Video Deepfake Detection Using Particle Swarm Optimization Improved Deep Neural Networks

Leandro Cunha[1], Li Zhang[1*], Bilal Sowan[2], Chee Peng Lim[3], and Yinghui Kong[4]

[1]Department of Computer Science, Royal Holloway, University of London, Surrey, TW20 0EX, UK.
[2]Department of Business Intelligence and Data Analytics, University of Petra, Amman, 11196, Jordan.
[3]Institute for Intelligent Systems Research and Innovation, Deakin University, Waurn Ponds, VIC 3216, Australia.
[4]Department of Electronics and Communication Engineering, North China Electric Power University, Hebei, 102206, China.

*Corresponding author(s). E-mail(s): li.zhang@rhul.ac.uk;
Contributing authors: leandro.cunha.2021@live.rhul.ac.uk; bilal.sowan@uop.edu.jo; chee.lim@deakin.edu.au; kongyhbd2015@ncepu.edu.cn;

Abstract

As complexity and capabilities of Artificial Intelligence (AI) technologies increase, so does its potential for misuse. Deepfake videos are an example. They are created with generative models which produce media that replicates the voices and faces of real people. Deepfake videos may be entertaining, but they may also put privacy and security at risk. A criminal may forge a video of a politician or another notable person in order to affect public opinions or deceive others. Approaches for detecting and protecting against these types of forgery must evolve as well as the methods of generation to ensure that proper information is supplied and to mitigate the risks associated with the fast evolution of deepfakes. This research exploits the effectiveness of deepfake detection algorithms with the application of a Particle Swarm Optimization

(PSO) variant for hyperparameter selection. Since Convolutional Neural Networks (CNNs) excel in recognising objects and patterns in visual data while Recurrent Neural Networks (RNNs) are proficient at handling sequential data, in this research, we propose a hybrid EfficientNet-Gated Recurrent Unit (GRU) network as well as EfficientNet-B0 based transfer learning for video forgery classification. A new PSO algorithm is proposed for hyperparameter search, which incorporates composite leaders and reinforcement learning-based search strategy allocation to mitigate premature convergence. To assess whether an image or a video is manipulated, both models are trained on datasets containing deepfake and genuine photographs and videos. The empirical results indicate that the proposed PSO-based EfficientNet-GRU and EfficientNet-B0 networks outperform the counterparts with manual and optimal learning configurations yielded by other search methods for several deepfake datasets.

Keywords: video deepfake detection; EfficientNet; EfficientNet-Gated Recurrent Unit; hyperparameter selection; Particle Swarm Optimization

1 Introduction

Generative models are increasingly being used. They have demonstrated a great deal of success in generating high-quality fake photographs, videos, and audios, which may frequently be impossible to be distinguished from the genuine ones. The Internet makes it possible for anybody to create this type of media. The use of deep learning algorithms to the production of this form of content is a significant contributor to the development of the notion known as "deepfake". A person with malicious intentions is able to create real-time video deepfakes using the tools that are currently available. Such video manipulations involve the replacement of a source individual with a target individual using a series of techniques such as face swapping or lip synchronization to generate an entirely new video.

As continuous advancement in deep generative models, it is getting increasingly difficult to distinguish between authentic and fraudulent photographs and videos. In a study, Nightingale et al. [1] have proven, in two different trials, that people's capacity to discern edited photographs of real-world settings is severely restricted. Their results suggested concern about the degree to which individuals may be misled in their day-to-day lives. According to the authors, this was supported by the fact that manipulated images already command a significant amount of attention in the media, e.g. social networking sites. Additionally, the researchers were unable to find any convincing evidence to support the idea that personal characteristics, such as skills in photography or opinions regarding the degree to which image manipulation is pervasive in society, are linked to a better ability to spot or locate manipulations. This was one of our primary motivations when conducting this study.

The primary means for video and image forgery generation are through the training of generative models using variational autoencoders (VAE), Generative Adversarial Networks (GANs), or various blends of these two types of models with other image processing techniques. The majority of the models will base their networks on Convolutional Neural Networks (CNNs), or, beginning in 2022, more modern models are also capable of using Vision Transformers (ViT).

