Neural Computing and Applications
https://doi.org/10.1007/s00521-022-07633-3

ORIGINAL ARTICLE

An integrated spatiotemporal-based methodology for deepfake detection

Aya Ismail[1] · Marwa Elpeltagy[2] · Mervat S. Zaki[3] · Kamal Eldahshan[4]

Received: 3 January 2022 / Accepted: 11 July 2022

Abstract
Rapid advances in deep learning models have made it easier for public and crackers to generate hyper-realistic deepfake videos in which faces are swapped. Such deepfake videos may constitute a significant threat to the world if they are misused to blackmail public figures and to deceive systems of face recognition. As a result, distinguishing these fake videos from real ones has become fundamental. This paper introduces a new deepfake video detection method. You Only Look Once (YOLO) face detector is used to detect faces from video frames. A proposed hybrid method based on proposing two different feature extraction methods is applied to these faces. The first feature extraction method, a proposed Convolution Neural Network (CNN), is based on the Histogram of Oriented Gradient (HOG) method. The second one is an ameliorated XceptionNet CNN. The two extracted sets of features are merged together and fed as input to a sequence of Gated Recurrent Units (GRUs) to extract the spatial and temporal features and then individuate the authenticity of videos. The proposed method is trained on the CelebDF-FaceForencics++ (c23) dataset and evaluated on the CelebDF test set. The experimental results and analysis confirm the superiority of the suggested method over the state-of-the-art methods.

Keywords Deepfake · YOLO · Face detector · CNN · HOG · XceptionNet · GRU · Deepfake video detection · Videos authenticity

1 Introduction

Currently, the evolution of digitally tampered media is attracting the attention of the public, policymakers, attackers, and researchers. The progress in artificial intelligence, especially deep networks, has enabled the creation of highly realistic deepfake videos. These videos portray the target subject saying or doing things said or done by the source one. Deepfakes are AI-synthesized and generated videos and audio. This kind of fake media has given rise to substantial concerns about potential misuse. In 2017, the term deepfake arises from the Reddit site when a user, called deepfake, posted tampered pornographic videos for target actors and shared the writing code to enable the public to follow suit. The deepfake technique is menacing to world security when its creation methods can be used to generate leaders' videos with counterfeit speeches for falsification goals. Therefore, it can be abused to extort victims, cause religious and political tensions between nations, deceive the public, affect the election results, and expose individuals, societies, governmental institutions, and countries to danger [1].

Most deepfake applications are based on two AI approaches: autoencoders and Generative Adversarial Networks (GANs). The autoencoders can extract the latent facial features from photographs and then use these

✉ Aya Ismail
aya.ismail@science.tanta.edu.eg

Marwa Elpeltagy
marwa.elpeltagy@ejust.edu.eg

Mervat S. Zaki
zakimervat0548@gmail.com

Kamal Eldahshan
dahshan@gmail.com

[1] Mathematics Department, Tanta University, Tanta, Egypt

[2] Systems and Computers Department, Al-Azhar University, Cairo, Egypt

[3] Mathematics Department, Al-Azhar University (Girls Branch), Cairo, Egypt

[4] Mathematics Department, Al-Azhar University, Cairo, Egypt

Published online: 02 August 2022

features to create photographs with a different expression. The GANs can learn how to model the input distribution by training two competing networks: generator and discriminator. The generator keeps on discovering how to generate fake data that can deceive the discriminator, while the discriminator is trained to differentiate between fake and genuine data. As the training goes forward, the discriminator will not be able to discern the difference between the generated fake data and the genuine one. Thus, the discriminator can be rejected, and the generator can then be utilized to generate hyper-realistic data that have not been seen before [2].

Malicious video deepfakes can be categorized into face-swap, puppet master, lip-sync, facial synthesis and attribute altering, and voice cloning deepfakes. Face-swap or replacement is when the face of the source subject is replaced with the target subject to create a deepfake video of the target one, attempting to depict the actions done by the source subject to the target one. Several implementations to create the face-swap technique have been deployed, including FaceSwap [3], DeepFaceLab [4], DFaker [5], FaceSwap GAN [6], and DeepFake-tf [7]. The deepfake puppet master or re-enactment is created by imitating either the expressions of the target subject, such as facial expressions, and head and eye movements, or the full body. Face2face [8], ReenactGAN [9], and Headon [10] are examples of the re-enactment deepfake technique. Lip-sync is when the lip movements of the target subject are transformed to be consistent with a certain audio recording as in synthesizing Barack Obama video [11]. Facial synthesis and attribute altering focus on the generation of realistic face photos and facial attribute manipulation. DCGAN [12], ProGAN [13], StyleGAN [14], BigGAN [15], IcGAN [16], StarGAN [17], and AttGAN [18] are some instances of the facial synthesis and attribute altering deepfake technique. The voice cloning [19, 20] or audio deepfake involves the generation of the voice of the target subject utilizing deep learning techniques to depict the target saying things they have never said.

Since the great advances in deepfake video creation methods, there is a necessity to keep up with the evolution of such creation methods thus arising a need for a deepfake video detection method that is generally applicable to videos generated by any deepfake method. This paper presents a new efficient method, YOLO-Feature extraction-merge hybrid method (YF), which captures the defects in the spatial and temporal domains of video frames and then distinguishes whether a given video is a deepfake or not. The YOLO face detector has been proven its efficiency in detecting the deepfake videos over the other state-of-the-art face detectors since it produces fewer false-positive instances [21–23], thus enhancing the performance of the proposed detection method. Therefore, it is used as a face detector extracting the faces from the video frames. Moreover, a new feature extraction-merge method is proposed based on a combination of two proposed feature extraction methods. The first method is a proposed CNN based on the HOG descriptor. The HOG descriptor is used to extract the spatial gradient orientation features that describe the local contour, silhouette, and some texture information of faces. The HOG has proven its effectiveness in several image processing and computer vision applications, especially for action recognition [24], object detection [25], facial expression and face recognition [26, 27], and videos and images forgery detection [28–30]. The output of the HOG is then fed as an input to a proposed CNN to learn the most discriminative spatial gradient orientation features by utilizing several building blocks. This aims to discover the discrepancies in spatial information between genuine and manipulated video frames. The second feature extraction method adopted here is the XceptionNet with some ameliorations. The XceptionNet had previously attained good results in detecting the forged videos [31, 32]. The proposed amelioration to the XceptionNet is aimed to produce an informative spatial representation of the video features and enhance the performance of the deepfake video detection method in real-world scenarios. Furthermore, the outputs of these two feature extraction methods are merged and fed to a sequence of GRUs to learn and extract the spatiotemporal features of videos. The GRU is a kind of Recurrent Neural Networks (RNNs). It shows a big efficiency in mitigating the gradient vanishing and exploding problem of the traditional RNNs when learning long-term dependencies. Its architecture is simpler while maintaining the effect of the Long Short-Term Memory (LSTM) [33].

In summary, this paper presents the following contributions:

- A new feature extraction-merge method based on a combination of two newly proposed different feature extraction methods; a proposed CNN based on HOG descriptor, and an ameliorated XceptionNet, is introduced to efficiently and powerfully learn the discriminant spatial video information. The final deepfake detection performance has been improved due to the combination of the two feature extraction methods.
- Many deepfake videos creation methods work on a frame-by-frame basis where faces of each frame are processed or swapped independently of the other frames. Thus, the generated videos lack coherence in temporal information. A sequence of GRUs is applied on the merged features to provide high learning capabilities to better understand of the incoherence in the temporal domain between genuine and deepfakes videos.

- A comparative study with current state-of-the-art methods of deepfake video detection is applied in terms of AUROC, accuracy, sensitivity, specificity, precision, recall, and F1-measure.

The rest of this paper is organized as follows: Sect. 2 presents a literature review about deepfake video detection methods. Section 3 introduces methods and materials for detecting deepfake videos. Section 4 introduces the time complexity of the proposed method. Section 5 is dedicated to the case study, results and analysis. Section 6 presents the conclusion and future work.

