



A hierarchical feature selection strategy for deepfake video detection

Sk Mohiuddin¹ · Khalid Hassan Sheikh² · Samir Malakar¹ · Juan D. Velásquez^{3,4} · Ram Sarkar²

Received: 28 March 2022 / Accepted: 6 January 2023 / Published online: 6 February 2023
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2023

Abstract

Digital face manipulation has become a concern in the last few years due to its harmful impacts on society. It is especially concerning for high-profile celebrities because their identities can be easily manipulated using mobile or web applications such as FaceSwap and FaceApp. These manipulated faces are so close to real ones that it becomes really hard to detect them, even with bare eyes. Though deep learning-based models are predominantly used by researchers, they hardly check for the presence of irrelevant or redundant features produced by those models. To this end, we have proposed a hierarchical feature selection (HFS)-based method to detect deepfake images or videos. First, we have extracted both handcrafted and deep learning features from the inputs. Next, the HFS is applied to select a near optimal set of features. In each stage of it, a hybrid feature selection method is employed that integrates a population-based meta-heuristic model, called Grey Wolf Optimization, and a single solution-based meta-heuristic model, called the Vortex Search algorithm. We have evaluated our model on three publicly available datasets, namely Celeb-DF (V2), FaceForensics++, and Deepfake Detection Challenge (DFDC). The model provides 99.35%, 99.16%, and 85.67% AUC scores on the Celeb-DF (V2), FaceForensics++, and DFDC datasets, respectively, while using only 11.50%, 12.65%, and 10.22% of actual features. Besides, our model outperforms most of the state-of-the-art methods found in our literature review and evaluated on these three datasets.

Keywords Deepfake detection · Handcrafted features · Deep learning features · Hierarchical feature selection

1 Introduction

Digital face manipulation has become a real concern in social media as well as in our society due to its detrimental consequences. Although face manipulation, in its early stages, used to be done for comic purposes, later it became a cause for concern for celebrities and political figures. Also, in the past, due to technological limitations, it was very difficult to create realistic face images, and thus identification of the manipulations could be done with ease with bare eyes. However, recent developments in artificial intelligence (AI) based on deep neural networks, especially generative adversarial networks (GANs), let forgers alter the identity of a person with almost no perceivable signs. As a result, existing faces or face attributes such as eyes and mouth in a video or image are very intelligently replaced or altered by borrowing the corresponding parts from other subjects. Such a manipulated approach is known as deepfake, which generates very natural face images, and thus it becomes really challenging to detect the changes made by bare human eyes. Figure 1 depicts how a target face image is coupled with a source image to generate a fake face image. Therefore, designing a solution to the

✉ Juan D. Velásquez
jvelasqu@dii.uchile.cl

Sk Mohiuddin
myselfmohiuddin@gmail.com

Khalid Hassan Sheikh
skkhalidhassan7@gmail.com

Samir Malakar
malakarsamir@gmail.com

Ram Sarkar
ram.sarkar@jadavpuruniversity.in

¹ Department of Computer Science, Asutosh College, Kolkata, India

² Department of Computer Science and Engineering, Jadavpur University, Kolkata, India

³ Department of Industrial Engineering, University of Chile, Santiago, Chile

⁴ Instituto Sistemas Complejos de Ingeniería (ISCI), Santiago, Chile



Fig. 1 Some examples of how target images/videos are used to convert source images/videos into realistic fake faces

problem of identifying manipulated faces from real ones becomes very critical.

In the past, researchers tried to detect GAN-based forged faces by learning straightforward properties of images like abnormal changes within faces [1], and left-over convolutional traces in the spatial domain [2] made by GAN. Extending the same in the temporal domain, object behaviour, such as eye blinking patterns [3], was tracked for detecting deepfake videos. These methods performed well when the underlying deepfaking methods leave some explicit visual clues. However, with the advent of Convolutional Neural Network (CNN)-based deep learning models and the presence of voluminous training data, current GAN models can generate fake face images with ease which possess almost no visual clues (see Fig. 1). GAN-aided tools like FaceApp¹ and FaceSwap² let people manipulate face images in videos just by pressing a few buttons on their digital devices. As a result, we have observed most of these videos/images on several social media platforms now and then. Hence, it becomes a pressing need to restrict the spreading of such manipulated digital content. Most researchers from every corner of the world are trying to come up with a suitable solution to overcome this problem.

To date, several methods [4–6] have been developed to detect deepfake videos or images using different CNN architectures. These methods employed some basic CNN models to extract the desired features. The main limitation of using such a simple CNN model is that it sees the face differently than humans do when detecting forgery. Particularly, the vanilla CNN-based fake face detectors look for left-over artifacts from a limited facial region, as shown in Fig. 2, while humans usually seek these from the entire face image. In Fig. 2, we have shown heatmaps and grad-CAM images of fake face images that are generated by

fine-tuned XceptionNet [7] that is considered as the baseline model for most computer vision tasks.

To mimic human observation, most researchers [9–14] used several attention networks that can widen the search space observed by the vanilla CNN models. Although an attention mechanism helps researchers to replicate human understanding quite satisfactorily, it fails to perform a multi-view analysis of an object, i.e., checking the consistencies among different parts. Moreover, these attention models mostly employed convolution operations on the entire image to obtain the required attention map, which may highlight some unimportant regions. Besides, they demand more computational capacity than a vanilla CNN model and might overfit due to a lack of training data.

Apart from these typical issues related to deep learning models, another important issue that may arise is that the feature set used by the existing models may contain some irrelevant and/or unimportant features [15]. The presence of irrelevant features does not add any additional knowledge to the classifiers, but rather sometimes misleads the classifiers while performing classification tasks [16, 17]. As a result, the underlying classifier not only fails to generate the desired output but also takes a considerably longer training time [16, 18]. The use of a good feature selection (FS) method would help overcome the aforementioned limitation through eliminating non-important features.

In this paper, we have designed a deepfake video detection method that delves into the above research arguments. The proposed method combines hand-crafted features with deep learning features. Hand-crafted features are designed to check similarity among different parts of a face image, while deep learning features represent root level purity. For deep feature extraction, we have used fine-tuned XceptionNet [7] considering its success in face manipulation detection [4]. In the case of hand-crafted features, we have first extracted Daisy features [19] from image patches owing to their usefulness for several image classification problems [20]. We have then estimated the feature level similarity score among the patch-level Daisy features to mimic the multi-view analysis used in many pattern recognition problems [19, 21]. Next, to select a near optimal feature set, we have employed a hierarchical feature selection (HFS) technique considering its strength in selecting a better feature subset than using an FS at one go while diversified features are combined [22]. As a base FS model, we have designed a new hybrid FS technique that integrates a population-based meta-heuristic, called Grey Wolf Optimizer (GWO) [23], with a single solution-based FS technique, called vortex search (VS) algorithm [24] which have been termed hereafter as GWO–VS algorithm. We have fed only the selected final feature set into a two level multi-layer perceptron (MLP) to distinguish real images/videos from their fake counterparts.

¹ <https://apps.apple.com/gb/app/faceapp-ai-face-editor/id1180884341>.

² <https://github.com/MarekKowalski/FaceSwap>.



Fig. 2 Heatmap and Grad-CAM visualization [8] of some fake faces generated using fine-tuned XceptionNet model

Our main contributions are as follows:

- The development of a deepfake image/video detection technique that couples deep learning features and handcrafted features that represent similarity among different patches in an image using the HFS technique.
- The choice of an HFS technique for deepfake video or image detection over traditional non-HFS techniques for selecting a feature subset from a diverse set of features. To the best of our knowledge, this is the first time that an HFS technique has been used in this application.
- The combined use of GWO and VS methods (i.e., the GWO–VS algorithm) enhances the GWO algorithm's exploitation capacity, and thus, it is used as the fundamental FS scheme in the HFS technique.
- The evaluation of the HFS model on three publicly available datasets and its comparison with state-of-the-art methods. The proposed HFS model not only reduces more than 85% of features present in the original feature set but also improves the classification performance.

The remaining part of this article is organized as follows. Section 2 describes some previous methods. The proposed method is described in Sect. 3 while Sect. 4 is used for results and discussion. Finally, the paper is concluded in Sect. 5.

2 Related work

2.1 Deepfake detection techniques

A few works used handcrafted features. For example, McCloskey and Albright [1] analyzed colour level discrepancies between GAN generated manipulated face images and the corresponding actual camera captured images to design a face forensic method. Durall et al. [2] used Discrete Fourier Transform (DFT) to convert facial

spatial information into a 1D power spectrum, which was then fed to classifiers like the logistic regression classifier and the support vector machine (SVM) to distinguish fake videos from real ones. Despite their simplicity, these methods performed well when all three components: the source image, the target image, and the underlying faking algorithm were previously known. However, in reality, such information might not always be available, and even the modern synthesis image/video preparation methods do not expose the difference between real and synthesized images/videos that much. In such cases, the performance of these detection methods degrades. As a result, several methods tracked the discrepancies in visual artifacts in the temporal domain among the facial attributes like eyes, teeth, and the head position. Matern et al. [25], for example, used separate attributes like iris region, and teeth color to extract color features. This method performed well for the videos that have only frontal faces in most of the frames and the facial attribute under consideration is well identifiable. But all the videos do not possess such features and thus non-availability of certain attributes may lead to a false prediction. Yang et al. [26] extracted 68 facial landmarks from central face regions to estimate head poses. The hypothesis that they considered is the position of facial landmarks is shifted during tampering. However, their method is limited to detecting forgery whenever tampering takes place in the region outside of the landmark area.

