184.3	0.2443	0.9695
29	0.9562	5.484	138.5	0.6459	0.9669
30	0.836	5.308	31.64	0.05296	0.9667
31	0.1133	15.74	294.1	0.9701	0.9708
32	0.9233	22.01	154.3	0.9929	0.9685
33	0.9996	34.8	124.9	0.8693	0.9685
34	0.9612	34.91	157.2	0.92	0.9688
35	0.08564	21.82	10.02	0.01642	0.9477
36	0.6485	34.81	29.24	0.06077	0.9692
37	0.9735	21.55	79.4	0.7765	0.9672
38	0.8287	6.332	286.3	0.8931	0.9698
39	0.1454	8.875	235.9	0.2883	0.9713
40	0.2336	21.64	242.7	0.9865	0.9739
41	0.8836	5.425	70.31	0.0631	0.9672
42	0.02863	34.14	86.71	0.9099	0.9731
43	0.3259	12.69	191.3	0.9835	0.9722
44	0.009779	20.77	92.33	0.8829	0.9637
45	0.04701	20.91	271.4	0.9828	0.9708
46	**0.02381**	**6.919**	**299.5**	**0.9287**	**0.9791**
47	0.01089	5.964	24.25	0.9974	0.9392
48	0.939	16.1	46.05	0.03474	0.9671
49	0.9566	6.149	92.66	0.9749	0.9687
50	0.8834	5.04	43.08	0.785	0.9684
51	0.9345	33.79	19.72	0.1153	0.9671
52	0.5599	34.34	209.2	0.9586	0.9707
53	0.04613	5.274	150.2	0.9913	0.9767

TABLE III The Performance of the Proposed Method when Trained on the CelebDF-FaceForencics++ (c23) Set and Evaluated on the CelebDF Test Set and DeepfakeTIMIT Testing Sets

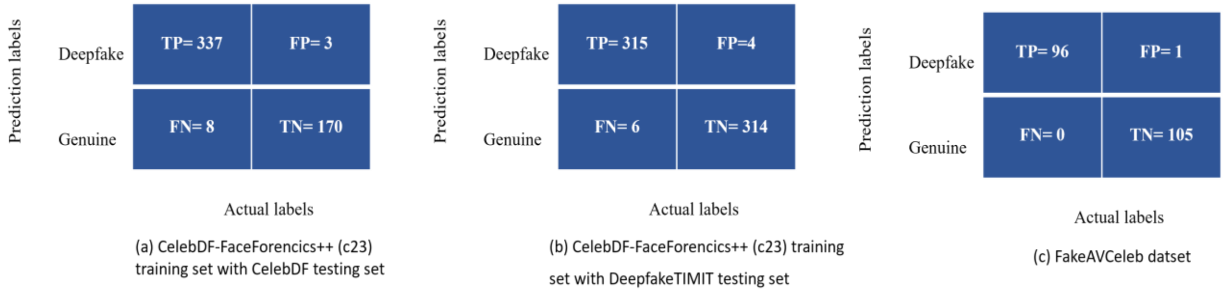
Dataset	Accuracy	Recall	Precision	F-measure	Specificity	Sensitivity	AUROC
CelebDF	%97.88	%97.68	%99.12	%98.39	%98.27	%97.68	%97.65
DeepfakeTIMIT	%98.44	%98.12	%98.74	%98.43	%98.75	%98.12	%98.44

TABLE IV The AUROC Validation Score for each Hyperparameter Combination on the FakeAVCeleb Visual Videos Dataset

Iteration	**learning_rate**	**max_depth**	**n_estimators**	**reg_alpha**	**AUROC score**
1	0.3751	24.21	285.5	0.07568	0.9874
2	0.7769	29.98	25.89	0.8177	0.9892
3	0.8854	26.67	10.74	0.9812	0.9835
4	**0.2503**	**5.005**	**151.1**	**0.1215**	**0.9925**
5	0.5979	34.99	181.7	0.749	0.988
6	0.7081	8.501	154.2	0.7557	0.9884
7	0.7139	5.133	146.5	0.08625	0.9823
8	0.3834	5.028	244.0	0.1609	0.9877
9	0.1924	5.96	299.9	0.5826	0.986
10	0.1453	5.599	34.12	0.07629	0.9871
11	0.1671	5.015	195.4	0.7694	0.9868
12	0.663	25.8	23.12	0.03216	0.989
13	0.1033	34.5	88.01	0.418	0.991