2 Literature review

The progress of machine learning algorithms has raised the ease of creating fake content: images, videos, and audio. Additionally, it has enhanced the realism of tampered information. It is extremely difficult for individuals to distinguish between genuine and fake multimedia. Since deepfake media violates privacy and poses a significant threat to the world, many researchers have paid considerable attention to create methods for detecting media manipulations and forgery. Current deepfakes detection approaches can be categorized into two categories. The first one consists of methods that depend on specific artifacts: spatial and temporal, generated using various deepfake creation methods. The second one comprises methods that discover differentiating properties via training on deepfakes datasets using deep neural networks, which is called data-driven classification. The spatial artifacts involve inconsistencies, GAN artificial fingerprints, and abnormalities in environment. Meanwhile, the temporal artifacts include incoherence, personal behavior variations, physiological signals changes, and un-synchronization among video frames [34, 35].

Li and Lyu [36] introduced a method for detecting deepfakes depending on the observations that existing DeepFake creation methods produce images of low resolutions that leave distinct artifacts when warped to be consistent with source faces. The faces were detected using the dlib software, and then four CNN models were trained to distinguish fake videos from real ones; VGG16, ResNet152, ResNet101, and ResNet50. This detection method was not robust regarding multiple video compression. Koopman et al. [37] analyzed the photo response non-uniformity (PRNU) noise pattern of the video frames to detect the deepfakes because it was expected that the manipulated facial region affects the local PRNU pattern in frames. The PRNU analysis demonstrated a noteworthy difference in the scores of mean-normalized cross correlations between real and deepfake videos. This work was only applied to a very small dataset. The work in [38] proposed a method for detecting the deepfakes based on the observation that such fakes were generated by splicing synthesized face area into the source image. First, the facial landmarks were extracted from face frames, and then the 3D head poses were estimated. After that, the computed difference of the head poses was fed as an input feature vector to the Support Vector Machine (SVM) classifier to differentiate between real and deepfake videos. The performance of this work degraded for blurry images due to the difficulties to estimate facial landmark locations. The work in [39] presented a deepfake detection method based on analyzing the frequency domain of real and fake face frames using discrete Fourier transform followed by applying the azimuthal average to produce a 1D feature representation vector. This feature vector was then fed to three machine learning classifiers; logistic regression, SVM, and K-means clustering.

Güera and Delp [40] proposed a temporal deepfake video detection method exploiting the incoherence in illumination across fake video frames that causes flicker artifacts in the face area. The Inception V3 was used to extract features from video frames, and the LSTM was then trained on these features to judge videos authenticity. This method yielded good performance on videos of length less than 2 s. Sabir et al. [41] observed that the artificial face generation methods do not usually enforce temporal coherence in the synthesis process. As a result, they first proposed to detect, crop, and align faces from video frames. Then, they applied either DenseNet or ResNet50 with bidirectional GRU on these aligned faces to learn the temporal artifacts to detect synthesized faces in video frames. The work in [42] proposed a method to detect deepfakes depending on the fact that the forged videos lack some physiological signals, eye blinking, in synthesized faces. First, the faces were detected from video frames using the dlib software and aligned based on facial landmarks. The areas corresponding to eyes were cropped out, and the lack of eye blinking in videos is then detected by determining the openness degree of an eye using VGG16-LSTM. Although this method achieved good performance, it cannot detect the manipulations in videos with closed eyes, frequent eye blinking, or realistic eye blinking. The work in [43] exploited the dissimilarities of the optical flow field across frames as a clue to distinguish between real and deepfake videos. The estimated optical flow was then given as input to two CNN models; VGG16 and ResNet50.

In [44], two shallow CNN models, Meso-4 and MesoInception-4, have been introduced focusing on the mesoscopic properties of forged content. This method detected manipulations in videos efficiently with a little computational cost; however, its performance degraded on low-quality videos. Rossler et al. [31] employed six

methods for detecting deepfake videos based both handcrafted and learned features. The faces were detected using a face tracking method. Then, the features were extracted from these faces and fed into six classifier models to distinguish videos authenticity: co-occurrence matrix-SVM, co-occurrence matrix-CNN, constrained CNN, Stats-2L network, MesoInception-4, and XceptionNet. Although the detection performance was good with XceptionNet, it degraded on compressed videos. Nguyen et al. [45] employed a multi-task learning-based designed CNN to simultaneously detect and locate forged content in videos. The method used an autoencoder for the classification of manipulated content, while it applied a y-shaped decoder to share the gained information for the segmentation and reconstruction tasks. The detection performance of this method degraded over unseen instances. In [46], the multi-task CNN was used to detect faces from video frames, where the EfficientNet-b5 was applied to these faces to extract the visual features. Then, the automatic face weighting mechanism along with bidirectional GRUs was employed to learn the temporal information and detect the manipulated videos. Wang et al. [47] proposed a method called FakeSpotter to detect the synthesized faces by monitoring neuron behaviors in deep face recognition systems with a simple classifier that consists of five fully connected layers. The VGG-Face with ResNet-50 was used to capture the activated neurons helping to get more subtle features that are significant for detecting the fakes. The work in [48] suggested using a part of VGG-19 to extract the hidden features from the detected faces. These features were fed as input to three primary capsules and two output capsules dedicated to real and fake images. Ismail et al. [22] proposed a hybrid method named YIX to discover inconsistencies and artifacts in spatial information of the forged video frames and then judge the authenticity of videos. This method used the YOLO detector to extract faces from video frames. Then, a fine-tuned InceptionResNetV2 model was employed as a feature extractor, followed by the XGBoost model as a classifier to distinguish a deepfake video from a genuine one. In [23], a method to detect the spatiotemporal discrepancies in deepfake videos was introduced. This method employed a refined version of the YOLO face to detect faces from video frames. Then, a fine-tuned EfficientNet-b5 Bidirectional-LSTM (Bi-LSTM) with a fully connected layer was employed to extract the spatial-temporal features and detect the video's authenticity.

In [49], a method for detecting the lip-sync deepfakes was proposed. The distances between mouth landmarks were employed as visual features together with the Mel Frequency Cepstral Coefficients (MFCC) as audio features. The principal component analysis was then applied to reduce the dimensionality of the joint visual-audio feature vector, which was fed as input to different classifiers: Gaussian mixture model, SVM, multilayer perceptron, and LSTM. This method with the LSTM classifier achieved better performance than other classifiers, however; its performance dropped as the training instances decreased. Korshunov et al. [50] demonstrated that replacing the MFCC audio features in [49] with embeddings from a deep neural network achieved a significant performance improvement in detecting the lip-sync deepfakes on challenging publicly datasets. The work in [51] exploited the inconsistencies between the mouth shape dynamics and the spoken phonemes to detect the lip-sync deepfake videos based on using either vertical intensity profile or CNN.

Since the existing deepfake detection methods have focused on either exploiting specific spatial and temporal artifacts left over from the creation methods or data-driven classification, the proposed method employs both directions. This aims to discover different features using various approaches to improve the detection method's performance. *First,* the proposed method targets some kind of spatial artifacts: visible splicing boundaries [52], using a CNN method based on one of the computer vision features; HOG. This is because the creation method synthesizes the target face per frame, and this may produce abnormal changes to several features. The HOG is a local descriptor that describes a pixel in a face frame based upon its local horizontal and vertical surroundings. It creates histograms of local intensity gradients to describe the appearance and shape of local objects. The HOG representation captures gradient structure or edge information, as well as texture near edges, corresponding to the underlying local shape [53]. It is based on calculating the color variation at all pixels of localized areas of face frame in both x and y directions, which yields two gradient-x to x axis; x axis derivative, and gradient-y to y axis; y axis derivative [54]. Thus, the HOG-based CNN feature extraction method may produce specific artifacts by learning the difference between the spatial HOG feature of genuine and deepfake face. *Second,* the proposed method also applies data-driven classification by employing the proposed ameliorated Xception deep network to automatically find informative spatial hierarchical features representing face frames. These along with GRUs sequence and dense layers are used to detect the temporal incoherence and inconsistencies among the video frames and then classify videos as genuine or deepfake. In addition, to boost the applicability of the proposed method in real-world scenarios, it is trained using a diverse CelebDF-FaceForencics++ (c23) dataset and tested using high-quality hyper-realistic videos from the CelebDF dataset.

3 Methods and materials

The proposed deepfake video detection method consists of three fundamental phases: data pre-processing phase, feature extraction-merge phase, and classification phase. These phases are displayed in Fig. 1, and each of them will be detailed in the following subsections.