In recent times, researchers mostly relied on deep learning architectures to come up with a feasible solution. Tolosana et al. [27] used the concept of transfer learning where they considered both the full-face and different face components like eyes, and nose as inputs to their model. Pre-trained Xception model and Capsule Network were used as the backbone CNN model. Rossler et al. [28] experimented exhaustively using several CNN-based deepfake detection methods on a self-made dataset which was later made public to the research community. They showed that XceptionNet [7] performed the best among others while trained and tested on different datasets setups. To focus on mesoscopic information in an image, a variant of the Inception module was used by Afchar et al. [29]. Two different CNN architectures viz., Meso-4 and MesoInception-4 were proposed by them to accomplish the forgery detection task. Amerini et al. [30] used the unusual motion of facial components for deepfake detection. In this work, a CNN-based model, called PWC-Net was used to extract optical flow matrices from the face images, which were then fed into VGG16-based semi-trainable model to filter out authentic videos.

In contrast to vanilla CNN-based methods, several works [10–13, 31] suggested integrating attention mechanism with the CNN model to focus on wider regions of the facial image. For example, Wang and Deng [10] detected

face forgeries by an attention-based method that guided the CNN aided detector to focus on the wider facial region for better detection. In this work, the authors utilized augmented data for better training of their designed attention model. Dang et al. [31] introduced another attention model to concentrate on the manipulated facial regions while extracting deep learning features. To exploit the local feature of the face region, an attention mechanism was applied by Chen et al. [11] that fused information presented in RGB and frequency domain that extends the similarity pattern of the regions. Zhao et al. [13] employed multiple attention models for deepfake detection while Miao et al. [12] utilized two attention models: one that guides the model to learn the forgery region (FR) and the other that enforces the model to pay less attention to identity semantics using a landmark guided dropout module (LM). They called their model FRLM. However, using attention mechanisms with CNN makes the models computationally expensive and may occasionally result in model overfitting. From a technical point of view, these methods tried to generate better feature maps, however, they did not investigate the presence of noisy or irrelevant features that might affect the end performance of the model. Apart from these, recently, two surveys [32, 33] reviewed various deepfake detection techniques.

2.2 FS techniques

In recent times, meta-heuristic algorithm-based FS techniques have gained enormous attention in the research community due to their wider applicability in diversified fields of computation and beyond. Three broad categories of FS methods [17, 34], namely filter, wrapper, and embedded are found in the literature. The main idea behind a filter-based FS approach is to rank features without using a classifier. Methods following filter-based approaches, in general, rely on statistical characteristics like correlation scores, and mutual information among the features to rank the features [17]. They are not only simple to implement but also computationally inexpensive. However, the main problem with the filter-based FS methods is that most filter methods evaluate features one by one without considering feature subsets.

On the contrary, a wrapper-based FS method first selects some subsets of features, known as candidate solutions, from the original feature set through different predefined norms and then evaluates the merits of the selected features based on an objective function. This is an NP-hard problem because determining the best feature subset from a set of N features has a complexity of 2^N [17, 35]. As a result, the researchers employ a variety of meta-heuristic methods to identify the potentially near-optimal feature subset [16, 35, 36]. Though these methods are computationally

expensive compared to filter methods, in general, they outperform the filter methods. An embedded method is aimed at reducing the computational time taken by wrapper-based methods by coupling the advantageous aspects of both filter and wrapper methods. In this case, the FS method is an integral part of a classification model. These methods, in general, fail to generate a low dimensional feature subset and are available with the classification methods. In short, each type of FS method has some benefits and limitations [35, 36]. Nowadays, FS methods based on meta-heuristic algorithms are a popular choice due to their advantages over the filter and embedded approaches. Meta-heuristic methods mostly followed one of two broad categories: population based or single solution based.

- *Evolutionary algorithms* Darwinian evaluation process inspired algorithms are evolutionary algorithms. The most commonly known algorithm in this category is Genetic Algorithm (GA) [37]. GA tries to mimic the biological process of crossover and mutation of genes. Some other Evolutionary algorithms are Bio-geography-based optimizer [38], and Differential Evolution [39].
- *Swarm-based algorithms* This category of algorithms is inspired by the behaviour of swarms, herds, and flocks of various creatures of nature. They try to mimic the swarm's behaviour to find optimum solutions. One of the pioneers of swarm-based algorithms is the particle swarm optimization (PSO) algorithm [40]. It mimics the social behaviour of a flock of birds. Some of the popularly used swarm-based algorithms are GWO [23], cuckoo search [41], and artificial bee colony algorithm [42].
- *Physics-based algorithms* Physics-based meta-heuristic algorithms are inspired by physical phenomena of nature. One of the popularly used physics-based meta-heuristic algorithms is simulated annealing (SA) [43]. SA is inspired by the metallurgical process of annealing. It is a single solution-based meta-heuristic. Other popularly used similar algorithms are harmony search (HS) [44], and gravitational search algorithm (GSA) [45].

In the single solution-based approach, a candidate solution tries to explore its nearby solutions over the iterations. It reduces the search locality to find an optimum solution. Some of the single solution-based algorithms are hill climbing [46], Tabu search [47], and VS algorithm. Often population-based meta-heuristics fail to attain the near optimal solution. This limitation could be overcome by using a population-based method coupled with a single solution-based algorithm. By doing so one can increase the exploration capabilities of the base meta-heuristic algorithm by a significant margin. In [48], a hybrid meta-

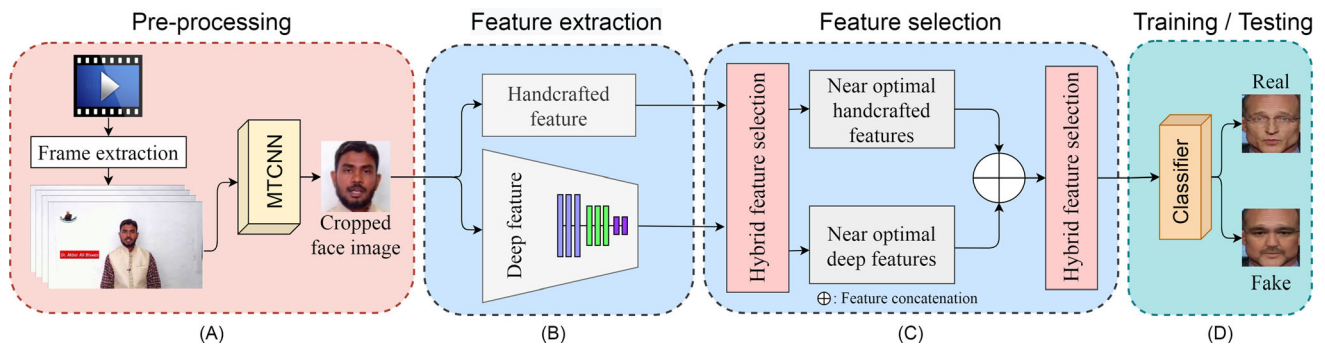


Fig. 3 Workflow of the proposed method having three stages **A** Preprocessing: it performs operations to generate cropped face image from the input video using the MTCNN-based face detection method, **B** Feature extraction: it extracts handcrafted and deep

features from the face image, **C** Feature selection: it selects a near optimal feature set by employing the HFS technique, and **D** Training/ Testing: it uses a classifier to classify input videos as forged or real

heuristic algorithm was proposed that used β -hill climbing to improve the exploration capability of the bat algorithm. In another work, Wang et al. [49] used the VS algorithm to overcome the problem of pre-mature convergence of the artificial bee colony optimization algorithm. Some more similar hybrid algorithms could be found in [34, 46, 49–51]. When different types of features are used for a classification problem, various authors employed an HFS approach in which the FS method was applied to combine multiple types of features at different levels. For example, the authors in [22] used various texture and shape descriptive features to recognize handwritten Bangla words.

3 Proposed work

In this work, we have proposed a method to distinguish face manipulated videos from real videos. At the outset, we have utilized I-frames of a questioned video (i.e., a test video) for further processing. The reason behind the consideration of I-frame over other categories of video frames is that it resides in uncompressed form and holds maximum color information compared to others. Hence, we have first extracted the I-frames from the questioned video and then passed it through the multi-task cascaded CNN (MTCNN) [52], a state-of-the-art face detection method, to extract the potential face region. Next, two types of features: handcrafted and deep learning features are extracted from these cropped face images. The features selected by employing the proposed HFS technique are fed to an MLP classifier for detection. In the HFS technique, we have used the GWO-VS technique. Figure 3 shows the steps followed in the proposed method.

3.1 Data processing

To detect the manipulated face in videos, I-frames are extracted using the ffmpeg³ technique. Since face manipulations are performed on face regions, at the beginning, we have tried the OpenCV Haar Cascade method to detect and crop the face regions from the extracted frames. However, this detection method suffers from false detection in many cases. Therefore, we have used a CNN-based face detection algorithm, called MTCNN, which performs well for different video qualities with diverse face alignments and sizes. After detecting the bounding box of each face region, we have increased the size of the bounding box by 10 pixels in each direction to include some background information around the face region. Finally, the cropped face images are resized to (225×225) and (299×299) for handcrafted and deep learning feature extraction purposes, respectively.

3.2 Feature extraction

In this work, we have considered both handcrafted and deep learning features described below.

3.2.1 Handcrafted feature extraction

To look at the face region following a multi-view approach [21], the cropped faces have been partitioned into 25 equal-sized image patches having dimension 45×45 . Next, we have extracted Daisy features [19] from each image patch. Daisy is a gradient orientation histogram-based image descriptor that describes both dense and shallow artifacts on the image patches. It is designed to extract dense properties of the images using a circularly symmetrical kernel and the Gaussian weight. The ‘scikit-image’ library is used to extract a bag of 25 Daisy descriptors each having

³ <https://ffmpeg.org/>.

104 elements from a single image patch. This way a single patch is represented by a 2600-dimensional feature vector.