TABLE V THE PERFORMANCE OF THE PROPOSED METHOD WHEN TRAINED ON THE FAKEAVCELEB VISUAL VIDEOS DATASET

ACCURACY	Recall	Precision	F-measure	Specificity	Sensitivity	AUROC
%99.50	%100	%98.97	%99.48	%99.06	%100	%99.21



Deepfake	TP= 337	FP= 3
Genuine	FN= 8	TN= 170

(a) CelebDF-FaceForencics++ (c23) training set with CelebDF testing set

Deepfake	TP= 315	FP=4
Genuine	FN= 6	TN= 314

(b) CelebDF-FaceForencics++ (c23) training set with DeepfakeTIMIT testing set

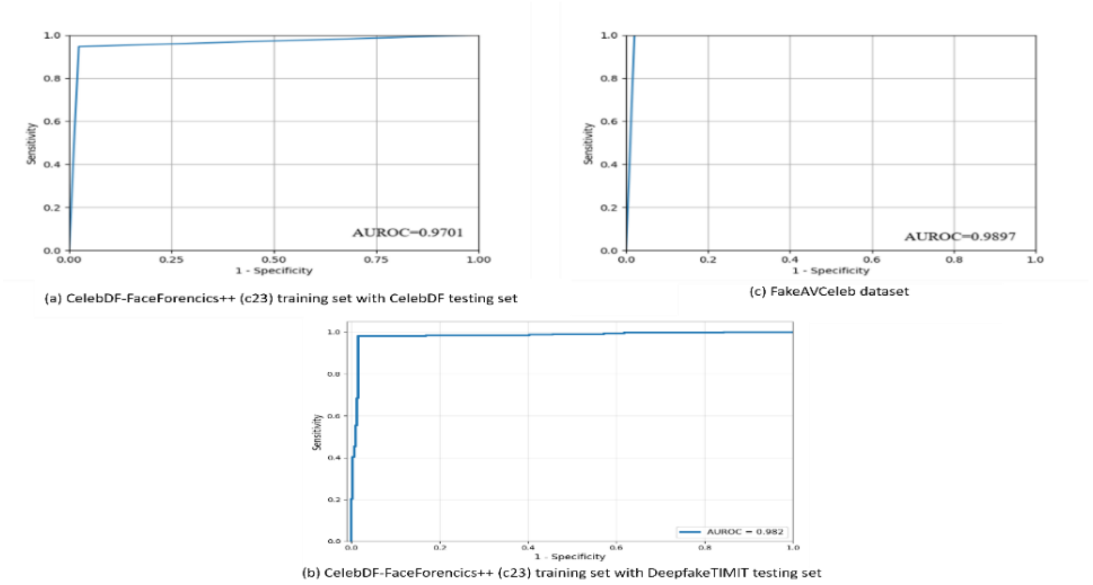
Deepfake	TP= 96	FP= 1
Genuine	FN= 0	TN= 105

(c) FakeAVCeleb datset

Fig. 4. The confusion matrix visualization of the proposed method on CelebDF-FaceForencics++ (c23) training set with CelebDF and DeepfakeTIMIT testing sets, and FakeAVCeleb visual videos dataset.

Fig. 6 compares the proposed deepfake video detection method with current state-of-the-art methods [10], [26], [53], [70], [74] using evaluation metrics on CelebDF-FaceForencics++ (c23) and FakeAVCeleb visual video datasets. As can be seen in Fig. 6, the proposed method has achieved higher performance as compared to the current methods. The experiments are performed on an OMEN HP laptop running Windows 11, an Intel (R) Core (TM) i7-9750H processor, and a 6-gigabyte RTX 2060 GPU. Python programming language is used to implement the proposed method. The implementation makes use of Python modules including keras, sklearn, openCV, matplotlib, os, random, tensorflow, numpy, xgboost and bayes_opt.

It can be concluded that employing the Mask R-CNN and selecting the optimal bounding box for face detection from video frames helped to reveal more artifacts. This improved the overall performance of the proposed deepfake video detection method. Additionally, a meaningful spatial representation of the detected faces was produced using the proposed improved version of the Xception-Network. This played an important role in differentiating between genuine and deepfake videos. Furthermore, using XGBoost with the BO algorithm on top of extracted representation produced optimal hyperparameters that prevent overfitting and improved the deepfake detection method performance by producing more precise predictions.