3.1 Data pre-processing

In this phase, the video is transformed into frames. Then, the faces are detected and cropped out from video frames since most forged methods focus on manipulating faces. The YOLO detector, one of the most popular state-of-the-art face detectors, is used for the face detection process [55, 56]. It is characterized by producing fewer false-positive samples compared to other face detectors: BlazeFace, dlib, and MTCNN [21–23]. The size of the detected bounding box of the face is increased by 22% relative to its height and width. This adds a large region of the head that may contains artifacts aiding to judge deepfakes. After that, the extracted face images are resized to 224 × 224, and its pixels are normalized to belong to [− 1,1].

3.2 The proposed "feature extraction-merge" hybrid method

In this phase, a hybrid solution for the deepfake detection problem is proposed based on the two proposed feature extraction methods: HOG-based CNN and an ameliorated XceptionNet. The two proposed components are explained hereafter. The different spatial features are extracted from the detected faces based on those two feature extraction methods. These features are then merged together aiming to get the optimal spatial representation of video information. After that, the temporal features are learned to discover the temporal incoherence among the manipulated video frames.

3.2.1 The proposed HOG-based CNN

The HOG method has been proposed by Dalal and Triggs, where the local features of an object in image can be effectively described by direction of the contours or the intensity distribution of gradients [57]. First, the image is divided into small spatial blocks and each block is divided into smaller areas called cells. Then, for each cell, the gradient magnitude $m(x_i, y_i)$ and orientation $\theta(x_i, y_i)$ are calculated based on the horizontal $G_x(x_i, y_i)$ and vertical $G_y(x_i, y_i)$ gradient information of each pixel (x_i, y_i). Those G_x and G_y gradients are computed using 1-D discrete derivations filter masks $[-0.17em1, 0, 1]$ and $[-0.17em1, 0, 1]^T$. They are also calculated singly for each color channel, and for each pixel, it selects the channel with the largest gradient magnitude. The following equations describe m and θ calculations:

$$m(x_i, y_i) = \sqrt{G_x(x_i, y_i)^2 + G_y(x_i, y_i)^2} \tag{1}$$

$$\theta(x_i, y_i) = \arctan\left(\frac{G_y(x_i, y_i)}{G_x(x_i, y_i)}\right) \tag{2}$$

After that, the histograms are created and normalized for each block built on cells. Those histograms, which are presented as vectors, are combined to configure the overall HOG visual feature descriptor of an image [58–60]. Figure 2 depicts the HOG feature extraction process.

Here, the HOG feature is extracted from the face and then is reshaped to be fed as an input to the proposed CNN to reduce the feature size and learn the most distinctive spatial gradient orientation features of the genuine and deepfake face as shown in Fig. 1. The difference between the HOG feature of the genuine and deepfake face frame in the gradient information and texture of the forehead, nose and its around, and cheeks areas is shown in Fig. 3.

The proposed CNN is a lightweight architecture inspired by using shortcut connections of residual networks with a stack of separable depth convolution layers [61]. The shortcut connections is known to effectively deal with the problem of vanishing gradients and enhance performance.

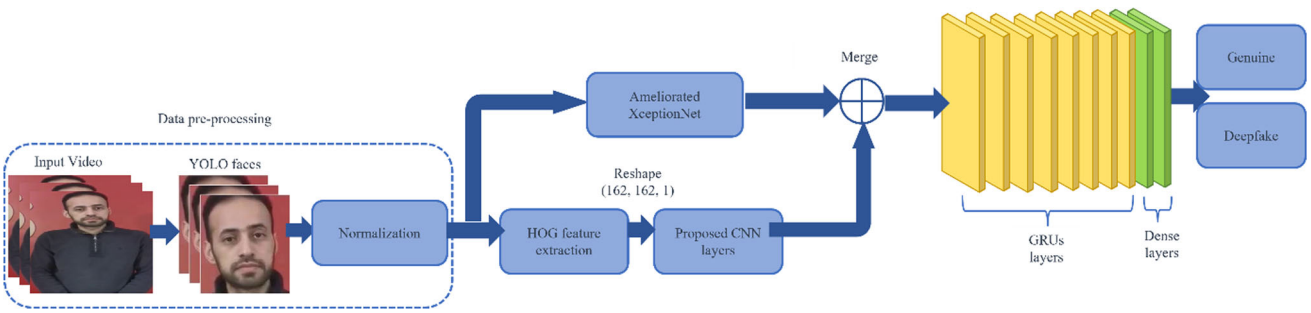

Fig. 1 The suggested YF method diagram

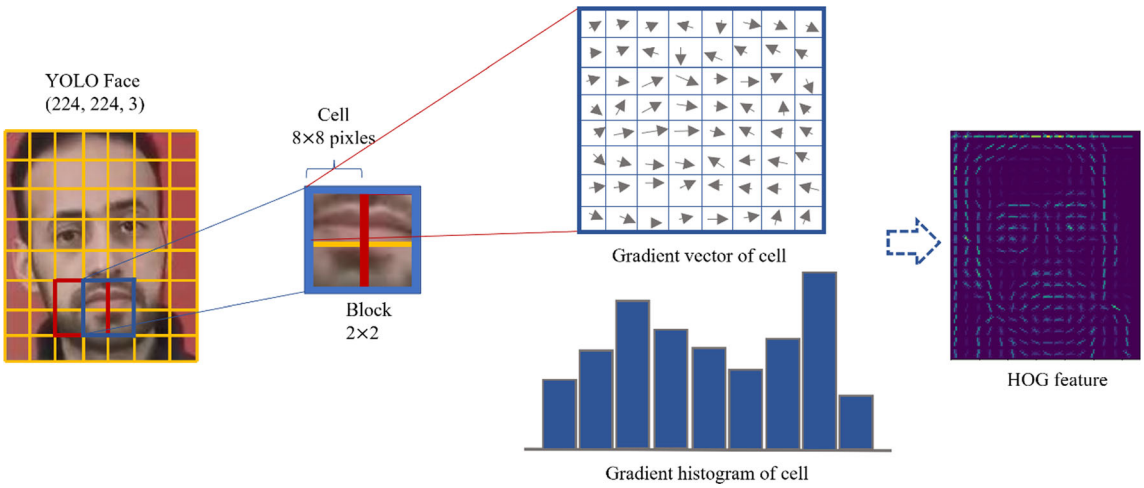

Fig. 2 The description of the HOG feature extraction process

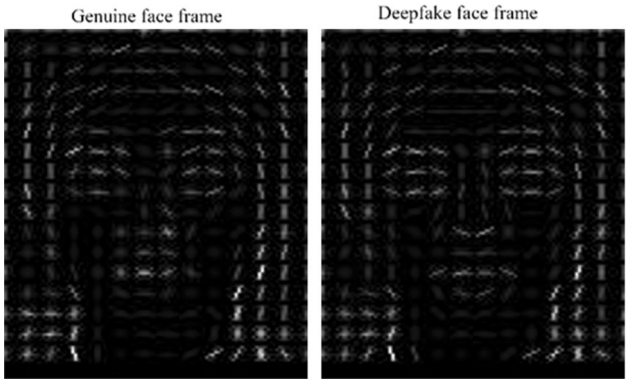

Fig. 3 The HOG feature of genuine and deepfake face frame

The separable depth convolution layers are a type of convolution that is believed to be substantially far more efficient in terms of computational complexity. The architecture diagram of the proposed CNN is shown in Fig. 4. It consists of several building blocks including convolution, batch normalization, Rectified Linear Unit (ReLU), separable depth convolution, shortcut connections, densely connected, dropout, and pooling layers. The convolution contains kernels that convolve over the height and width of face frames and produces feature maps. Each convolution layer with a 3×3 kernel size is followed by batch normalization and ReLU activation layers, while each separable convolution layer is followed by a batch normalization layer. The batch normalization is used to normalize data into zero mean and unit variance for each mini-batch, which accelerates the training. The ReLU activation adds nonlinearity to the layer by forcing all negative input values to be zeros, which accelerates the training, reduces the vanishing gradient problem, and contributes to achieve better predictions and reduce the overfitting. The pooling layers are employed to minimize the parameters' number and computation time by reducing the feature dimension. The dropout layer drops out the neurons randomly with a desire probability rate per training step, which helps to prevent overfitting. The length of the output feature vector is 1024. The details of the proposed CNN layers are shown in Table 1.