Next, we have estimated feature similarity using Euclidean distance (see Eq. 1) among all the extracted patches. In Eq. 1, E_{ij} represents the Euclidean distance between i th and j th patches while \vec{P}_i and \vec{P}_j denote feature vectors for the i th and j th patches, respectively. For every value of $i \in \{1, 2, \dots, 24\}$, we have varied the value of j from $i + 1$ to 25 (i.e., $j \in \{i + 1, i + 2, \dots, 25\}$) to extract similarity among all patches. Finally, concatenating all E_{ij} s, a 300-dimensional feature vector is obtained from a single face image. Figure 4 depicts the feature extraction process pictorially.

$$E_{ij} = |\vec{P}_i - \vec{P}_j| = \sqrt{(P_i - P_{i+1})^2 + (P_i - P_{i+2})^2 \dots + (P_i - P_{25})^2} \quad (1)$$

3.2.2 Deep feature extraction

The proposed method uses fine-tuned XceptionNet [7] network to learn the important features from face images. The architecture of the Xception model is based on depth-wise separable convolutional layers that are inspired by the Inception model. Each channel uses 1×1 depth-wise convolutions to capture the cross-channel correlations. During the depth-wise separable convolutions, the Xception network performs channel-wise convolutions first and then employs 1×1 convolution while the Inception network uses 1×1 convolution first and then uses channel-wise convolutions. In addition to this, the number of layers is reduced from 159 layers in Inception V3 to 126 layers in the Xception architecture. The fine-tuned Xception model extracts 2048-dimensional deep learning features from a face image of size 299×299 .

3.3 Feature selection

In this work, we have proposed an HFS model to perform FS. A hybrid FS method is designed using the advantageous aspects of GWO [23] and VS algorithm [24]. Here we have described all the components of the proposed HFS model.

3.4 Grey wolf optimizer

GWO is inspired by the preying strategies of a pack of grey wolves. One of the interesting characteristics of grey wolves is that they form packs with a well-defined hierarchical structure. A typical pack of grey wolves consists of 5–12 grey wolves on average. The leader of the pack is known as the alpha who is the main decision maker.

Instructions of the alpha wolf are followed by the remaining wolves of the pack. Despite the dominance of the alpha wolf in the decision making process, the pack also shows traits of democratic behaviour as the alpha wolf takes advice from others. The second most important type of wolf in the hierarchy is beta wolves who instruct other wolves in the pack on behalf of the alpha and provide feedback to the alpha. They are also the most suitable candidate for being alpha. The third most important type of wolf is the delta, and the lowest ranked wolves are called the omega. Delta has an upper hand over omega, although they submit to the command of alpha and beta. One of the other interesting factors of a grey wolf pack is their hunting behaviour. The main phases of their hunting are as follows:

- Tracking, chasing, and approaching the prey.
- Pursuing, encircling, and harassing the prey until it stops moving.
- Attacking the prey.

For FS, a solution can be represented in the following way $\vec{F} = [f_0, f_1, \dots, f_{N-1}]$, where N and f_i ($i = 0, 1, \dots, N - 1$) represent the number of features and i th feature in the feature set, respectively. Here, any f_i can take a value of 0 or 1 to indicate non-inclusion or inclusion of i th feature in the final feature subset. Suppose a position represented by any candidate solution at a particular iteration of a wolf at some instance is \vec{W} . The position of α , β , and δ wolves are \vec{W}_α , \vec{W}_β , and \vec{W}_δ , respectively. The mathematical model with which the wolves in the pack update their position is shown in Eqs. 2 and 3.

$$\vec{D}_\alpha = k\vec{1} \cdot \vec{W}_\alpha - \vec{W}, \vec{D}_\beta = k\vec{2} \cdot \vec{W}_\beta - \vec{W}, \vec{D}_\delta = k\vec{3} \cdot \vec{W}_\delta - \vec{W} \quad (2)$$

$$\vec{W}_1 = \vec{W}_\alpha - \vec{A}_1 \cdot \vec{D}_\alpha, \vec{W}_2 = \vec{W}_\beta - \vec{A}_2 \cdot \vec{D}_\beta, \vec{W}_3 = \vec{W}_\delta - \vec{A}_3 \cdot \vec{D}_\delta \quad (3)$$

The parameters: A_i and k_i ($i \in \{1, 2, 3\}$) in Eqs. 2 and 3 are defined by Eq. 4.

$$A_i = 2 \cdot \vec{a} \cdot \vec{r}_i^1 - \vec{a}, k_i = 2 \cdot \vec{r}_i^2 \quad (4)$$

In Eq. 4, \vec{r}_i^1 and \vec{r}_i^2 are two random vectors and components of \vec{a} are linearly decreased over the iteration from 2 to 0. It is to be noted here that if $\vec{X} = [X_0, X_1, \dots, X_{N-1}]$ and $\vec{Y} = [Y_0, Y_1, \dots, Y_{N-1}]$ are two candidate solutions or N -dimensional vectors then $\vec{X} \cdot \vec{Y}$ is defined by Eq. 5.

$$\vec{X} \cdot \vec{Y} = [X_0 \cdot Y_0, X_1 \cdot Y_1, \dots, X_{N-1} \cdot Y_{N-1}] \quad (5)$$

Suppose \vec{W} can be represented as $[w^1, w^2, \dots, w^{N-1}]$. The possible position of the wolf in the next iteration is defined by the Eq. 6.

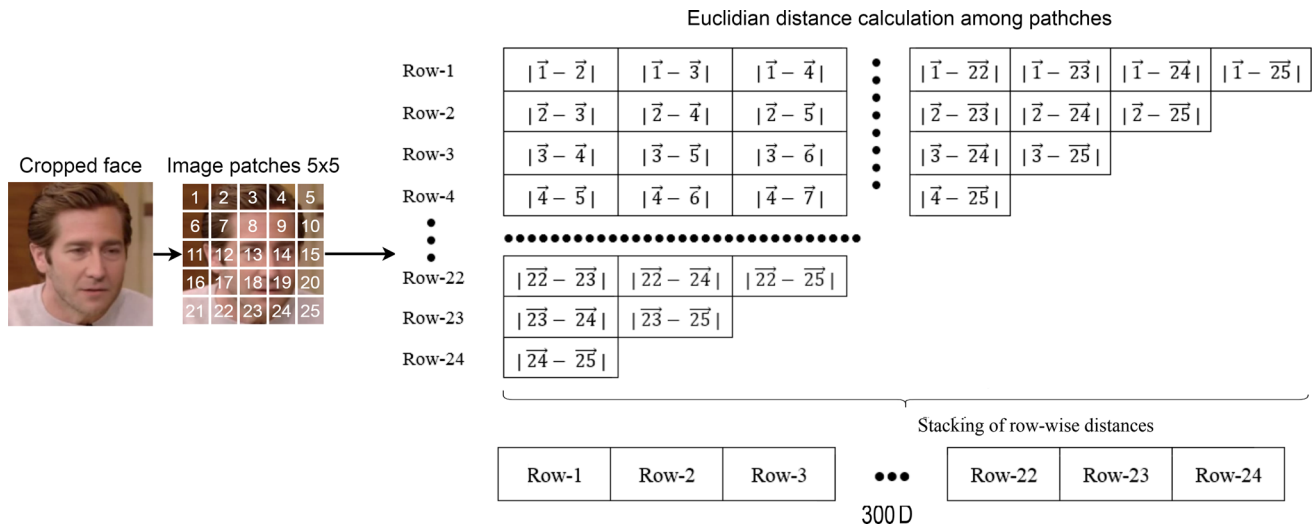


Fig. 4 Illustration of the hand-crafted feature extraction process from a cropped face image. The second image from the left shows the label number $t \in \{1, 2, \dots, n\}$ of the extracted patches. Next, from each patch, we have extracted Daisy features represented by \vec{t} which are then used to find similarities among the different patches. The pairwise norm-2 distances (i.e., $|\vec{t} - \vec{j}|$, where $i \in \{1, 2, \dots, n-1\}$ and

$j \in \{i+1, i+2, \dots, n\}$) are then calculated and stored in list named as Row- i . Thus, the length of Row- i is $n-i$. Finally, all these lists are stacked to form the final feature vector. The length of feature vector is $= (n-1) + (n-2) + \dots + 1 = \frac{n(n-1)}{2}$. In our case $n = 25$ and thus we have obtained a feature vector of length 300

$$\vec{W}(t+1) = \text{Func}\left(\frac{\vec{W}_1 + \vec{W}_2 + \vec{W}_3}{3}\right) \quad (6)$$

In Eq. 6, $\text{Func}()$ is a transfer function that converts a continuous value into a discrete binary value. Next, the fitness values of the newly generated position of a wolf (i.e., the new candidate solution) and its old position are calculated. If the fitness value of the new position is better than the previous one, we have updated the position of the wolf.

3.5 Vortex search algorithm

VS algorithm was proposed by Doğan and Ölmez [24], which is a single solution-based meta-heuristic algorithm. Suppose $S(\vec{V})$ is the set of candidate solutions generated from the current solution \vec{V} . In the next phase i.e., in the replacement phase, a suitable solution (say, $\vec{V}' \in S(\vec{V})$) is selected from the set of generated candidate solutions that may replace the current solution. This process continues until the termination condition is met. The phases of the VS algorithm are as follows:

1. Generating the initial solution.
2. Generating the set of candidate solutions.
3. Ensuring all candidate solutions lie in the search space.
4. Replacing the current solution with the best candidate solution if any.
5. Decreasing the radius of search space.