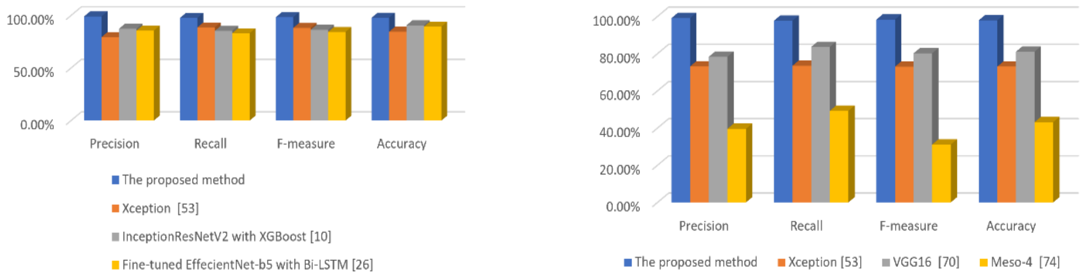


(a) CelebDF-FaceForencics++ (c23) training set with CelebDF testing set

(c) FakeAVCeleb dataset

(b) CelebDF-FaceForencics++ (c23) training set with DeepfakeTIMIT testing set

Fig. 5. The ROC curve and the AUROC curve metric of the proposed method on CelebDF-FaceForencics++ (c23) training set with CelebDF and DeepfakeTIMIT testing sets, and FakeAVCeleb visual videos dataset.



(a) CelebDF-FaceForencies++ (c23) training dataset, and CelebDF testing set

(b) FakeAVCeleb

Fig. 6. The evaluation metrics of the proposed method for deepfake video detection compared to current state-of-the-art methods.

V. Conclusion and Future Work

A new methodology for detecting deepfake videos has been introduced. It seeks to discover artifacts and visual discrepancies from video and then determine its authenticity. The Mask R-CNN is utilized to detect human faces from video frames. The optimal bounding box representing the facial area per frame is then chosen to find more artifacts which assists in ameliorating the method performance. An improved version of the Xception-Network is employed to produce an instructive spatial representation of face frames. It helps to distinguish between genuine and fake videos. The XGBoost with the Bayesian Optimization (BO) algorithm is applied to the extracted representation to decide video authenticity. The BO algorithm produced optimal hyperparameters of the XGBoost which assists in preventing overfitting. This provides more accurate predictions and ameliorates the overall performance of the proposed deepfake video detection method. CelebDF-FaceForencics++ (c23) and FakeAVCeleb visual videos datasets have been employed to train and validate the proposed method. CelebDF, DeepfakeTIMIT, and FakeAVCeleb datasets have been employed to test the proposed method. The proposed method achieved %97.88 accuracy, %97.68 recall, %99.12 precision, %98.27 F-measure, %98.27 specificity, %97.68 sensitivity, and %97.65 AUROC on the trained CelebDF-FaceForencics++ (c23) and tested CelebDF datasets. Additionally, it yielded %98.44 accuracy, %98.12 recall, %98.74 precision, %98.43 F-measure, %98.75 specificity, %98.12 sensitivity, and %98.44 AUROC on the trained CelebDF-FaceForencics++ (c23) and tested DeepfakeTIMIT datasets. Moreover, it yielded %99.50 accuracy, %100 recall, %98.97 precision, %99.48 F-measure, %99.06 specificity, %100 sensitivity, and %99.21 AUROC on the FakeAVCeleb visual dataset. As a result, the proposed method effectively outperformed the current state-of-the-art methods.

As the volume of fake content is continuously growing, there is a need to keep up by ameliorating the current deepfake detection methods to be able to detect the fakes produced by various manipulation methods. This could be accomplished using various augmentation techniques, other optimization algorithms, and more developed architectures. Additionally, there is a need to create a huge video dataset that resembles those circulating Online in an attempt to improve the generalization ability of the detection methods.

Conflict of Interest

The authors declare no conflict of interest.

Data Availability

The FakeAVCeleb dataset is available from the FakeAVCeleb site:

https://github.com/DASH-Lab/FakeAVCeleb.

The FaceForencics++ dataset is available from the FaceForensics site:

https://github.com/ondyari/FaceForensics.

The Celeb-DF dataset is available from the celeb-deepfakeforensics site:

https://github.com/yuezunli/celeb-deepfakeforensics.

Acknowledgment

This study is supported via funding from the Prince Sattambin Abdulaziz University (project number PSAU/2025/R/1446).

References

[1] Kingma DP, Welling M (2013) Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114.

[2] Karras T, Laine S, Aila T (2019) A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 4401-4410.