3.2.2 The proposed ameliorated XceptionNet method

The Xception network, which stands for extreme Inception, employs the concept of a depth-wise separable convolution to decouple the channel and spatial dimensions of an image [61]. The Xception architecture starts with two convolution layers having 32 and 64 filters, respectively, each with a 3×3 kernel size. Each convolution layer is followed by the ReLU activation layer. This is followed by five blocks, where each input to the Xception block is passed via two separable convolution layers followed by a max-pooling layer and is also passed via a pointwise convolution layer through shortcut residual connection except the fourth block. The fourth block comprises three separable convolution layers, and this Xception block iterates eight times. The last Xception block is followed by a couple of layers: separable convolution and ReLU activation. This is followed by a global average pooling layer. After that, the densely connected layers are added where the last one being the output layer [62]. In the Xception architecture, all convolution and separable convolution layers are followed by batch normalization layer. In addition, all separable convolution layers are preceded by the ReLU activation layer except the first one and the last two layers.

As depicted in Fig. 5, the Xception network is ameliorated by injecting two layers before the ReLU activation layer of the last separable convolution layer. These two layers are the typical convolution and batch normalization

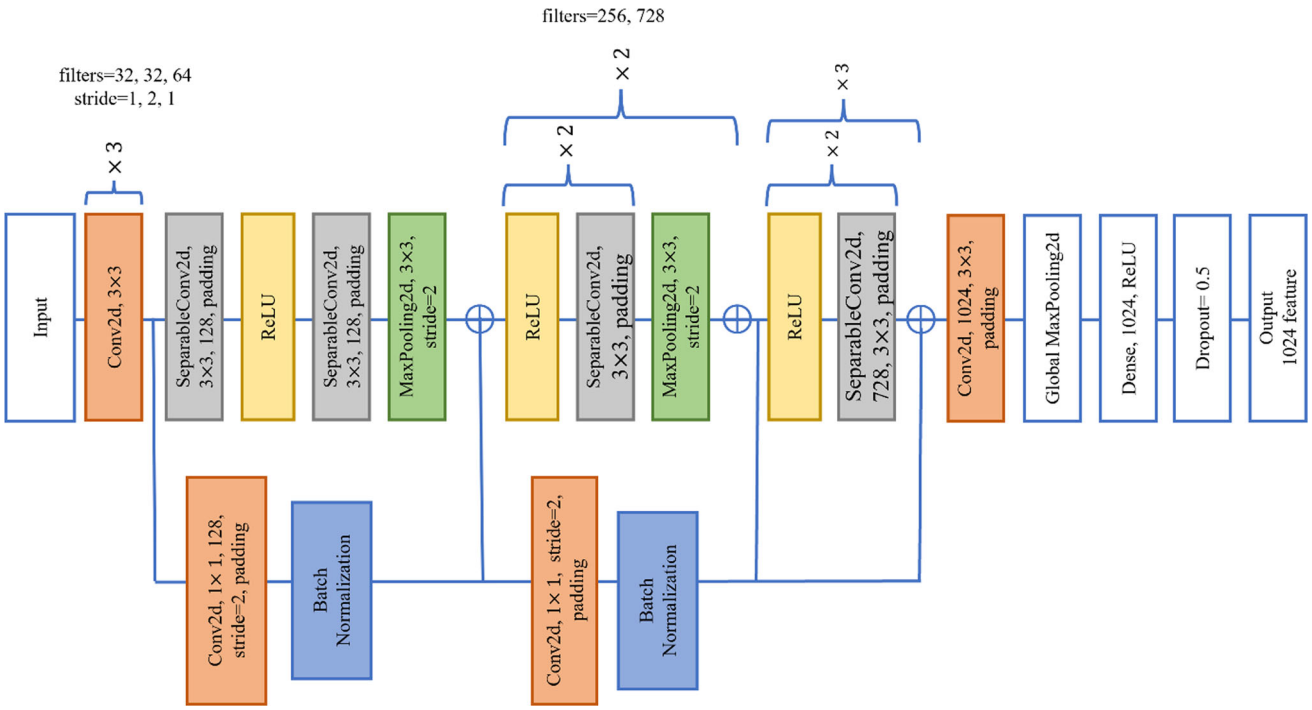

Fig. 4 The proposed CNN architecture diagram

Table 1 The proposed CNN layers details

Layers	Activations
Input	162 × 162 × 1
C, B, R	162 × 162 × 32
C, B, R	79 × 79 × 32
C, B, R	77 × 77 × 64
S, B, R, S, B	77 × 77 × 128
C, M, B, Add (M, B), R	39 × 39 × 128
S, B, R, S, B	39 × 39 × 256
C, M, B, Add_1 (M, B), R	20 × 20 × 256
S, B, R, S, B	20 × 20 × 728
C, M, B, Add_2 (M, B), [R, S, B] × 3, Add_3 (B, Add_2)	10 × 10 × 728
[[R, S, B] × 3, Add_i (B, Add_j)] × 2; i = 3,4 and $j = i - 1$	10 × 10 × 728
C, B, R	10 × 10 × 1024
G, DS, D	1024

C, convolutional layer; S, separable convolution layer; B, batch normalization layer; R, rectified linear unit layer; M, max-pooling layer; G, global average pooling layer; D, dropout layer; DS, densely connected layer

layers. Then, after the global average pooling layer, the following layers are injected: densely connected layer with ReLU activation function, and dropout layer to prevent the overfitting. The detected face is now passed into this ameliorated network to obtain 1024 features. The ameliorated XceptionNet layers are presented in Table 2. The proposed ameliorations in the XceptionNet aim to produce an informative spatial representation of face hierarchal features. This assists to enhance the deepfake detection method performance in real-world scenarios.

3.2.3 Features merge

In this step, the spatial features extracted from the HOG-based CNN are merged with those of the ameliorated XceptionNet to get 2048 optimal spatial features for each face. Thus, the output of the features merge step for a given

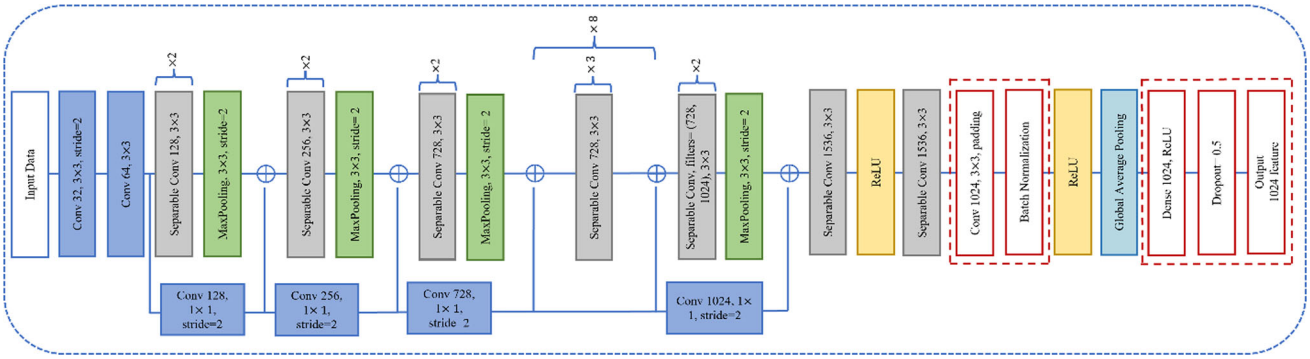

Fig. 5 The ameliorated XceptionNet architecture

Table 2 The ameliorated XceptionNet layers details

Layers	Activations
Input	224 × 224 × 3
C, B, R	111 × 111 × 32
C, B, R	109 × 109 × 64
S, B, R, S, B	109 × 109 × 128
C, M, B, Add (M, B), R	55 × 55 × 128
S, B, R, S, B	55 × 55 × 256
C, M, B, Add_1 (M, B), R	28 × 28 × 256
S, B, R, S, B	28 × 28 × 728
C, M, B, Add_2 (M, B)	14 × 14 × 728
[[R, S, B] × 3, Add_i (B, Add_j)] × 8; i = 3, …,10 and $j = i - 1$	14 × 14 × 728
R, S, B, R	14 × 14 × 728
S, B	14 × 14 × 1024
C, M, B, Add_11 (M, B)	7 × 7 × 1024
S, B, R	7 × 7 × 536
S, B	7 × 7 × 2048
C, B, R	7 × 7 × 1024
G, DS, D	1024

C, convolutional layer; S, separable convolution layer; B, batch normalization layer; R, rectified linear unit layer; M, max-pooling layer; G, global average pooling layer; D, dropout layer; DS, densely connected layer

dataset is (videos, frames per video, 2048). This outcome will be utilized as an input to the next step to learn the temporal features of genuine and fake videos.