An initial solution (say, \vec{V}) is determined by Eqs. 7 and 8.

$$\vec{V} = [V_0, V_1, \dots, V_{N-1}] \quad (7)$$

$$V_i = \frac{u\text{Lim}_i + l\text{Lim}_i}{2} \quad (8)$$

In Eq. 8, $u\text{Lim}_i$ and $l\text{Lim}_i$ are the maximum and minimum possible values in the i th dimension, respectively. Please note that $i \in \{0, 1, 2, \dots, N-1\}$ where N is the dimension of the single solution which is the cardinality of the feature set. Next, a set of neighbour solutions (say, $S(\vec{V})$) is generated randomly using the Gaussian distribution. The multivariate Gaussian distribution can be formulated using Eq. 9.

$$p(\vec{V}|\vec{x}, \Sigma) = \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} \exp\left\{-\frac{1}{2}(\vec{x} - \vec{V})^T \Sigma^{-1}(\vec{x} - \vec{V})\right\} \quad (9)$$

Here \vec{x} is a randomly generated N dimensional vector. For spherical distribution, the Σ can be defined using Eq. 10.

$$\Sigma = \sigma^2 \cdot [I]_{N \times N} \quad (10)$$

Here $[I]_{N \times N}$ is an identity matrix of order $N \times N$, and σ^2 represents the variance of the distribution. The initial standard deviation is calculated using eq. 11.

$$\sigma_0 = \frac{\max(u\text{Lim}_0) - \min(l\text{Lim}_0)}{2} \quad (11)$$

σ_0 can be considered radius of distribution. The initial phases of this algorithm need weak locality for better exploration ability; hence the maximum possible radius of

distribution is considered here. Before the replacement of the center of distribution that is the current single solution, all candidate solutions that lie beyond the search space must be adjusted into the search space following Eq. 12.

$$v'_i = \begin{cases} v'_i, & \text{if } l\text{Lim}_i \leq v'_i \leq u\text{Lim}_i \\ \text{rand}(u\text{Lim}_i - l\text{Lim}_i) + l\text{Lim}_i, & \text{else} \end{cases} \quad (12)$$

In Eq. 12, v' is any of the random candidate solutions. In the next step, if the best candidate solution outperforms the existing solution based on fitness score, the existing solution is modified to the best candidate solution. Afterward, this algorithm gradually decreases the radius of the search space by reducing the standard deviation (SD) following Eqs. 13 and 14.

$$\sigma_t = \sigma_0 \cdot \text{gammmaincinv}(x, \alpha_t) \quad (13)$$

$$\alpha_t = \alpha_0 - \frac{\text{iter}}{\text{TotalIter}} \quad (14)$$

In Eq. 13, $\text{gammmaincinv}()$ is the incomplete inverse gamma function [53]. In summary, VS starts with a solution first and then generates a set of candidate solutions using the Gaussian distribution defined in Eq. 9 keeping the solution as the center. Next, using steps 3–5 of the VS algorithm mentioned earlier, the VS algorithm tries to find a better solution.

3.6 Proposed hybrid FS method

The GWO, a population-based meta-heuristic algorithm, suffers from premature convergence like other similar algorithms. Inspired by the work by Wang et al. [49], here we have used a single solution-based meta-heuristic, called VS algorithm to deal with premature convergence of GWO. In the proposed hybrid FS method (i.e., GWO–VS), the VS algorithm helps in finding a better position for the wolves at each iteration. In other words, the VS algorithm, used in the GWO–VS algorithm, starts with the position of a wolf i.e., a candidate solution (say, \vec{W}_i , where $i = 0, 1, \dots, P_G - 1$, and P_G is the number of wolves in a pack) from the GWO algorithm as the initial solution and then follows the described steps to find a better position for the wolf under consideration. The entire process of the GWO–VS-based FS technique is described through Eqs. 15, 16, and 17.

$$\text{Set}(\vec{W}_i) = \{\vec{S}_0, \vec{S}_1, \dots, \vec{S}_{P_V-1}\} \quad (15)$$

In Eq. 15, P_V is the number of candidate solutions of VS algorithm that consider the i th candidate solution (i.e., $\vec{W}_i(t)$, where $i = 0, 1, \dots, P_G - 1$) of the GWO algorithm

as the initial solution. It is to be noted here that $\vec{S}_k, k = 0, 1, \dots, P_V - 1$ is a vector in N -dimensional feature space.

$$\vec{W}_i(t+1) = \text{Best}(\text{Set}(\vec{W}_i(t))) \quad (16)$$

Equation 16 describes the selection of the best candidate solution for the $(t+1)$ th iteration of VS algorithm. Here, $t = 1, 2, \dots, j_{\max}$ and j_{\max} is the number of iterations for which VS algorithm runs.

$$p(\vec{w}|\vec{x}, \Sigma) = \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} \exp \left\{ -\frac{1}{2} (\vec{x} - \vec{w})^T \Sigma^{-1} (\vec{x} - \vec{w}) \right\} \quad (17)$$

The VS algorithm uses a multivariate Gaussian distribution (see Eq. 17) to generate candidate solutions. Here w is the current solution or center of the search space, and x is a random N dimensional vector. Σ is the covariance matrix. The value of Σ can be calculated using Eqs. 10 and 11. Over several iterations, the scope of search space is similarly reduced using Eqs. 13 and 14. The flowchart of the GWO–VS algorithm is shown in Fig. 5. The time complexity of the GWO–VS algorithm is approximately $O(P_G * P_V * \text{iter}_{\max} * j_{\max})$, where $P_G, P_V, \text{iter}_{\max}, j_{\max}$ represent population size of GWO, the population size of VS, the maximum iteration number of GWO, and the maximum iteration number of VS, respectively. It is to be noted here that whenever we have used the operator '*' it means the arithmetic multiplication operator. A detailed complexity analysis of the GWO–VS algorithm is provided in Section A of the supplementary document. We have also added the pseudocode of this algorithm in Section B of the supplementary document.

3.6.1 Fitness function

A fitness function is used to evaluate the performance of a candidate solution. In our work, the fitness function is designed to deal with a trade-off between the number of selected features (N_{selected}) and the classification accuracy (ACC), which is defined in Eq. 18. In this work, ACC is evaluated on a validation dataset and an MLP is used as a classifier.

$$\text{fitness}(\vec{w}) = wt * \text{ACC} + \frac{N_{\text{selected}}}{N} * (1 - wt) \quad (18)$$

In Eq. 18, $wt \in [0, 1]$ (in our case, $wt = 0.90$) and $(1 - wt)$ represent the weights for the recognition accuracy and the number of selected features, respectively.

3.6.2 Transfer function

As FS is a binary optimization problem, the solution vector can only contain '0' or '1', where '0' and '1' mean the exclusion and inclusion of the corresponding feature into

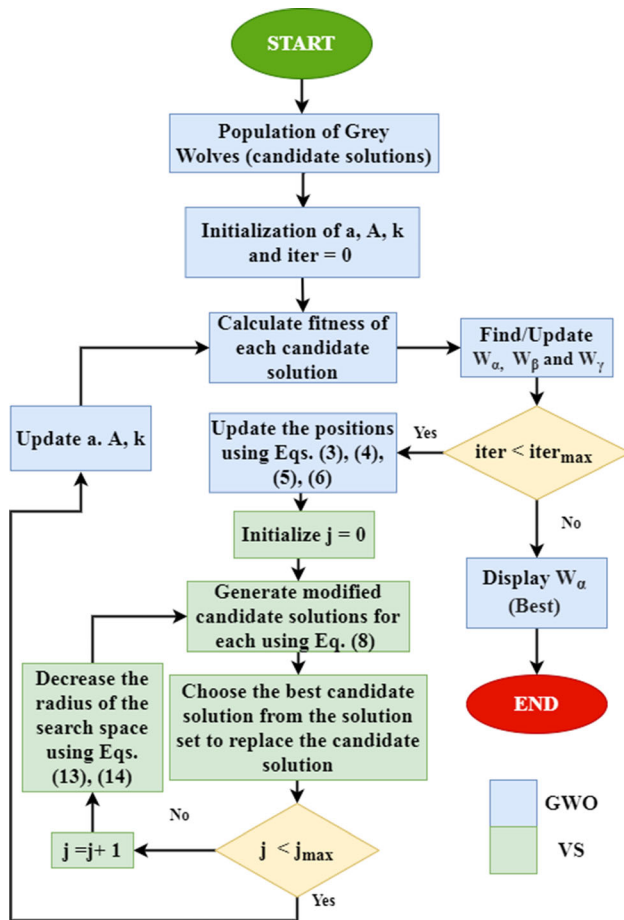


Fig. 5 Flowchart of the proposed hybrid FS technique

the selected feature set from the actual feature set, respectively. However, for GWO the value cannot be binary due to the nature of mathematical operations. To ensure that the output always stays within the expected range, we need to apply a transfer function on each candidate solution. This task is performed by the sigmoid (S-shaped) transfer function. The S-shaped transfer function is formulated using Eq. 19.

$$T(x) = \frac{1}{1 + e^{-x}} \quad (19)$$

$$X^d(t) = \begin{cases} 1, & \text{if } \text{rnd} < T(X^d(t)) \\ 0, & \text{if } \text{rnd} \geq T(X^d(t)) \end{cases} \quad (20)$$

In Eq. 20, $\text{rnd} \in (0, 1)$ is a random number.