[3] Zhang H, Goodfellow I, Metaxas D, Odena A (2019, May) Self-attention generative adversarial networks. In International conference on machine learning. PMLR, pp 7354-7363.

[4] Lin K, Zhao H, Lv J, Li C, Liu X, Chen R, Zhao R (2020) Face detection and segmentation based on improved mask R-CNN. Discrete dynamics in nature and society.

[5] Bakr H, Hamad A, Amin K (2021) Mask R-CNN for Moving Shadow Detection and Segmentation. IJCI. International Journal of Computers and Information 8(1): 1-18.

[6] Chitturi G (2020) Building Detection in Deformed Satellite Images Using Mask R-CNN.

[7] Buric M, Pobar M, Ivasic-Kos M (2018, December) Ball detection using YOLO and Mask R-CNN. In 2018 International Conference on Computational Science and Computational Intelligence (CSCI). IEEE, pp 319-323.

[8] Ren X, Guo H, Li S, Wang S, Li J (2017, August) A novel image classification method with CNN-XGBoost model. In International Workshop on Digital Watermarking. Springer, Cham, pp 378-390.

[9] Li B, Ai D, Liu X (2022) CNN-XG: A Hybrid Framework for sgRNA On-Target Prediction. Biomolecules 12(3): 409.

[10] Ismail A, Elpeltagy M, Zaki MS, Eldahshan K (2021) A New Deep Learning-Based Methodology for Video Deepfake Detection Using XGBoost. Sensors 21(16): 5413.

[11] Wei A, Yu K, Dai F, Gu F, Zhang W, Liu Y (2022). Application of Tree-Based Ensemble Models to Landslide Susceptibility Mapping: A Comparative Study. Sustainability, 14(10): 6330.

[12] Vamsi VVVNS, Shet SS, Reddy SSM, Rose SS, Shetty SR, Sathvika S, Supriya MS, Shankar SP (2022) Deepfake Detection in Digital Media Forensics. Global Transitions Proceedings.

[13] Taeb M, Chi H (2022) Comparison of Deepfake Detection Techniques through Deep Learning. Journal of Cybersecurity and Privacy 2(1): 89-106.

[14] Jung T, Kim S, Kim K (2020) Deepvision: Deepfakes detection using human eye blinking pattern. IEEE Access 8: 83144-83154.

[15] Yang X, Li Y, Lyu S (2019, May) Exposing deep fakes using inconsistent head poses. In ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp 8261-8265.

[16] Lutz K, Bassett R (2021) DeepFake Detection with Inconsistent Head Poses: Reproducibility and Analysis. arXiv preprint arXiv:2108.12715.

[17] Elhassan A, Al-Fawa'reh M, Jafar MT, Ababneh M, Jafar ST (2022) DFT-MF: Enhanced deepfake detection using mouth movement and transfer learning. SoftwareX, 19, 101115.

[18] Demir, I., & Çiftçi, U. A. (2024). How Do Deepfakes Move? Motion Magnification for Deepfake Source Detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision* (pp. 4780-4790).

[19] Matern F, Riess C, Stamminger M (2019, January) Exploiting visual artifacts to expose deepfakes and face manipulations. In 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW). IEEE pp 83-92.

[20] McCloskey S, Albright M (2019, September). Detecting GAN-generated imagery using saturation cues. In 2019 IEEE international conference on image processing (ICIP). IEEE, pp 4584-4588.

[21] Zhang X, Karaman S, Chang SF (2019, December) Detecting and simulating artifacts in gan fake images. In 2019 IEEE international workshop on information forensics and security (WIFS). IEEE, pp. 1-6.

[22] Nirkin Y, Wolf L, Keller Y, Hassner T (2021) DeepFake detection based on discrepancies between faces and their context. IEEE Transactions on Pattern Analysis and Machine Intelligence.

[23] Habeeba S, Lijiya A, Chacko AM (2021) Detection of deepfakes using visual artifacts and neural network classifier. In Innovations in Electrical and Electronic Engineering. Springer, Singapore, pp 411-422.

[24] Luo Y, Zhang Y, Yan J, Liu W (2021) Generalizing face forgery detection with high-frequency features. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 16317-16326.

[25] Prashnani E, Goebel M, Manjunath BS (2024) Generalizable deepfake detection with phase-based motion analysis. *IEEE Transactions on Image Processing*.

[26] Ismail A, Elpeltagy M, Zaki MS, ElDahshan KA (2021) Deepfake video detection: YOLO-Face convolution recurrent approach. PeerJ Computer Science 7: e730.