3.2.4 Temporal feature extraction in the proposed hybrid method

In this step, a sequence of the Gated Recurrent Units (GRUs) is employed to learn the temporal features of videos. The GRU is an improved version of the standard RNN with memory cells. It aims to solve the gradient vanishing problem of RNN with a gating mechanism. It employs the update and reset gates to modulate the flow of information inside the cell state. At each time step t, the cell takes the input sequence x_t and the hidden state h_{t-1} which comes from the previous time step $t-1$, and outputs a new hidden state h_t which again passes to the next time step. The reset gate is employed to determine the amount of past information that needs to forget. The update gate helps to decide the amount of past information that needs to be passed to the future state. The following formulae are utilized in the GRU output calculations at each time step [33, 63, 64]:

$$reset_t = sigmoid(W_r x_t + U_r\, hidden_{t-1}) \tag{3}$$

$$update_t = sigmoid(W_u x_t + U_u hidden_{t-1}) \tag{4}$$

$$hid'den_t = \tanh(W_h x_t + U_h(reset_t \odot \ hidden_{t-1})) \tag{5}$$

$$hidden_t = (1 - update_t) \odot \ hidden_{t-1} + update_t \odot \ hid'den_t \tag{6}$$

The variables x_t, $reset_t$, $update_t$, $hid'den_t$, and $hidden_t$ denote the input vector, reset gate, an update gate, candidate activation vector, and output vector, respectively, and all W and U variables represent the weight matrices. The symbol $\odot$ denotes an element-wise product.

As each manipulated face frame is generated individually, it inevitably causes an evident flicker and discontinuity of the face surface. To exploit this vulnerability, the optimal spatial extracted features of each video are fed into a sequence of GRUs to learn the temporal incoherence among the forged video frames. It had been agreed upon that using a sequence of GRUs might ameliorate the overall performance [46]. As illustrated in Fig. 1, this sequence consists of eight GRUs where each GRU couple is employed with units 2048, 1024, 512, and 256, respectively. The experimental results showed that the proposed system performed better as we increased the number of GRUs. The discovered optimal value is eight GRUs after which the performance degrades. The GRUs layers' details are described in Table 3.

3.3 Classification

Once the temporal features have been extracted, a densely connected layer with 256 units and ReLU activation function is added on the top of the features. The ReLU function, $g(x) = \max(0, x)$, avoids the vanishing gradient problem and makes it possible to learn complex relations in the data. Finally, a densely connected layers with 2 units representing the output classes; genuine, and deepfake, and Softmax activation function is added to distinguish the genuine video from the deepfake one.

Table 3 The GRUs layers description of the proposed YF method

Layer (type)	Output shape	Parameters number
main_input (InputLayer)	[(None, 10, 2048)]	0
gru (GRU)	(None, 10, 2048)	25,178,112
gru_1 (GRU)	(None, 10, 2048)	25,178,112
gru_2 (GRU)	(None, 10, 1024)	9,443,328
gru_3 (GRU)	(None, 10, 1024)	6,297,600
gru_4 (GRU)	(None, 10, 1024)	6,297,600
gru_5 (GRU)	(None, 10, 512)	2,362,368
gru_6 (GRU)	(None, 10, 512)	1,575,936
gru_7 (GRU)	(None, 10, 256)	59,1360
gru_8 (GRU)	(None, 256)	394,752
dense (Dense)	(None, 256)	65,792
dense_1 (Dense)	(None, 2)	514

Total parameters number: 77,385,474. Trainable parameters number: 77,385,474. Non-trainable parameters number: 0

3.4 Dataset

The proposed hybrid method has been applied to a diversifiable real and fake videos dataset, namely: CelebDF-FaceForencics++ (c23), that results by combining two popular datasets: CelebDF and FaceForencics++ (c23). This helped to assess the robustness of the proposed fake detection method and boosted the applicability of the proposed method in the real world [22, 23].

The CelebDF dataset consists of 590 genuine videos of celebrities selected from YouTube that vary in ethnic groups, genders, and ages, and 5639 generated DeepFake videos. It also includes 300 additional genuine videos of random subjects collected from YouTube [52]. The FaceForensics++ dataset consists of 1000 genuine videos that have been altered with four face manipulation methods: Deepfakes, FaceSwap, Face2Face, and NeuralTextures. It is generated with three compression levels: high (c40), light (c23), and raw (H.264) [26].

To train the YF method, 1424 genuine and fake CelebDF videos are combined with 1424 real and deepfake FaceForencics++ (c23) videos. The training set is split into training and validation subsets. To test the proposed YF method, the CelebDF test set which originally consists of 518 genuine and fake videos is employed. This situation simulates real-world scenarios because the CelebDF has high visual quality videos that resemble those circulated on the Internet where they are generated using an improved deepfake generation algorithm [52].

4 Time complexity analysis of the proposed method

In general, the time complexity of CNN comprises convolution, pooling, and dense layers. The pooling and dense layers use just 5–10% of the total calculation time, whereas convolution layers take the overwhelming majority of the computational time. For simplicity, the time complexity of the proposed method is assessed based on the convolution layers [65–67].

The first phase of the proposed method, data pre-processing, is based on using the YOLO face detector. The YOLO detector applied here relies on the pre-trained YOLOv3 which is based on the darknet-35 network architecture. This architecture employs successive convolutional layers with some residual connections to produce the predicted bounding boxes from frames with a confidence score and then loop through all the bounding boxes, filtering out the ones with low scores. The non-maximum suppression is applied to the remaining boxes. This eliminates overlapping bounding boxes [56, 68]. Thus, the

calculation of the bounding boxes' coordinates that represents the detected YOLO faces from a frame depends on the computational complexity of the darknet-53 network's convolution layers. The first phase has $O(\sum_{l=1}^{L} C^2K^2hw)$ time complexity. The symbol l represents the convolution layer index, L denotes the convolution layers number, C represents the channel size, K refers to the kernel size, and h and w represent the spatial height and width of the output feature map, respectively [65–67, 69]. The second phase, the feature extraction-merge hybrid method, is based on merging the extracted features from the HOG-based CNN and an ameliorated XceptionNet and then feeding the merge outcome to GRUs sequence for learning the temporal features of genuine and deepfake videos. The calculation of the face's HOG feature has a time complexity $O(HW)$ where H and W represent the height and width of the face frame, respectively [70]. The extracted HOG feature of the face is then fed into the proposed CNN for feature reduction and further learning. Since the proposed CNN mainly comprises convolution layers and separable depth convolution layers, it has $O(\sum_{l=1}^{d} C^2K^2hw)+ O(\sum_{l_1=1}^{d_1} Chw(C+K^2))$ time complexity. The first and second terms represent the total computational complexity of convolution and separable depth convolution layers, respectively. The symbol $d(<L)$ denotes the convolution layers number, l_1 denotes the separable depth convolution layer index, and d_1 refers to the number of separable convolution layers [65–67, 69]. Thus, for each face frame, the HOG-based CNN has $O(HW + \sum_{l=1}^{d} C^2K^2hw + \sum_{l_1=1}^{d_1} Chw(C+K^2))$ time complexity. The ameliorated XceptionNet mainly consists of convolution layers and separable depth convolution layers, it has $O(\sum_{l=1}^{d} C^2K^2hw)+ O(\sum_{l_1=1}^{D_1} Chw(C+K^2))$ time complexity per frame where $D_1(>d_1)$ denotes the number of separable depth convolution layers. Consequently, for each frame, the time complexity of applying the HOG-based CNN and the ameliorated XceptionNet and merging their outcomes is $O(HW + \sum_{l=1}^{2d} C^2K^2hw + \sum_{l_1=1}^{d_1+D_1} Chw(C+K^2))$. Since the input to the GRU is a sequence of m frames representing a given video, the time complexity of a single GRU is $O(md_h^2 + md_hd_i)$ where d_h and d_i represent dimensions of hidden state and input, respectively [71, 72]. The GRUs sequence has a time complexity $O(\sum_{k=1}^{K}(md_h^2 + md_hd_i))$ where k and K represent the GRU index and the number of GRUs, respectively. As a result, the total time complexity per video in the second phase is $O(m.[HW + \sum_{l=1}^{2d} C^2 \quad K^2hw + \sum_{l_1=1}^{d_1+D_1} Chw(C+K^2)] + \sum_{k=1}^{K}(md_h^2 + md_hd_i))$.