3.7 Hierarchical feature selection

In the HFS model, multiple types of features are combined to come up with a near optimal feature subset to perform the said classification task. The combination and selection of features are carried out hierarchically. In the HFS model, firstly we have employed our proposed hybrid FS technique on

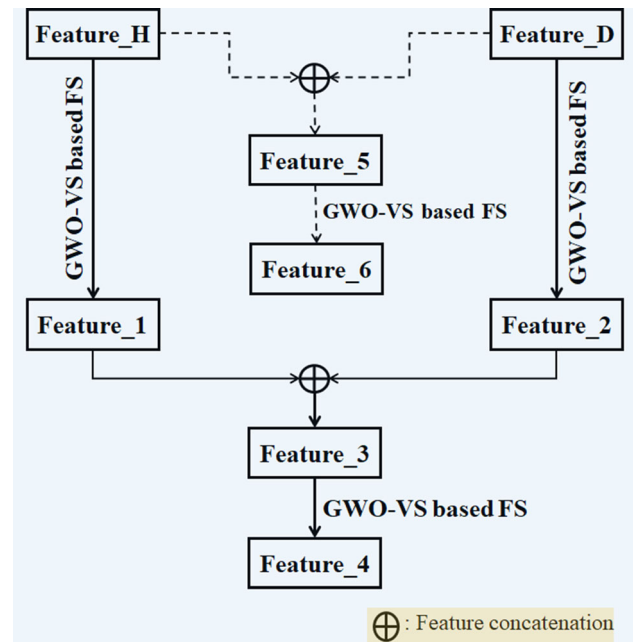


Fig. 6 In the proposed HFS model, steps followed to generate the feature vectors are connected using solid-lined arrows, while steps for the non-HFS stages are connected using dashed-lined arrows. Feature_4 and Feature_6 are the obtained near optimal features following HFS and non-HFS approaches, respectively

extracted handcrafted (Feature_H in Fig. 6) and deep learning features (Feature_D in Fig. 6) individually. This is done to remove the redundant features from each type of feature and we have obtained the near optimal feature sets: Feature_1 and Feature_2 from Feature_H and Feature_D, respectively, as shown in Fig. 6. Next, we have concatenated the obtained near optimal feature subsets and obtain a new feature subset (Feature_3 in Fig. 6). This combined feature set (i.e., Feature_3) might possess some irrelevant features [22]. Thus, at this end, the proposed hybrid FS model is again employed on the combined feature set and we have obtained the final feature subset (i.e., Feature_4 in Fig. 6). This feature set is considered for classification. Figure 6 shows the commonly used FS strategy, where features are concatenated first and then the combined features (i.e., Feature_5 in Fig. 6) are passed through the FS method to generate the near optimal features (Feature_6 in Fig. 6).

4 Results and discussion

In this work, we have designed an HFS technique to detect whether a questioned video/image has been manipulated by some deepfake techniques or not. The proposed model is evaluated on three popular video datasets—Celeb-DF (V2) [4] termed hereafter as CeDF, FaceForensics++ (FF++) [28] and Deepfake Detection Challenge (DFDC) [54].

Table 1 Distribution of videos and extracted frames/cropped faces used in training, validation, and test sets for both datasets

Dataset	# Videos in						# Frames/cropped faces in					
	Train set		Validation set		Test set		Train set		Validation set		Test set	
	Real	Fake	Real	Fake	Real	Fake	Real	Fake	Real	Fake	Real	Fake
CeDF	612	4399	100	900	178	340	1130	8022	100	900	178	340
FF++	700	700	200	200	100	100	2994	2876	200	200	100	100

4.1 Dataset preparation

To assess the performance of the proposed deepfake detection technique, we have used three prevalent benchmark datasets: CeDF [4], FF++ [28], and DFDC [54]. The CeDF contains 5639 high quality synthesis videos of different celebrities. All the synthesized videos are generated from 590 original videos collected from YouTube with different subjects like ages, genders, and backgrounds. It provides 518 videos (178 real and 340 fake) for assessing the performance of a new model. The FF++ video dataset contains 4 types of forged videos, namely Deepfakes, Face2Face, FaceSwap, and NeuralTextures. Each category contains 1000 videos generated from 1000 pristine videos. The whole video dataset is available in two different compression rates but here we have used the c23 version. Since the proposed method targets detecting deepfake videos thus in our experiments, we have considered only Deepfake category videos as fake videos and pristine category videos as real videos. At the outset of data preparation, we have divided the datasets into the train, test, and validation sets. Table 1 shows the breakup details for both datasets. The Sect. 4.8 presents details about dataset division and performance on the DFDC dataset.

Next, we have extracted frames from the train, test, and validation video sets. In the case of training video sets, equidistant frames in the temporal domain are chosen while for test and validation sets we have considered only the first I-frame from each video. Such choice of the first I-frame is inspired by the methods reported in [6, 14]. During the generation of cropped face images using MTCNN, we have considered only one face having the highest confidence score from each of the extracted frames. Table 1 shows the distribution frames/cropped face images extracted from both the video datasets.

4.2 Experimental setup

The Daisy features are extracted using ‘scikit-image’ library where the parameters are set as: radius = 13, rings = 2, histograms = 6 and orientations = 8. For fine-tuning XceptionNet, we have set learning rate = 0.0001, batch size = 32, number of output layers = 2, number of epochs = 50, steps per epoch = 40 and

optimizer = Adam. For the implementation of the GWO–VS model, we have used the parameter values as suggested in [23] and [24] for the GWO algorithm and VS algorithm, respectively. However, for other meta-heuristic algorithms used here for comparison, we have considered the parameter values as used in [46].

4.3 Experimental results

In this section, we have described the performance of the proposed HFS-based deepfake detection method. The performance evaluation of the proposed HFS method considers three parameters namely, Area Under the ROC Curve (AUC) score, test accuracy, and the number of selected features. We have executed the proposed HFS model 10 times on both datasets to test the effect of the randomness of meta-heuristic results. The results are shown in Fig. 7. From the results is clear that in most of the cases, we have obtained 98.00% and 97.30% test accuracy on FF++ and CeDF datasets, respectively. The detailed experimental outcomes are shown in Tables 2 and 3 for CeDF and FF++ datasets, respectively. These results help us to understand the performance of the entire model as well as all its components (see Fig. 6). We can observe from Table 2 that test accuracy and AUC score obtained on CeDF using Feature_4 (obtained with the proposed HFS method) are 97.30% and 99.35%, respectively, which are the best performances among all the other features. Likewise, the HFS model provides the best result in terms of both test accuracy and AUC score on FF++ dataset (see Table 3). In this case, the test accuracy and AUC score are 98.00% and 99.16%, respectively. By closely analyzing the results in Tables 2 and 3, the following statements can be inferred:

- If we have compared the performance of Feature_D and Feature_H then it is found that Feature_D is dominating. However, by concatenating these two (i.e., Feature_5 in Fig. 6), an improved performance is obtained. From these results, we can safely comment that the use of handcrafted features representing multi-view analysis along with deep learning features is beneficial.
- To test the usefulness of Feature_H in solving the current problem we have performed another set of experiments with the proposed model by using the deep

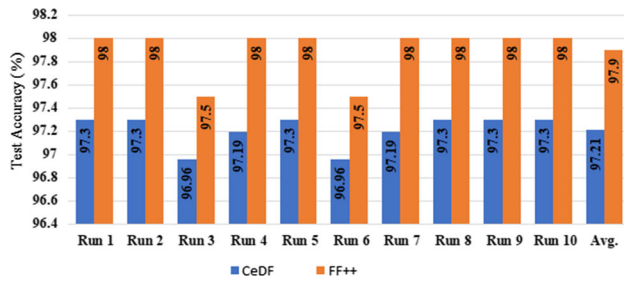


Fig. 7 Test accuracies of the proposed HFS-based deepfake detection method over ten independent runs

learning features extracted from MesoNet [29]. The results are shown in Table C.1 of Section C of the

supplementary document confirming the usefulness of Feature_H in the context of the current problem.

- Almost in all the cases i.e., whenever the proposed hybrid FS (i.e., GWO–VS) is employed, we have observe improved performance and reduced feature dimension. This proves the usefulness of the proposed hybrid FS technique.
- Most of the methods, irrespective of the applications where they were used, generated the final feature vector by concatenating the features i.e., Feature_5 in Fig. 6 to perform the classification task. On both datasets, the use of the HFS to perform FS aided classification proves to be beneficial over this traditional approach. Proposed HFS not only reduces the feature dimension by 88.50%

Table 2 Performance of the proposed HFS model and its components (see Fig. 6) on CeDF dataset

Features	Performance metrics											
	Test accuracy (in %)				AUC score (in %)				# Selected features			
	Max	Min	Avg.	SD	Max	Min	Avg.	SD	Max	Min	Avg.	SD
Feature_H	65.64	64.76	65.16	0.299	73.49	73.07	73.32	0.157	300	300	300	0.00
Feature_D	95.37	94.68	95.01	0.249	98.88	97.92	98.51	0.394	2048	2048	2048	0.00
Feature_1	65.83	65.02	65.56	0.253	71.60	70.95	71.29	0.252	91	76	81.67	4.99
Feature_2	95.14	94.78	95.05	0.129	99.02	98.27	98.80	0.271	722	703	712	6.40
Feature_3	96.91	96.33	96.65	0.211	98.14	97.60	97.96	0.180	813	780	795.5	11.43
Feature_4	97.30	96.83	97.19	0.181	99.35	98.26	98.92	0.405	270	240	257	10.55
Feature_5	96.72	95.75	96.38	0.343	98.76	98.00	98.38	0.310	2348	2348	2348	0.00
Feature_6	96.91	95.95	96.45	0.374	98.14	98.10	98.21	0.136	830	790	812.33	14.53

Here, performances under Feature_4 and Feature_6 represent performances of HFS and non-HFS-based FS, respectively. Here Max, Min, Avg, and SD stand for maximum, minimum, average, and standard deviation, respectively

Table 3 Performance of the proposed HFS model and its components (see Fig. 6) on the FF++ dataset

Features	Performance metrics											
	Test accuracy (in %)				AUC score (in %)				# Selected features			
	Max	Min	Avg.	SD	Max	Min	Avg.	SD	Max	Min	Avg.	SD
Feature_H	63.5	62.5	62.90	0.374	70.22	69.84	70.04	0.148	300	300	300	0.00
Feature_D	96.00	95.50	95.70	0.245	98.34	97.8	98.13	0.211	2048	2048	2048	0.00
Feature_1	71.00	70.00	70.55	0.350	76.22	75.56	75.96	0.264	97	90	95	2.89
Feature_2	96.50	95.50	96.10	0.374	97.83	96.78	97.48	0.361	659	640	643.33	11.47
Feature_3	97.50	96.50	97.25	0.335	98.68	98.10	98.46	0.181	756	720	737.17	14.63
Feature_4	98.00	97.50	97.70	0.245	99.16	98.33	98.89	0.273	247	219	234	10.52
Feature_5	97.00	96.00	96.55	0.350	98.61	97.75	98.24	0.340	2348	2348	2348	0.00
Feature_6	97.00	96.50	96.80	0.245	98.68	98.20	98.44	0.162	714	690	703.83	9.39

Here, performances under Feature_4 and Feature_6 represent performances of HFS and non-HFS-based FS, respectively. Here Max, Min, Avg, and SD stand for maximum, minimum, average, and standard deviation, respectively

and 87.35% of the actual feature dimension for CeDF and FF++ datasets, respectively, but also improves the end performance (improvements on CeDF: test accuracy = 0.58% and AUC score = 0.59% and FF++: test accuracy = 1.00% and AUC score = 0.55%) over the traditional approach.