Some of the newest generation of models are able to produce images with a very subtle level of artefacts. As a result, as pointed out by Sabir et al. [2], the only way to determine whether or not a face is real or fake is by looking for features such as an unnaturally asymmetric face, weird teeth, and other more obvious inconsistencies not localised on the face but in the background. We introduce different types of attacks as follows.

Face Swapping: The face of a source individual in a video is changed to match the form and characteristics of that of a target individual [3]. After having the face of the target individual initially extracted from an image, it is then subsequently transferred to a newly generated image or video. In order to produce the manipulated image, the process typically involves training two encoders on both the source image and the target image and then switching the decoders in order to rebuild the face from image A onto image B. Some applications have gained popularity due to the ease with which they can be deployed and the results they produce. These applications enable even people with little knowledge to create fake images. Natsume et al. [4] proposed an region-separative GAN (RSGAN) model for the generation of synthetic images independently for faces and hair, which led to improved outcomes on face swapping.

Facial Reenactment: The facial reenactment techniques change or reconstruct particular aspects of a face, such as one's head position, expression, eye gaze, or lip movement. GAN is the most adopted facial reenactment image generator. In 2016, Thies et al. [5] developed one of the first tools, namely Face2Face, for facial reenactment. It was a real-time system that created a 3D face model based on the input image and used its 3D geometry to render the fake face. Face2Face was one of the first tools of its kind. Reenactment can also be carried out using purely one video input with the use of a method proposed by [6]. Specifically, the head movement, facial expression, eye gazing, and blinking of the eyes were collected initially, and then transferred to a target actor who was also using a 3D head model. The detection and classification of face swapping and facial reenactment are the primary focuses of this research.

Due to the gravity of the problem and the possible danger that deepfakes pose to social stability, there has been an increase in research aimed at finding a solution to the challenge of identifying deepfakes. Constructing a CNN specifically tailored to the problem at hand, in this instance, detecting deepfakes, is one approach that may be used. But even so, there are a variety of channels that might be explored. For instance, some may choose from a variety

of network designs, while others may customise a number of hyperparameters in accordance with specific tasks. In addition, there are several studies adopting algorithms to handle the processing of an image or a video. An algorithm may, for instance, take into account individual video frames and attempt to locate instances of spatial inconsistency. Alternatively, the algorithm may compare successive video frames in an effort to identify instances of temporal inconsistency.

In this research, we propose transfer learning of CNNs and hybrid CNN-Recurrent Neural Network (RNN) models with Particle Swarm Optimization (PSO)-based hyperparameter selection for deepfake detection. Our system comprises three key steps. (1) Firstly a data preprocessing procedure is applied to crop facial regions to eliminate background distraction. The cropped facial regions are then used as inputs to deep networks for video classification. (2) Specifically, an ImageNet pretrained EfficientNet is fine-tuned using the deepfake datasets with video frames as inputs, while EfficientNet serialized with a Gated Recurrent Unit (GRU) network is used with videos as inputs directly for synthetic video classification. During the training stage, a new PSO algorithm is proposed to conduct optimal hyperparameter search for EfficientNet and EfficientNet-GRU, which integrates composite leader signal generation and reinforcement learning-based search operation deployment to increase search flexibility. (3) Finally, the yielded optimized settings are used to establish the final transfer learning and hybrid networks for fake/real video classification. The research novelties are elaborated as follows.

- To reduce background distraction, a face cropping procedure using a multi-task cascaded deep learning model is used for facial region extraction from video frames.
- A hybrid EfficientNet-GRU network and transfer learning using EfficientNet are proposed for identifying fake from real videos, owing to their great efficiency in extracting spatial-temporal cues and capturing inter/intra-frame inconsistencies. Automated hyperparameter search using the proposed PSO algorithm is also conducted for both networks to further boost performance. The new PSO algorithm combines adaptive nonlinear functions for composite leader generation as well as the Q-learning algorithm for optimal dispatch of different search operations, to overcome local optima traps. Evaluated using several well-known deepfake datasets, the proposed PSO-based EfficientNet-B0 and EfficientNet-GRU networks achieve superior performance over those of existing state-of-the-art methods for video authenticity identification. The proposed optimizer also shows statistical superiority over other search methods in solving a variety of unimodal and multimodal benchmark functions.