Finally, assume the sample size is v, representing the total number of videos, and the training process has e epochs. The total theoretical time complexity of the proposed method has been proven to be equal to:

$$O\left(ev.\left\{m.\left[HW + \sum_{l=1}^{L+2d} C^2K^2hw + \sum_{l_1=1}^{d_1+D_1} Chw(C+K^2)\right] + \sum_{k=1}^{K}\left(md_h^2 + md_hd_i\right)\right\}\right). \tag{7}$$

5 Case study, results and analysis

The YF proposed deepfake video detection method is trained by the CelebDF-FaceForencics++ (c23) dataset, and the assessment is conducted on the original CelebDF test set. The training set is divided into random subsets: train and validation.

To evaluate the classification performance of the suggested YF method for detecting the deepfake videos, several standard metrics are applied: an area under the receiver operating characteristic (AUROC) curve, accuracy, precision, recall, F-score, sensitivity, specificity, and confusion matrix.

The receiver operating characteristic (ROC) curve is a more robust approach in evaluating predictive methods. It provides a graphical manner to visualize the performance in terms of sensitivity, true positive rate (*tpr*), against 1-Specificity, false positive rate (*fpr*), across a series of thresholds. The closer the curve to the top left, the more efficient the detection method's performance [73]. The AUROC score measures the discriminative ability of the proposed learning method to correctly classify positive and negative random instances and is employed as a single number summary of the ROC. It can be calculated as follows [74]:

$$\text{AUROC} = \int_0^1 tpr\left(fpr^{-1}(\text{x})\right)\text{d}x = p(x_2 > x_1) \tag{8}$$

where x_2 is the predicted positive instance and x_1 is the predicted negative instance. The higher the AUROC score, the better the detection method performance is at distinguishing between deepfake and genuine classes. The mathematical expressions of the rest of the standard evaluation metrics are defined as follows [75]:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FN} + \text{FP}} \tag{9}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{10}$$

$$\text{Recall} = \text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{11}$$

$$F{-}score = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{12}$$

$$Specificity = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{13}$$

where TP represents the number of actual positive instances that are predicted as positive. FP denotes the number of actual negative instances that are predicted as positive. The FN represents the number of actual positive instances that are predicted as negative. TN represents the number of actual negative instances that are predicted as negative.

The proposed YF deepfake detection method is trained for 42 epochs using the stochastic gradient descent optimizer [76] with a learning rate started from 0.002 and decayed by a factor of 0.000004, and momentum 0.9. This updates the weight parameters and aims to reduce the difference between the target and predicted outcomes. The batch size is set to 64. The cross-entropy function is used as a loss function to measure the efficiency of the proposed YF method, and its equation [77] is defined as follows:

$$\text{Loss} = -\frac{1}{N}\sum_{i=1}^{N}(y_i log(p_i) + (1 - y_i)\log(1 - p_i)) \tag{14}$$

where N is the total number of videos and y_i represents the ground truth label for the ith video, while p_i represents its predicted probability.

The experiments are conducted on the following benchmark: an OMEN HP laptop that has an Intel (R) Core (TM) i7-9750H CPU with 16 GB, an RTX 2060 GPU with 6 GB, and Windows 10. The Python language is used to implement the suggested YF method. Tensorflow, Keras, OpenCV, Skimage, Numpy, OS, Random, and PIL are some Python libraries utilized for achieving the proposed method.

The experimental results show that the proposed approach outperforms the recent state-of-the-art approaches. Figure 6 shows the accuracy and loss curves for the proposed YF method. The confusion matrix of the YF method for detecting the deepfake videos is shown in Fig. 7. Additionally, Fig. 8 depicts the ROC curve and the AUROC curve metric corresponding to the YF method performance. As can be seen from Fig. 8, the ROC curve is very close to the upper left confirming a high performance by the suggested YF method.

Table 4 shows the AUROC scores for the proposed approach compared to recent deepfake videos detection approaches. As seen from Table 4, the YF method scored the highest performance. It recorded an AUROC score of

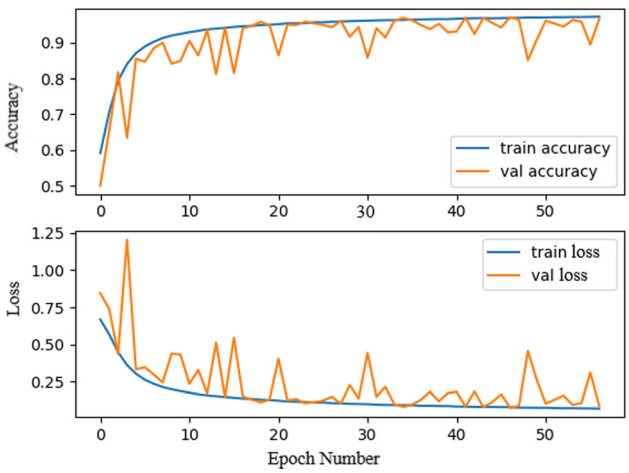

Fig. 6 The accuracy and loss curves of the proposed YF method on training and validation sets

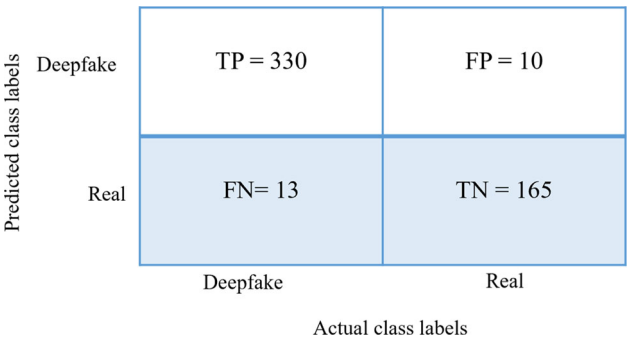

Fig. 7 The confusion matrix visualization of the proposed YF method

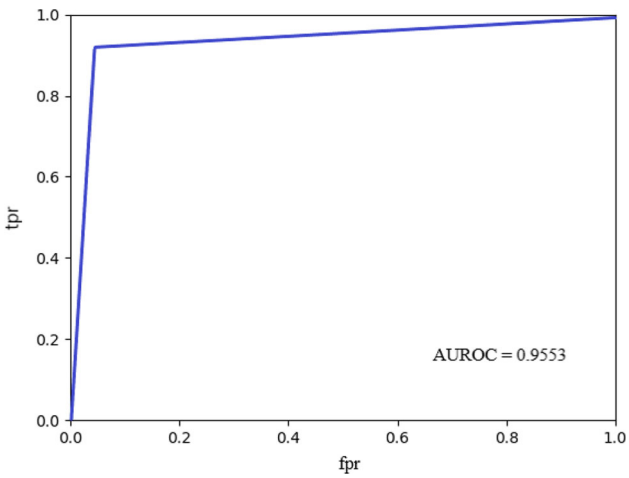

Fig. 8 The ROC curve corresponding to the proposed YF method performance

95.53% which exceeds that of other current detection methods [22, 23, 46, 52] with an average increase of 7.695%. In addition, the running time is recorded in Table 4. It is susceptible to implementation and hardware

Table 4 The AUROC score and the running time for the proposed YF method compared to previous detection methods when trained by the CelebDF-FaceForencies++ (c23) dataset and tested by the CelebDF test set.