- In general, the methods that used FS to perform some classification tasks followed a non-HFS approach i.e., the authors generated Feature_6 as shown in Fig. 6. On both datasets, the use of the HFS to perform the FS task proves to be beneficial over this non-HFS FS approach. Proposed HFS not only selects 3.07 and 2.71 times fewer features to the dimension of Feature_6 for CeDF and FF++ datasets, respectively, but also improves the end performance (improvements on CeDF: test accuracy = 0.39% and AUC score = 0.36% and FF++: test accuracy = 1.00% and AUC score = 0.48%) over the single-stage FS approach.
- If we compare the performances obtained using Feature_3 and Feature_4 (see Fig. 6) then we can observe very little performance improvement for both datasets. However, a notable reduction in feature dimension can be observed. The dimension of Feature_4 is 3.01 and 3.06 times less concerning Feature_3 for CeDF and FF++ datasets, respectively. From these results, we can safely comment that the use of the proposed hybrid FS on Feature_3 is also beneficial while considering the dimension of the final feature set and the training cost of the model (see Fig. 8). Figure 8 also shows that the model with reduced features is trained almost 2.5 times faster than the one.

4.3.1 Statistical test among different near optimal features

We have also performed the statistical significance test to ensure the truthiness of the aforementioned observations. The objective of this statistical test is to test whether there

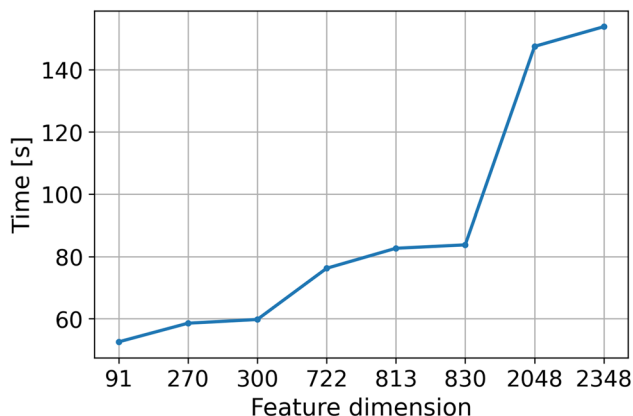


Fig. 8 Training time versus number of features for the CeDF dataset

is enough evidence to reject the null hypothesis: the performance of a near optimal feature subset (i.e., Feature_{*i*}, where $i = \{1, 2, 3, 5, 6\}$) over 10 independent trials is not significantly different from the performance of Feature₄ while performing deepfake detection. To do this, we have computed the p value using the Wilcoxon rank-sum test following the approach used in [55]. We reject the null hypothesis if the estimated result is less than 0.05 (i.e., < 5% of the total). We have executed the FS stages/techniques that generate near optimal feature subsets: Feature₁, Feature₂, ..., Feature₆ 10 times independently and recorded the performances in terms of AUC score, test accuracy, and the number of selected features. Based upon these records, we have estimated the p value, and the estimated p values are listed in Table 4. From the results recorded in Table 4, we can reject the null hypothesis at a 5% confidence level. Therefore, we can safely claim that the performance of the deepfake detection technique using Feature₄ is better than the performance of other cases i.e., performing deepfake detection using other selected near optimal feature subsets (i.e., Feature_{*i*}, where $i = \{1, 2, 3, 5, 6\}$) at the 5% significance level.

4.4 Generalization capability of selected features

In this section, we have performed another set of experiments to estimate the generalization capability of the proposed HFS model. For this set of experiments, first, we have employed the HFS technique on the first dataset to obtain the indices of the selected features, and then using these selected feature indices, the near-optimal features are extracted from all samples of the second dataset to perform the classification task. For example, we have extracted the selected features obtained by employing HFS on the CeDF dataset (see Table 2) from the FF++ dataset using indices of selected features and performed the classification using those selected features only. The results are shown in Table 5. Following a similar approach, the classification performances on the CeDF dataset are recorded in Table 5. These results signify that the features selected using the proposed HFS are more effective in comparison to the actual features at least on these datasets used here.

4.5 Generalization capability of the proposed HFS method

We have conducted a set of experiments to test the generalization capability of the proposed HFS method. For this, we have conducted experiments in a cross-dataset setup used in [4, 6, 14]. The results are recorded in Table 6 along with some state-of-the-art works. In Table 6, the term CE_FF represents the experiment when the proposed

Table 4 p values of the Wilcoxon rank-sum test. Here Fet_1 to Fet_6 stands for Feature_1 to Feature_6. Also, Acc, AUC, and #Feature represent test accuracy, AUC score, and the number of features selected, respectively

	Dataset	Parameter	Fet_1	Fet_2	Fet_3	Fet_5	Fet_6
Fet_4	CeDF	Acc	0.000079	0.000079	0.000079	0.000079	0.000079
		AUC	0.000079	0.000334	0.034821	0.048152	0.017147
		#Feature	0.000079	0.000079	0.000079	0.000079	0.000079
	FF++	Acc	0.000079	0.001248	0.002036	0.001101	0.001600
		AUC	0.000079	0.000143	0.029309	0.001599	0.002035
		#Feature	0.000079	0.000079	0.000079	0.000079	0.000079

Table 5 Performances when HFS is employed on one dataset to obtain the near optimal feature subset and the indices of selected features present in the selected feature subset, and then only the selected features are extracted using these feature indices from another dataset for performing the classification task

Parameter	Feature_D	Feature_H	Feature_5	Feature_6	Feature_4
<i>Case 1</i>					
Test accuracy (in %)	95.37	65.64	96.72	96.72	97.30
AUC score (in %)	98.88	73.49	99.35	98.99	99.64
Number of features	2048	300	2348	804	297
<i>Case 2</i>					
Test accuracy (in %)	96.00	63.50	97.00	96.50	97.50
AUC score (in %)	98.34	70.22	97.83	98.68	99.22
Number of features	2048	300	2348	830	270

Here Case 1 represents the scenario when HFS employed on the FF++ to get indices of the features present in the selected feature subset and classification performed using only the selected features extracted from CeDF while Case 2 represents the opposite scenario

HFS-based deepfake detection model is trained on CeDF and tested on the FF++ dataset, while FF_CE represents the reverse case. It is worth mentioning here that the performances of the state-of-the-art methods are evaluated on our setups. From the results, we can safely conclude that the proposed method works well in the cross-dataset setup in comparison with state-of-the-art methods.

4.6 Comparison with state-of-the-art FS methods

In this work, we have designed a hybrid FS technique termed as GWO–VS method. All the experiments were conducted to compare the performance of the proposed hybrid FS technique with other state-of-the-art FS techniques. For the comparative purpose, we have considered five popularly used meta-heuristic methods: GA, PSO, HS, WOA, and GWO. For simplicity, we have concatenated

Table 6 Performance of the proposed model on a cross-dataset experimental setup

Exp.	Methods	Test accuracy (in %)	AUC score (in %)
CE_FF	Li et al. [4]	64.50	75.19
	Afchar et al. [29]	50.50	51.51
	Qian et al. [56]	54.50	54.04
	Mohiuddin et al. [6]	60.00	59.93
	Ganguly et al. [14]	65.00	63.80
	Proposed	66.50	76.72
	Li et al. [4]	58.06	55.60
	Afchar et al. [29]	66.41	65.58
	Qian et al. [56]	63.89	53.89
FF_CE	Mohiuddin et al. [6]	63.71	56.43
	Ganguly et al. [14]	68.04	66.12
	Zhao et al. [13]	–	67.44
	Miao et al. [12]	–	66.12
	Proposed	79.34	87.58

handcrafted features with deep learning features first and then employed all these FS techniques for comparison purposes. All these experiments are conducted 10 times and the results obtained are shown in Fig. 9. These comparative results show that, in terms of average test accuracy and the number of selected features, the proposed hybrid FS outperforms other FS methods used here for comparison. Moreover, the better performance of the GWO–VS-based FS technique over the GWO-based FS justifies the choice of GWO–VS over the GWO algorithm within the proposed HFS methods.

4.6.1 Statistical test among different FS proposals

We have also performed a statistical test to check whether the result of the HFS method is statistically significant with respect to the other FS methods used here for comparison. The objective of the test is to check whether there is enough evidence to “reject” the null hypothesis: there is no statistically significant difference between the performances of HFS and another method. Here we have used the non-parametric statistical test known as Wilcoxon rank-sum test following the approach described in [55]. According to this test, if the estimated p value < 0.05 , then we reject the null hypothesis at a 5% significance level. To calculate the p value, we performed 10 independent trials for all these FS methods and the estimated p values are recorded in Table 7. From the estimated p values we can

infer that the null hypothesis can be rejected i.e., the higher performance obtained by the proposed HFS method is statistically significant.