[27] Jeong Y, Kim D, Ro Y, Choi J (2022) FrePGAN: Robust Deepfake Detection Using Frequency-level Perturbations. arXiv preprint arXiv:2202.03347.

[28] de Lima O, Franklin S, Basu S, Karwoski B, George A (2020) Deepfake detection using spatiotemporal convolutional networks. arXiv preprint arXiv:2006.14749.

[29] Agnihotri A (2021). DeepFake Detection using Deep Neural Networks (Doctoral dissertation, Dublin, National College of Ireland).

[30] Gong D, Yogan JK, Goh OS, Ye Z, Chi W (2021) DeepfakeNet, an efficient deepfake detection method. International Journal of Advanced Computer Science and Applications 12(6).

[31] Deng L, Suo H, Li D (2022) Deepfake Video Detection Based on EfficientNet-V2 Network. Computational Intelligence and Neuroscience.

[32] Suganthi ST, Ayoobkhan MUA, Bacanin N, Venkatachalam K, Štěpán H, Pavel T (2022) Deep learning model for deep fake face recognition and detection. PeerJ Computer Science, 8: e881.

[33] Khan SA, Dang-Nguyen DT (2022) Hybrid Transformer Network for Deepfake Detection. arXiv preprint arXiv:2208.05820.

[34] Maiano L, Papa L, Vocaj K, Amerini I (2022) DepthFake: a depth-based strategy for detecting Deepfake videos. arXiv preprint arXiv:2208.11074.

[35] Elpeltagy M, Ismail A, Zaki MS, Eldahshan K (2023) A novel smart deepfake video detection system. International Journal of Advanced Computer Science and Applications (IJACSA) 14(1).

[36] Cunha L, Zhang L, Sowan B, Lim CP, Kong Y (2024) Video deepfake detection using Particle Swarm Optimization improved deep neural networks. Neural Computing and Applications 1-37.

[37] Javed M, Zhang Z, Dahri FH, Laghari AA (2024) Real-time deepfake video detection using eye movement analysis with a hybrid deep learning approach. *Electronics*. *13*(15): 2947.

[38] Sundaram V, Senthil B, Vekkot S (2024, June) Enhancing Deepfake Detection: Leveraging Deep Models for Video Authentication. In *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT).* IEEE, pp 1-7.

[39] Khalil SS, Youssef SM, Saleh SN (2021) iCaps-Dfake: An integrated capsule-based model for deepfake image and video detection. Future Internet 13(4): 93.

[40] Ismail A, Elpeltagy M, Zaki MS, Eldahshan K (2022) An integrated spatiotemporal-based methodology for deepfake detection. Neural Computing and Applications 1-15.

[41] Rathoure N, Pateriya RK, Bharot N, Verma P (2024) Combating deepfakes: a comprehensive multilayer deepfake video detection framework. *Multimedia Tools and Applications,* pp 1-18.

[42] Lyu S (2022) DeepFake Detection. In Multimedia Forensics. Springer, Singapore, pp 313-331.

[43] Girshick R (2015) Fast r-cnn. In Proceedings of the IEEE international conference on computer vision, pp 1440-1448.

[44] He K, Gkioxari G, Dollár P, Girshick R (2017) Mask r-cnn. In Proceedings of the IEEE international conference on computer vision, pp 2961-2969.

[45] Lin TY, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2117-2125.

[46] Xavier AI, Villavicencio C, Macrohon JJ, Jeng JH, Hsieh JG (2022) Object Detection via Gradient-Based Mask R-CNN Using Machine Learning Algorithms. Machines 10(5): 340.

[47] Cui Z, Lu N, Jing X, Shi X (2018, November) Fast dynamic convolutional neural networks for visual tracking. In Asian Conference on Machine Learning. PMLR pp 770-785.

[48] Gonzalez S, Arellano C, Tapia JE (2019) Deepblueberry: Quantification of blueberries in the wild using instance segmentation. IEEE Access 7: 105776-105788.

[49] Zhang X, Zhu K, Chen G, Tan X, Zhang L, Dai F, Liao P, Gong Y (2019) Geospatial object detection on high resolution remote sensing imagery based on double multi-scale feature pyramid network. Remote Sensing 11(7): 755.

[50] Chen QQ, Gan XX, Huang W, Feng JJ, Shim H (2020) Road damage detection and classification using mask R-CNN with DenseNet backbone. CMC-Computers Materials & Continua 65(3): 2201-2215.