Method	AUROC score (%)	Running time (h)
YF proposed method	95.53%	13.31
InceptionResNetV2+XGBoost [22]	90.62%	15.59
Fine-tuned EffecientNet-b5+Bi-LSTM [23]	89.35%	14.93
EfficientNet-b5+Bi-GRU [46]	86.82%	14.08
XceptionNet [52]	84.55%	11.55

[63]. The previously described benchmark has been used to run four algorithms and the proposed one. The running time of the YF proposed method is 13.31 h, and it is lower than that of current detection methods [22, 23, 46] except for the detection method based on the XceptionNet [52]. The XceptionNet has a slightly lower running time than that of the YF method because it converges faster than the YF, so it has a lower number of epochs. However, it achieves low performance and high loss compared to the YF. The YF method recorded 0.1004 loss, 95.56% accuracy, and 95.53% AUROC, while the XceptionNet recorded 0.5411 loss, 84.94% accuracy, and 84.55% AUROC.

Figure 9 shows the evaluation metrics-based comparative analysis of the YF proposed method with recent state-of-the-art methods. As can be seen from Fig. 9, the YF method has achieved higher performance compared to the other methods. The YF yields 95.53% AUROC, 95.56% accuracy, 97.06% precision, 96.21% recall, 96.63% F-score, 96.21% sensitivity, and 94.29% specificity.

Additionally, the YF method is evaluated without using the HOG-based CNN feature extraction method and achieved 94.38% AUROC, 94.40% accuracy, 95.88% precision, 95.6% recall, 95.7% F-score, 92.09% sensitivity, and 94.29% specificity. It can be concluded that merging two different feature extraction methods improved the learning process of the proposed detection method. It provided an optimal spatial representation of real and deepfake faces.

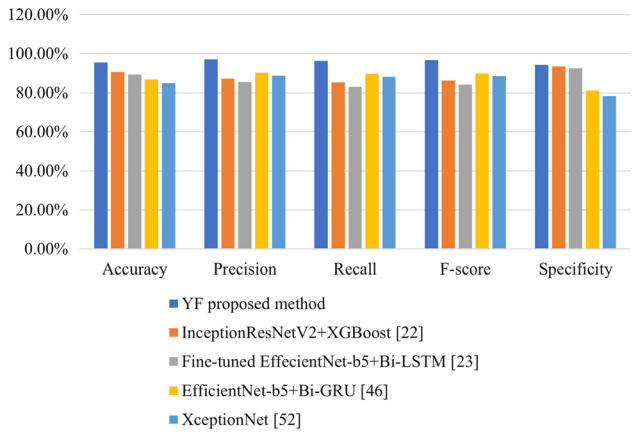

Fig. 9 The performance of the YF method compared to recent state-of-the-art detection methods when trained by the CelebDF-FaceForencics++ (c23) dataset and evaluated by the CelebDF test set

The proposed HOG-based CNN feature extraction method produces discriminative specific artifacts by learning the difference between the spatial HOG feature of a genuine and deepfake face. The ameliorated XceptionNet feature extraction method produced an informative spatial hierarchical representation of the face that helped to discriminate between the real and deepfake face. The GRUs, which are applied on the top of merged features, helped to adaptively capture the temporal incoherence among the deepfake video frames. The GRU is characterized by a simple structure with few parameters while ensuring that important features will not be lost during transmission among video frames. It saves a lot of time without immolating performance. Thus, the GRUs contributed to enhancing the overall performance of the proposed method. The experimental results have demonstrated the superiority of the YF suggested method compared to the current methods.

6 Conclusion and future work

This paper introduced a new spatiotemporal-based methodology, called YF, for detecting deepfake videos. The YOLO detector has been utilized for face detection from the frames of videos since it reduces the false-positive samples. Two different proposed feature extraction methods have been applied to the detected faces aiming to enrich learning while training the proposed method. Combining these methods, HOG-based CNN and ameliorated Xception network, has succeeded in producing an informative spatial representation of genuine and fake faces. The produced features are merged and fed as an input to a sequence of GRUs. This helps to discover the defects in the temporal domain across the deepfake videos frames. The proposed YF method has been trained on the CelebDF-FaceForencics++ (c23) dataset and evaluated on the CelebDF test set which has videos that resemble those in real life. The experimental results have shown a high performance of the YF method. The proposed method achieved 95.53% AUROC score, 95.56% accuracy, 97.06% precision, 96.21% recall, 96.63% F-score, 96.21% sensitivity, and 94.29% specificity. Moreover, the comparative analysis confirmed that the suggested YF method outperforms the recent state-of-the-art methods by an average improvement of 7.695% in terms of the AUROC score.

In the future work, more efforts are required to enhance the detection method so that it becomes lighter with achieving a higher performance level. Additionally, the proposed method may be expanded to discover the deepfakes in multimodal videos that include both visual-video and auditory modalities. Furthermore, a huge video dataset may be used to ameliorate the deepfake detection method's performance.

Author contribution AI: conceptualization, methodology, software, writing—original draft preparation; ME: conceptualization, methodology, writing—original draft preparation, supervision; MSZ: conceptualization, supervision; KE: conceptualization, methodology, writing—original draft preparation, supervision.

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB).

Data availability Datasets available online: https://github.com/ondyari/FaceForensics and https://github.com/yuezunli/celeb-deepfakeforensics (accessed on 05 June 2022).

Declarations

Conflict of interest The authors declare no conflict of interest.

References

1. Nguyen TT, Nguyen CM, Nguyen DT, Nguyen DT, Nahavandi S (2019) Deep learning for deepfakes creation and detection: a survey. arXiv preprint arXiv:1909.11573.
2. Atienza R (2020) Advanced deep learning with tensorFlow 2 and keras: Apply DL, GANs, VAEs, deep RL, unsupervised learning, object detection and segmentation, and more. Packt Publishing Ltd
3. Faceswap. https://github.com/deepfakes/faceswap
4. DeepFaceLab. https://github.com/iperov/DeepFaceLab
5. DFaker. https://github.com/dfaker/df
6. Faceswap-GAN. https://github.com/shaoanlu/faceswap-GAN
7. DeepFake-tf. https://github.com/StromWine/DeepFake_tf
8. Thies J, Zollhofer M, Stamminger M, Theobalt C, Nießner M (2016) Face2face: real-time face capture and reenactment of rgb videos. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2387–2395
9. Wu W, Zhang Y, Li C, Qian C, Loy CC (2018) Reenactgan: learning to reenact faces via boundary transfer. In: Proceedings of the European conference on computer vision (ECCV), pp 603–619
10. Thies J, Zollhöfer M, Theobalt C, Stamminger M, Nießner M (2018) Headon: real-time reenactment of human portrait videos. ACM Trans Gr (TOG) 37(4):1–13
11. Suwajanakorn S, Seitz SM, Kemelmacher-Shlizerman I (2017) Synthesizing obama: learning lip sync from audio. ACM Trans Gr (ToG) 36(4):1–13
12. Radford A, Metz L, Chintala S (2015) Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.
13. Karras T, Aila T, Laine S, Lehtinen J (2017) Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196.
14. 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
15. Brock A, Donahue J, Simonyan K (2018) Large scale GAN training for high fidelity natural image synthesis. arXiv preprint arXiv: 1809.11096
16. Perarnau G, Van De Weijer J, Raducanu B, Álvarez JM (2016) Invertible conditional gans for image editing. arXiv preprint arXiv:1611.06355
17. Choi Y, Choi M, Kim M, Ha JW, Kim S, Choo J (2018) Stargan: unified generative adversarial networks for multi-domain image-to-image translation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8789–8797
18. He Z, Zuo W, Kan M, Shan S, Chen X (2019) Attgan: facial attribute editing by only changing what you want. IEEE Trans Image Process 28(11):5464–5478
19. Arik SO, Chen J, Peng K, Ping W, Zhou Y (2018) Neural voice cloning with a few samples. arXiv preprint arXiv:1802.06006
20. Luong HT, Yamagishi J (2020) Nautilus: a versatile voice cloning system. IEEE/ACM Trans Audio Speech Lang Process 28:2967–2981
21. 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
22. Ismail A et al (2021) A new deep learning-based methodology for video deepfake detection using XGBoost. Sensors 21(16):5413
23. Ismail A, Elpeltagy M, Zaki M, ElDahshan KA (2021) Deepfake video detection: YOLO-face convolution recurrent approach. PeerJ Comput Sci 7:e730. https://doi.org/10.7717/peerj-cs.730
24. Zhang J, Han Y, Jiang J (2016) Tucker decomposition-based tensor learning for human action recognition. Multimed Syst 22(3):343–353
25. Patwary MJA, Parvin S, Akter S (2015) Significant HOG-histogram of oriented gradient feature selection for human detection. Int J Comput Appl 132(17):20
26. Carcagnì P, Del Coco M, Leo M, Distante C (2015) Facial expression recognition and histograms of oriented gradients: a comprehensive study. SpringerPlus 4(1):1–25
27. Xin W, Gongde G, Hui W (2015) A multiscale method for HOG-based face recognition. In: Proceedings of the IEEE international conference on intelligent robotics and applications, Portsmouth, UK, pp 24–27
28. Fadl S, Han Q, Qiong L (2020) Exposing video inter-frame forgery via histogram of oriented gradients and motion energy image. Multidimens Syst Signal Process 31(4):1365–1384
29. Mohan M, Preetha VH (2017) Gabor filter—HOG based copy move forgery detection. J Electron Commun Eng 2:41–45
30. Subramanyam AV, Emmanuel S (2012) Video forgery detection using HOG features and compression properties. In: 2012 IEEE