2 Related Work

In this section, we discuss state-of-the-art deep neural networks for deepfake detection and swarm intelligence algorithms for optimal hyperparameter fine-tuning.

2.1 Deepfake Detection

One of the first end-to-end trainable architectures for video classification using CNN and RNN was proposed in 2015 by Liang and Hu [7]. Their work exploited recurrent CNN (RCNN) for undertaking object recognition. In 2016, Donahue et al. [8] studied Long-Term Recurrent Convolutional Network (LRCN), where a CNN processed raw visual input and fed it to a stack of recurrent sequential models for spatial-temporal feature extraction. Specifically, the LRCN model adopted CNNs to learn visual features from video frames and passed a sequence of image embeddings through Long Short-Term Memory (LSTM) networks for video classification.

In 2018, after the first deepfakes appeared, the idea to use the hybrid architecture for deepfake detection was first researched and published combining the advantageous characteristics of the RNN to enhance the performance of the CNN. According to Sabir et al. [2], the body of literature that has been most explored to gain insight about video classification for deep fake detection was video action recognition [9] [10] [11] [12] because of the extensive development in the field in recent years and similar spatial-temporal processing nature to that of deepfake detection. One of the main methods of human action recognition is a "two-stream" network methodology, which processes video frames and optical flow in two different branches before fusing them for video classification. [13] presented a deepfake detection with this two-stream technique. In addition, an RCNN model was employed by [2] for deepfake detection. It processed each frame using a CNN, and the extracted spatial features were further processed using an RNN for video forgery identification. Other strategies utilized biological signals and attention layers [14] [15] to enhance the efficiency for manipulated video detection. In particular, these techniques paid special attention to lip movements and eye gaze to check for inconsistencies.

Moreover, Sabir et al. [2] focused on using a CNN followed by a recurrent model with the input as the query video frame sequences. Their model exploited frame-to-frame temporal differences. Their work claimed that since image manipulations were conducted frame-by-frame and temporal discrepancies were expected, low-level face manipulation techniques should show temporal artefacts with inconsistent features across frames. Their work thus aimed to identify such temporal inconsistencies. Specifically, they used a DenseNet to extract features like discontinuous jawlines and blurred eyes, and then retrieved the RNN's final output rather than averaging recurrent features across all time steps as in traditional video classification pipelines.

2.2 Hyperparameter Search

Hyperparameter configurations of deep neural networks have significant effects in reducing or preventing oscillations in gradient descents as well as correcting gradient directions for weight adjustment towards global optima. The local optima, plateau and saddle points in the loss space are the major challenges that deep networks encounter. If hyperparameters of deep neural networks are not appropriately optimized, the networks' performance will be affected significantly by the above factors. Although methods, such as grid and random search, work well for hyperparameter search with discrete values in a small search space, there are other more effective methods like employing swarm-based metaheuristic methods to determine optimal hyperparameter configurations specially in a continuous large search space like loss spaces in deep networks. We explore such an option through an evolutionary algorithm called PSO, which is simple to implement and has been proven by the literature to produce great robustness for learning configuration selection in neural networks.

The evolutionary algorithms are optimization methods that take inspiration from biological processes. The PSO algorithm was proposed by [16] in 1995, which takes inspiration from fish or bird swarm movement.

Because PSO does not rely on gradient descent, one can use an objective function to optimize deep network parameters without relying on its derivatives [9]. In this work, hyperparameters of deep learning models such as the learning rate, dropout rate, image input size and number of frames extracted from videos will be optimized. The objective function will be associated with loss function of the deep learning model to advise search of optimal learning settings.

The way PSO works is by initialising a group of particles in the search space of the function randomly, and at each iteration it checks which particle achieves the lowest value (i.e. the most optimal loss) on the objective function. At each following step, each particle uses the information of the best solutions found by the swarm and itself, along with random exploration factors, to guide the particle's movement [16].