4.7 Comparison with state-of-the-art deepfake detection methods

We have compared the performances of the proposed deepfake detection method with the following state-of-the-art deepfake methods [4, 5, 12, 13, 29, 56]. To avoid any bias imposed by different evaluation processes, all the methods except [12, 13] have been evaluated on the current dataset splits (see Table 1) and using our own experimental setup. Table 8 lists the comparative performances of all the methods. Table 10 shows the image level detection results of these state-of-the-art models. In Table 10, the terms ‘Yes’ and ‘No’ indicate whether the image under consideration is correctly and incorrectly recognized by the underlying method, respectively, whereas the symbol ‘–’ indicates that the corresponding method has not been evaluated on this dataset. The results show that the performance of the proposed method is comparable with the state-of-the-art methods. The proposed method outperforms all the methods used here for comparison on the CeDF dataset but fails to achieve state-of-the-art results in the FF++ dataset in terms of the AUC score. We would also like to mention that our experimental setups are different from the works proposed by Zhao et al. [13] and Miao et al. [12] in terms of the number of samples used for training and testing, preparing cropped face images, the number of frames per video, etc. which might be the reason for worse performance as compared to these works.

4.8 Performance on the DFDC dataset

The Deepfake Detection Challenge (DFDC) dataset was introduced in a Kaggle contest [54] and later made public by Facebook AI [57], which is one of the most comprehensive third-generation forensics datasets to date. The training, validation, public test, and private test sets consist of 119,154, 4000, 5000, and 10000 clips (10 s each), respectively. In our work, we have used 86,166 video clips out of 119,154 available for training our model due to computational constraints. This subset is obtained by

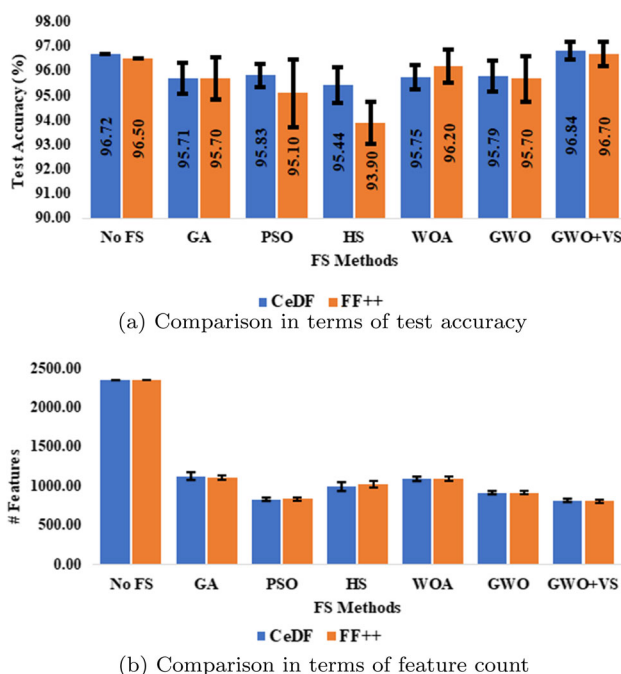


Fig. 9 Comparison of accuracies for various state-of-the-art FS methods on CeDF and FF++ datasets. Here, No FS indicates that classification is performed using the original features (i.e., Feature_5)

Table 7 Estimated p values using Wilcoxon rank-sum test

Dataset	GA	PSO	HS	WOA	GWO
CeDF	0.0020	0.0031	0.0014	0.0002	0.0001
FF++	0.0005	0.0015	0.0005	0.0001	0.0001

Table 8 Performance of the proposed model on the intra-dataset experimental setup

Dataset	Methods	Model	# Features	Test accuracy (in %)	AUC score (in %)
CeDF	Li et al. [4]	XceptionNet	2048	95.37	98.88
	Afchar et al. [29]	MesoNet	1024	65.64	65.33
	Afchar et al. [29]	MesoInception	1024	65.83	64.80
	Qian et al. [56]	F3-Net	4096	87.06	81.48
	Guo et al. [5]	AMTEN	300	68.33	78.04
	Proposed	HFS	270	97.30	99.35
	Li et al. [4]	XceptionNet	2048	96.00	98.34
	Afchar et al. [29]	MesoNet	1024	67.50	66.81
	Afchar et al. [29]	MesoInception	1024	65.00	66.93
FF++	Qian et al. [56]	F3-Net	4096	95.50	96.52
	Guo et al. [5]	AMTEN	300	78.50	87.62
	Zhao et al. [13]	Multi-Attention + XceptionNet	–	96.37	98.97
	Zhao et al. [13]	Multi-Attention + EfficientNet-B4	–	97.60	99.29
	Miao et al. [12]	FRLM	–	97.63	99.50
	Proposed	HFS	297	98.00	99.16

Bold values describe the maximum accuracy and AUC reached by the corresponding method

Table 9 Performance of the proposed HFS model and its components (see Fig. 6) on DFDC dataset

Parameter	Feature_H	Feature_D	Feature_1	Feature_2	Feature_3	Feature_4	Feature_5	Feature_6
Test accuracy (in %)	49.91	69.19	51.91	72.03	66.83	75.34	71.81	73.07
AUC score (in %)	51.01	77.59	53.66	79.88	75.63	85.67	80.19	81.24
# Features	300	2048	102	653	755	240	2348	859

considering only the first frame of each video clip in the training set. However, we have used the whole testing set of 5000 public video clips to evaluate the performance of our model. We have also used all the videos from the validation set to evaluate objective function values during the FS phase. The experimental results are recorded in Table 9. We have also reported the performances of four state-of-the-art methods [5, 6, 9, 14] (publicly available codes were used) in Table 11. Table 10 shows the image-level detection results of the state-of-the-art models on the DFDC dataset. From these results, it can be observed that the performance of the proposed method is comparable with state-of-the-art methods considered here for comparison (Table 11).

5 Conclusion

In recent times, the detection of deepfake videos and images has become a challenging task for researchers because of their striking resemblance to the real ones. In this paper, an HFS-based method is proposed, which considers both handcrafted and deep learning features obtained from the input, and selects only important feature elements by ignoring the irrelevant ones. Moreover, the proposed HFS method allows for a drastic reduction of the dimensions of the descriptors with a high impact on the performance of the system. An exhaustive set of experiments has been conducted on three benchmark datasets, namely CeDF, FF++, and DFDC, and obtained 99.35%, 99.16%, and 85.67% AUC scores, respectively. The results achieved by the proposed method are comparable with most state-of-the-art methods considered here for comparison. However, there are still some rooms to improve the proposed method as its performances are not much satisfactory, especially when considering the cross-dataset

Table 10 Comparison with state-of-the-art methods at image level


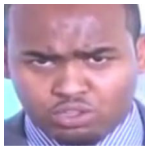
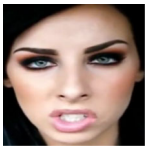



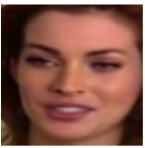



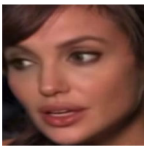

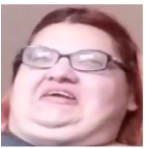
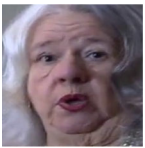

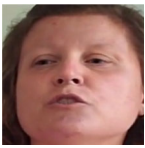
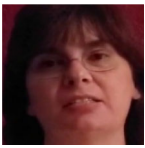

Method	Real			Fake		
						
<i>FF++</i>						
Li et al. [4]	Yes	Yes	No	Yes	Yes	No
Afchar et al. [29]	Yes	Yes	No	Yes	No	No
Guo et al. [5]	Yes	No	Yes	No	Yes	No
Mohiuddin et al. [6]	Yes	Yes	No	No	Yes	No
Ganguly et al. [14]	No	Yes	No	Yes	Yes	No
Proposed	Yes	Yes	No	Yes	Yes	No
Method	Real			Fake		
						
<i>CeDF</i>						
Li et al. [4]	No	Yes	No	Yes	No	No
Afchar et al. [29]	No	Yes	No	No	Yes	No
Guo et al. [5]	Yes	No	Yes	Yes	No	No
Mohiuddin et al. [6]	Yes	No	Yes	No	No	Yes
Ganguly et al. [14]	Yes	No	Yes	No	Yes	No
Proposed	Yes	Yes	Yes	Yes	No	Yes
Method	Real			Fake		
						
<i>DFDC</i>						
Li et al. [4]	–	–	–	–	–	–
Afchar et al. [29]	–	–	–	–	–	–
Guo et al. [5]	No	No	Yes	Yes	No	Yes
Mohiuddin et al. [6]	No	No	Yes	Yes	Yes	No
Ganguly et al. [14]	No	Yes	No	Yes	Yes	Yes
Proposed	Yes	Yes	Yes	Yes	No	Yes

Table 11 Performance comparison of different models tested on the DFDC dataset

Method	Performance (in %)	
	AUC score	Test accuracy
Guo et al. [5]	74.19	76.82
Mohiuddin et al. [6]	71.77	71.30
Ganguly et al. [14]	83.59	75.20
Ganguly et al. [9]	86.32	73.21
Proposed	85.67	75.34

Bold values describe the maximum accuracy and AUC reached by the corresponding method

scenario, by considering unpleasant effects on society caused by deepfake videos/images. The base model XceptionNet can be replaced by other CNN models to check the performance. In the future, attention-based CNN networks can also be tried as deep learning feature extractors. Besides, the proposed method can be applied to other datasets to test its robustness.

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s00521-023-08201-z>.

Acknowledgements We are thankful to the Center for Microprocessor Applications for Training Education and Research (CMATER) research laboratory of the Computer Science and Engineering Department, Jadavpur University, Kolkata, India for providing infrastructural support.

Funding The authors gratefully acknowledge financial support from ANID Fondecyt 1231122, PIA/PUENTE AFB220003.

Data availability statement All the data used in this work are publicly available to the researchers through the works [4, 28, 54].

Declarations

Conflict of interest All the authors declare that they have no conflict of interest or competing interest.