[51] Yang Z, Dong R, Xu H, Gu J (2020) Instance segmentation method based on improved mask R-CNN for the stacked electronic components. Electronics 9(6): 886.

[52] Chollet F (2017) Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1251-1258.

[53] Rossler A, Cozzolino D, Verdoliva L, Riess C, Thies J, Nießner M (2019) Faceforensics++: Learning to detect manipulated facial images. In Proceedings of the IEEE/CVF international conference on computer vision, pp 1-11.

[54] Alheejawi S, Mandal M, Xu H, Lu C, Berendt R, Jha N (2020) Deep learning-based histopathological image analysis for automated detection and staging of melanoma. In Deep Learning Techniques for Biomedical and Health Informatics. Academic Press, pp 237-265.

[55] Nwankpa C, Ijomah W, Gachagan A, Marshall S (2018) Activation functions: Comparison of trends in practice and research for deep learning. arXiv preprint arXiv:1811.03378.

[56] Močkus J (1975) On Bayesian methods for seeking the extremum. In Optimization techniques IFIP technical conference. Springer, Berlin, Heidelberg, pp 400-404.

[57] Gardner JR, Kusner MJ, Xu ZE, Weinberger KQ, Cunningham JP (2014, June) Bayesian optimization with inequality constraints. In ICML Vol 2014, pp 937-945.

[58] Wang H, van Stein B, Emmerich M, Back T (2017, October) A new acquisition function for Bayesian optimization based on the moment-generating function. In 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE, pp 507-512.

[59] Klein A (2020) Efficient bayesian hyperparameter optimization. Doctoral dissertation, Dissertation, Universität Freiburg.

[60] Jiao W, Hao X, Qin C (2021) The Image Classification Method with CNN-XGBoost Model Based on Adaptive Particle Swarm Optimization. Information. 12(4): 156.

[61] Qin C, Zhang Y, Bao F, Zhang C, Liu P, Liu P (2021) XGBoost optimized by adaptive particle swarm optimization for credit scoring. Mathematical Problems in Engineering.

[62] Gao J, Ma C, Wu D, Xu X, Wang S, Yao J (2022) Recognition of Human Motion Intentions Based on Bayesian-Optimized XGBOOST Algorithm. Journal of Sensors.

[63] Muhammad A, Moustafa M (2018, December) Improving region-based CNN object detector using bayesian optimization. In 2018 IEEE International Conference on Image Processing, Applications and Systems (IPAS). IEEE, pp 32-36.

[64] Klein A, Falkner S, Bartels S, Hennig P, Hutter F (2017, April) Fast bayesian optimization of machine learning hyperparameters on large datasets. In Artificial intelligence and statistics. PMLR, pp 528-536.

[65] Masum M, Shahriar H, Haddad H, Faruk MJH, Valero M, Khan MA, Rahman MA, Adnan MI, Cuzzocrea A, Wu F (2021, December) Bayesian hyperparameter optimization for deep neural network-based network intrusion detection. In 2021 IEEE International Conference on Big Data (Big Data). IEEE, pp 5413-5419.

[66] Murphy KP (2012) Machine learning: a probabilistic perspective. MIT press.

[67] Shah A, Wilson A, Ghahramani Z (2014, April) Student-t processes as alternatives to Gaussian processes. In Artificial intelligence and statistics. PMLR, pp 877-885.

[68] Nandy A, Kumar C, Mewada D, Sharma S (2020) Bayesian Optimization--Multi-Armed Bandit Problem. arXiv preprint arXiv:2012.07885.

[69] Wang X, Jin Y, Schmitt S, Olhofer M (2022) Recent Advances in Bayesian Optimization. arXiv preprint arXiv:2206.03301.

[70] Khalid H, Tariq S, Kim M, Woo SS (2021) FakeAVCeleb: a novel audio-video multimodal deepfake dataset. arXiv preprint arXiv:2108.05080.

[71] Korshunov P, Marcel S (2018) Deepfakes: a new threat to face recognition? assessment and detection. *arXiv preprint arXiv:1812.08685*.

[72] Dalianis H (2018) Evaluation metrics and evaluation. In Clinical text mining. Springer, Cham, pp 45-53.

[73] Hossin M, Sulaiman MN (2015) A review on evaluation metrics for data classification evaluations. International journal of data mining & knowledge management process. 5(2): 1.

[74] Afchar D, Nozick V, Yamagishi J, Echizen I (2018, December) Mesonet: a compact facial video forgery detection network. In 2018 IEEE international workshop on information forensics and security (WIFS). IEEE. pp 1-7.