Taking k as the iteration number, the velocity of a given particle i is given by:

$$v_i(k+1) = wv_i(k) + c_1 r_1 (x_i^{pbest}(k) - x_i(k)) + c_2 r_2 (gbest(k) - x_i(k)) \quad (1)$$

where:

- $v_i(k)$ is the velocity of particle i at iteration k;
- w is an inertia weight;
- c_1 and c_2 are parameters called the "cognitive" and "social" coefficients, respectively;
- r_1 and r_2 are randomly generated numbers between 0 and 1;

- *pbest* is the "personal best" position of the particle (i.e. the best position it has achieved so far);
- *gbest* is the "global best" position among all particles in the swarm (i.e. the best position achieved by the swarm);

The position of the particle i at iteration $k+1$ will be updated with the velocity as follows:

$$x_i(k+1) = x_i(k) + v_i(k+1) \tag{2}$$

The inertia weight controls how much of the particle's previous velocity is kept in the update. A greater w setting indicates that the particle's previous velocity has strong effects to the new velocity generation. The cognitive and social weights, define how much the particle is impacted by its own best past experiences (*pbest*) and the best experiences of the other particles in the swarm (*gbest*), respectively [17]. In general, the values of c_1 and c_2 should be greater than 0, and less than or equivalent to 2.5. Setting these values too low may lead the particles to fail to sufficiently explore the search space, whereas setting them too high may cause the particles to become extremely sensitive to changes in the swarm and exhibit suboptimal behaviour.

In short, PSO is a powerful optimization technique that has been used to solve a wide range of optimization problems. The velocity update formula is critical in establishing how the swarm particles move and update their positions in search of a satisfactory optimal solution. Variant methods have also been proposed to tackle local optima traps of the original PSO algorithm, which were widely adopted in hyperparameter and architecture search in deep networks [18] [19] [20].

There are inspiring related studies for hyperparameter and architecture search using multi-task learning. For example, automatic generation of multi-task learning models was conducted by Zhang et al. [21] for solving a variety of semantic segmentation problems. The automation process utilized a randomly assigned backbone network in conjunction with a set of tasks as inputs with the attempt to generate a multi-task model with a reasonable trade-off between performance and cost. A gradient-based search method was used for architecture search. The optimization process determined the assignment of different network nodes for each task and how these selected nodes were shared with other tasks. A unique characteristic of their work was the adoption of parameter sharing at the operator (neuron) level via a joint optimization of shared policies and network weights. Their yielded multi-task model showed great capabilities in tackling diverse multi-class semantic segmentation problems. In addition, automated production of search parameter and search mechanisms for metaheuristic algorithms was exploited by Stützle and López-Ibáñez [22]. Such techniques were capable of developing optimizers with effective new search strategies. They also showed great efficiency in enhancing existing search

methods' performance via optimal parameter selection. Furthermore, an adaptive hybridized multi-task learning framework was developed by Lialestani et al. [23] pertaining to temperature prediction at different depth levels. Their work performed architecture generation of a multi-task multilayer perceptron neural network using a FA variant developed by Shahri et al. [24]. The FA variant conducted multi-task network architecture generation, where the absorption and randomization parameters were fine-tuned by the population brightness variance.

Cheng et al. [25] developed a multi-task learning model with a hybrid CNN-transformer encoder for simultaneous image segmentation and classification using multimodal MRI image inputs. A U-Net-like encoder-decoder architecture was proposed with an additional transformer unit embedded in the bottom of the CNN-based encoder. The hybrid CNN-transformer encoder fused high-level spatial and global features extracted by a CNN-stream and a transformer-based operation respectively. The joint learning of both segmentation and classification tasks was conducted via a compound loss function integrating segmentation and classification losses with uncertain weights. To tackle data sparsity and unlabelled data, a semi-supervised joint learning mechanism was deployed to enhance classification performance by integrating with uncertainty-based label selection.