References

- McCloskey S, Albright M (2019) Detecting GAN-generated imagery using saturation cues. In: IEEE international conference on image processing (ICIP). IEEE, pp 4584–4588
- Durall R, Keuper M, Pfrendt FJ, Keuper J (2019) Unmasking deepFakes with simple features. arXiv preprint [arXiv:1911.00686](https://arxiv.org/abs/1911.00686)
- Li Y, Chang MC, Lyu S (2018) In ictu oculi: exposing AI generated fake face videos by detecting eye blinking. In: IEEE international workshop on information forensics and security (WIFS). IEEE, pp 1–7
- 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
- Guo Z, Yang G, Chen J, Sun X (2021) Fake face detection via adaptive manipulation traces extraction network. *Comput Vis Image Underst* 204:103170
- Mohiuddin S, Ganguly S, Malakar S, Kaplun D, Sarkar R (2022) A feature fusion based deep learning model for deepfake video detection. In: International conference on mathematics and its applications in new computer systems. Springer, pp 197–206
- Chollet F (2017) Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1251–1258
- Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2017) Grad-CAM: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision, pp 618–626
- Ganguly S, Ganguly A, Mohiuddin S, Malakar S, Sarkar R (2022) ViXNet: vision transformer with Xception Network for deepfakes based video and image forgery detection. *Expert Syst Appl* 210:118423
- Wang C, Deng W (2021) Representative forgery mining for fake face detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 14923–14932
- Chen S, Yao T, Chen Y, Ding S, Li J, Ji R (2021) Local relation learning for face forgery detection. In: Proceedings of the AAAI conference on artificial intelligence, pp 1081–1088
- Miao C, Chu Q, Li W, Li S, Tan Z, Zhuang W et al (2021) Learning forgery region-aware and ID-independent features for face manipulation detection. *IEEE Trans Biom Behav Ident Sci* 4(1):71–84
- Zhao H, Zhou W, Chen D, Wei T, Zhang W, Yu N (2021) Multi-attentional deepfake detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 2185–2194
- Ganguly S, Mohiuddin S, Malakar S, Cuevas E, Sarkar R (2022) Visual attention based deepfake video forgery detection. *Pattern Anal Appl* 25:1–12
- Das S, Chatterjee A, Dey S, Saha S, Malakar S (2023) Breast cancer detection from histology images using deep feature selection. In: Basu S, Kole DK, Maji AK, Plewczynski D, Bhattacharjee D (eds) Proceedings of international conference on frontiers in computing and systems: COMSYS 2021. Springer, Berlin, pp 323–330
- Banerjee D, Chatterjee B, Bhowal P, Bhattacharyya T, Malakar S, Sarkar R (2021) A new wrapper feature selection method for language-invariant offline signature verification. *Expert Syst Appl* 186:115756
- Ghosh M, Malakar S, Bhowmik S, Sarkar R, Nasipuri M (2017) Memetic algorithm based feature selection for handwritten city name recognition. In: International conference on computational intelligence, communications, and business analytics. Springer, pp 599–613
- Bommert A, Sun X, Bischl B, Rahnenführer J, Lang M (2020) Benchmark for filter methods for feature selection in high-dimensional classification data. *Comput Stat Data Anal* 143:106839
- Tola E, Lepetit V, Fua P (2009) DAISY: an efficient dense descriptor applied to wide-baseline stereo. *IEEE Trans Pattern Anal Mach Intell* 32(5):815–830
- Chatterjee A, Malakar S, Sarkar R, Nasipuri M (2018) Handwritten digit recognition using DAISY descriptor: a study. In: 2018 Fifth international conference on emerging applications of information technology (EAIT). IEEE, pp 1–4
- Majumder S, Ghosh S, Malakar S, Sarkar R, Nasipuri M (2021) A voting-based technique for word spotting in handwritten document images. *Multimed Tools Appl* 80(8):12411–12434
- Malakar S, Ghosh M, Bhowmik S, Sarkar R, Nasipuri M (2019) A GA based hierarchical feature selection approach for

- handwritten word recognition. *Neural Comput Appl* 32(7):2533–2552. <https://doi.org/10.1007/s00521-018-3937-8>
23. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. *Adv Eng Softw* 69:46–61
 24. Doğan B, Ölmez T (2015) A new metaheuristic for numerical function optimization: vortex Search algorithm. *Inf Sci* 293:125–145. <https://doi.org/10.1016/j.ins.2014.08.053>
 25. Matern F, Riess C, Stamminger M (2019) Exploiting artifacts visual, to expose deepfakes and face manipulations. In: IEEE winter applications of computer vision workshops (WACVW). IEEE, pp 83–92
 26. 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
 27. Tolosana R, Romero-Tapiador S, Fierrez J, Vera-Rodriguez R (2021) DeepFakes evolution: analysis of facial regions and fake detection performance. In: International conference on pattern recognition. Springer, pp 442–456
 28. 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
 29. Afchar D, Nozick V, Yamagishi J, Echizen I (2018) MesoNet: a compact facial video forgery detection network. In: IEEE international workshop on information forensics and security (WIFS). IEEE, pp 1–7
 30. 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
 31. Dang H, Liu F, Stehouwer J, Liu X, Jain AK (2020) On the detection of digital face manipulation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5781–5790
 32. Tolosana R, Vera-Rodriguez R, Fierrez J, Morales A, Ortega-Garcia J (2020) Deepfakes and beyond: a survey of face manipulation and fake detection. *Inf Fusion* 64:131–148
 33. Mirsky Y, Lee W (2021) The creation and detection of deepfakes: a survey. *ACM Comput Surv (CSUR)* 54(1):1–41
 34. Shaw SS, Ahmed S, Malakar S, Garcia-Hernandez L, Abraham A, Sarkar R (2021) Hybridization of ring theory-based evolutionary algorithm and particle swarm optimization to solve class imbalance problem. *Complex Intell Syst* 7(4):2069–2091
 35. Malakar S, Ghosh M, Chatterjee A, Bhowmik S, Sarkar R (2020) Offline music symbol recognition using Daisy feature and quantum Grey wolf optimization based feature selection. *Multimed Tools Appl* 79(43):32011–32036
 36. Sarkar S, Ghosh M, Chatterjee A, Malakar S, Sarkar R (2018) An advanced particle swarm optimization based feature selection method for tri-script handwritten digit recognition. In: International conference on computational intelligence, communications, and business analytics. Springer, pp 82–94
 37. Davis L (1991) Handbook of genetic algorithms. In: *CumInCAD*
 38. Simon D (2008) Biogeography-based optimization. *IEEE Trans Evol Comput* 12(6):702–713
 39. Qin AK, Huang VL, Suganthan PN (2008) Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans Evol Comput* 13(2):398–417
 40. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks, vol 4. IEEE, pp 1942–1948
 41. Yang XS, Deb S (2009) Cuckoo search via Lévy flights. In: World congress on nature and biologically inspired computing (NaBIC). IEEE, pp 210–214
 42. Karaboga D (2010) Artificial bee colony algorithm. *Scholarpedia* 5(3):6915
 43. Goffe WL, Ferrier GD, Rogers J (1994) Global optimization of statistical functions with simulated annealing. *J Econom* 60(1–2):65–99
 44. Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. *Simulation* 76(2):60–68
 45. Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. *Inf Sci* 179(13):2232–2248
 46. Ahmed S, Ghosh KK, Garcia-Hernandez L, Abraham A, Sarkar R (2020) Improved coral reefs optimization with adaptive β -hill climbing for feature selection. *Neural Comput Appl* 33(12):6467–6486. <https://doi.org/10.1007/s00521-020-05409-1>
 47. Glover F, Laguna M (1998) Tabu search. In: Du D-Z, Pardalos PM (eds) *Handbook of combinatorial optimization*. Springer, Berlin, pp 2093–2229
 48. Alomari OA, Khader AT, Al-Betar MA, Awadallah MA (2018) A novel gene selection method using modified MRMR and hybrid bat-inspired algorithm with β -hill climbing. *Appl Intell* 48(11):4429–4447. <https://doi.org/10.1007/s10489-018-1207-1>
 49. Wang Z, Wu G, Wan Z (2017) A novel hybrid vortex search and artificial bee colony algorithm for numerical optimization problems. *Wuhan Univ J Nat Sci* 22(4):295–306
 50. Chatterjee B, Bhattacharyya T, Ghosh KK, Singh PK, Geem ZW, Sarkar R (2020) Late acceptance hill climbing based social ski driver algorithm for feature selection. *IEEE Access* 8:75393–75408. <https://doi.org/10.1109/access.2020.2988157>
 51. Dey C, Bose R, Ghosh KK, Malakar S, Sarkar R (2022) LAGOA: learning automata based grasshopper optimization algorithm for feature selection in disease datasets. *J Ambient Intell Humaniz Comput* 13(6):3175–3194
 52. Jiang B, Ren Q, Dai F, Xiong J, Yang J, Gui G (2018) Multi-task cascaded convolutional neural networks for real-time dynamic face recognition method. In: International conference in communications, signal processing, and systems. Springer, pp 59–66
 53. Alzer H (1997) On some inequalities for the incomplete gamma function. *Math Comput* 66(218):771–778
 54. Dolhansky B, Howes R, Pflaum B, Baram N, Ferrer CC (2019) The deepfake detection challenge (DFDC) preview dataset. *arXiv preprint arXiv:1910.08854*
 55. Mondal R, Malakar S, Barney Smith EH, Sarkar R (2022) Handwritten English word recognition using a deep learning based object detection architecture. *Multimed Tools Appl* 81(1):975–1000
 56. Qian Y, Yin G, Sheng L, Chen Z, Shao J (2020) Thinking in frequency: face forgery detection by mining frequency-aware clues. In: European conference on computer vision. Springer, pp 86–103
 57. Dolhansky B, Bitton J, Pflaum B, Lu J, Howes R, Wang M et al (2020) The deepfake detection challenge (DFDC) dataset. *arXiv preprint arXiv:2006.07397*

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.