14th international workshop on multimedia signal processing (MMSP), IEEE, pp 89–94
31. 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
32. Kumar A, Bhavsar A, Verma R (2020) Detecting deepfakes with metric learning. In: 2020 8th International workshop on biometrics and forensics (IWBF), IEEE, pp 1–6
33. Shen G, Tan Q, Zhang H, Zeng P, Xu J (2018) Deep learning with gated recurrent unit networks for financial sequence predictions. Procedia Comput Sci 131:895–903
34. Lyu S (2020) Deepfake detection: current challenges and next steps. In: 2020 IEEE international conference on multimedia and expo workshops (ICMEW), IEEE, pp 1–6
35. Masood M, Nawaz M, Malik KM, Javed A, Irtaza A (2021) Deepfakes generation and detection: state-of-the-art, open challenges, countermeasures, and way forward. arXiv preprint arXiv:2103.00484
36. Li Y, Lyu S (2018) Exposing deepfake videos by detecting face warping artifacts. arXiv preprint arXiv:1811.00656
37. Koopman M, Rodriguez AM, Geradts Z (2018) Detection of deepfake video manipulation. In: The 20th Irish machine vision and image processing conference (IMVIP), pp 133–136
38. Yang X, Li Y, Lyu S (2019) 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
39. Durall R, Keuper M, Pfreundt FJ, Keuper J (2019) Unmasking deepfakes with simple features. arXiv preprint arXiv:1911.00686
40. Güera D, Delp EJ (2018) Deepfake video detection using recurrent neural networks. In: 2018 15th IEEE international conference on advanced video and signal based surveillance (AVSS), IEEE, pp 1–6
41. Sabir E, Cheng J, Jaiswal A, AbdAlmageed W, Masi I, Natarajan P (2019) Recurrent convolutional strategies for face manipulation detection in videos. Interfaces (GUI) 3(1):80–87
42. Li Y, Chang MC, Farid H, Lyu S (2018) In ictu oculi: exposing AI generated fake face videos by detecting eye blinking. arXiv preprint arXiv:1806.02877
43. Amerini I, Galteri L, Caldelli R, Del Bimbo A (2019) Deepfake video detection through optical flow based cnn. In: Proceedings of the IEEE/CVF international conference on computer vision workshops
44. Afchar D, Nozick V, Yamagishi J, Echizen I (2018) Mesonet: a compact facial video forgery detection network. In: 2018 IEEE international workshop on information forensics and security (WIFS), IEEE, pp 1–7
45. Nguyen HH, Fang F, Yamagishi J, Echizen I (2019) Multi-task learning for detecting and segmenting manipulated facial images and videos. arXiv preprint arXiv:1906.06876
46. Montserrat DM, Hao H, Yarlagadda SK, Baireddy S, Shao R, Horváth J, Bartusiak E, Yang J, Guera D, Zhu F, Delp EJ (2020) Deepfakes detection with automatic face weighting. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp 668–669
47. Wang R, Juefei-Xu F, Ma L, Xie X, Huang Y, Wang J, Liu Y (2019) Fakespotter: a simple yet robust baseline for spotting ai-synthesized fake faces. arXiv preprint arXiv:1909.06122
48. Nguyen HH, Yamagishi J, Echizen I (2019) Capsule-forensics: using capsule networks to detect forged images and videos. In: ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, pp 2307–2311
49. Korshunov P, Marcel S (2018) Speaker inconsistency detection in tampered video. In: 2018 26th European signal processing conference (EUSIPCO), IEEE, pp 2375–2379
50. Korshunov P, Halstead M, Castan D, Graciarena M, McLaren M, Burns B, Lawson A, Marcel S (2019) Tampered speaker inconsistency detection with phonetically aware audio-visual features. In: International conference on machine learning (No. CONF)
51. Agarwal S, Farid H, Fried O, Agrawala M (2020) Detecting deepfake videos from phoneme-viseme mismatches. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp 660–661
52. Li Y, Yang X, Sun P, Qi H, Lyu S (2020) Celeb-df: a large-scale challenging dataset for deepfake forensics. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 3207–3216
53. Iqbal F (2017) Detection of texture-less occluded objects by deformable part models. Doctoral dissertation, The University of Regina (Canada)
54. Hung BT (2021). Face recognition using hybrid HOG-CNN approach. In: Research in intelligent and computing in engineering, Springer, Singapore, pp 715–723
55. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 779–788
56. Chen W, Huang H, Peng S, Zhou C, Zhang C (2021) YOLO-face: a real-time face detector. Visual Comput 37(4):805–813
57. Kachouane M, Sahki S, Lakrouf M, Ouadah N (2012) HOG based fast human detection. In: 2012 24th international conference on microelectronics (ICM). IEEE, pp 1–4
58. Wang S, Han K, Jin J (2019) Review of image low-level feature extraction methods for content-based image retrieval. Sens Rev
59. Ruiz Sancho C (2014) Pedestrian detection using a boosted cascade of histogram of oriented gradients
60. Gong S, Bourennane EB (2019) A method based on texture feature and edge detection for people counting in a crowded area. In: Digital image and signal processing (DISP'19)
61. 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
62. Shibly MMA, Tisha TA, Tani TA, Ripon S (2021) Convolutional neural network-based ensemble methods to recognize Bangla handwritten character. PeerJ Comput Sci 7:e565
63. Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555
64. Shewalkar AN (2018) Comparison of rnn, lstm and gru on speech recognition data
65. He K, Sun J (2015) Convolutional neural networks at constrained time cost. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5353–5360
66. Lu L, Yang Y, Jiang Y, Ai H, Tu W (2018) Shallow convolutional neural networks for acoustic scene classification. Wuhan Univ J Nat Sci 23(2):178–184
67. Lei F, Liu X, Dai Q, Ling BWK (2020) Shallow convolutional neural network for image classification. SN Appl Sci 2(1):1–8
68. Redmon J, Farhadi A (2018) Yolov3: an incremental improvement. arXiv preprint arXiv:1804.02767.
69. Wei T, Tian Y, Chen CW (2020) Rethinking convolution: towards an optimal efficiency
70. Zhang R, Zhao R, Zhao X, Wu D, Zheng W, Feng X, Zhou F (2018) pyHIVE, a health-related image visualization and engineering system using Python. BMC Bioinform 19(1):1–6
71. Rotman M, Wolf L (2020) Shuffling recurrent neural networks. arXiv preprint arXiv:2007.07324
72. Lee MC (2022) Research on the feasibility of applying GRU and attention mechanism combined with technical indicators in stock trading strategies. Appl Sci 12(3):1007

73. Hajian-Tilaki K (2013) Receiver operating characteristic (ROC) curve analysis for medical diagnostic test evaluation. Caspian J Intern Med 4(2):627
74. Pezoulas V, Exarchos T, Fotiadis DI (2020) Medical data sharing, harmonization and analytics. Academic Press
75. Vujović ŽĐ (2021) Classification model evaluation metrics. IJACSA Int J Adv Comput Sci Appl 12:6
76. Achlioptas P (2019) Stochastic gradient descent in theory and practice
77. Ho Y, Wookey S (2019) The real-world-weight cross-entropy loss function: modeling the costs of mislabeling. IEEE Access 8:4806–4813