2.3 Other Swarm Intelligence Algorithms

Besides PSO, in recent years, a number of new state-of-the-art swarm intelligence algorithms have been proposed including Spotted Hyena Optimizer, Symbiotic Organisms Search, Tree Seed Algorithm, Sparrow Search Algorithm and Tunicate Swarm Algorithm, for tackling engineering, mathematics and image processing optimization problems. Proposed by Cheng and Prayogo [26], Symbiotic Organisms Search employs mutualism, commensalism, and parasitism processes to simulate mutual interaction of two organisms to lead the search of global optimality. Firstly, during mutualism, the mean position vector of the current search agent and another randomly selected organism is calculated, which is used in conjunction with the global best solution to update the positions of both the current and randomly selected individuals. Their offspring solutions are used to replace them if the new solutions are fitter. Secondly, for the commensalism stage, the difference between the global best solution and a randomly selected individual is used to guide the movement of the current search agent. Subsequently, the parasitism operation randomly mutates the dimensions of the current search agent, which is used to substitute a randomly selected organism if this mutated solution is fitter. The effectiveness of Symbiotic Organisms Search was evidenced by its capabilities in handling diverse engineering and benchmark optimization problems, as indicated in a related study [27]. Modified Symbiotic Organisms Search algorithms and its hybridation with other search methods were also extensively studied in [27] to guide future development. Motivated by the cluster hunting behaviours of the spotted hyenas, the search process of Spotted Hyena Optimizer [28] comprises

encircling/hunting and attack mechanisms. The encircling/hunting operation performs local exploitation and intensifies the search around the global best solution. Specifically, the leader spotted hyena with the best fitness score is used to re-allocate the remaining spotted hyenas to its optimal neighbouring regions. Those spotted hyenas with high correlations to the leader form a cluster where their mean position is used to generate a new swarm leader. Adaptive search coefficients are exploited to balance local and global search operations. Diverse Spotted Hyena Optimizer variant methods including the integration with other swarm intelligence algorithms such as PSO and Simulated Annealing (SA) as well as other local/global search strategies were extensively studied in Ghafori and Gharehchopogh [29]. Their flexibilities were further demonstrated in solving a variety of complex single and multi-objective optimization problems [29].

A Sparrow Search Algorithm was developed by Xue and Shen [30] where the population was composed of top ranking producer and lower ranking scrounger subswarms. In addition, 10%-20% of the sparrows are capable of perceiving danger. The top ranking producers perform global exploration when a randomly generated alarm coefficient is lower than the pre-defined safety threshold, otherwise the producer subswarm conducts local exploitation using Gaussian distribution. The lower ranking scrounger subswarm is guided by the producers to exploit optimal local regions of the respective producers. The sparrows with the capabilities of sensing danger follow the swarm leader while staying away from the global worst solution. The algorithm outperformed other classical search methods for solving a number of numerical optimization problems. To overcome slow convergence of the model, a number of variant methods of the Sparrow Search Algorithm were studied by Gharehchopogh et al. [31]. These include the incorporation of PSO, Firefly Algorithm (FA) [32], Differential Evolution (DE) and Since Cosine Algorithm (SCA) [33] with Sparrow Search Algorithm respectively. Other enhancement mechanisms such as random walk based on Levy flights and chaotic map-based swarm initialization are also exploited to increase search robustness. Related studies of neural architecture and hyperparameter search using the Sparrow Search Algorithm were also investigated in [31]. A Tree Seed Algorithm was exploited by Kiran [34]. A swarm of tree solutions is randomly initialized. For each tree, a number of seed solutions are generated. Specifically, each new seed solution is generated using two sub-dimension-based search operations. One is guided by the best tree solution and a randomly selected tree position while the other is led by the current tree position and a randomly selected tree location. For the generation of a specific dimension of a seed solution, the selection of these two search strategies is controlled by a randomly generated threshold parameter. The number of the new seed solutions that can be produced for each tree is dynamic between 10% and 25% of the population size, in order to increase search exploitation. If the best offspring seed solution is fitter than the tree solution, it is used to substitute the tree solution. The algorithm obtained competitive performance in comparison with other search methods such as PSO

and FA for solving 24 numerical test functions. A comprehensive survey of the Tree Seed Algorithm was conducted by [35] where a variety of variants of Tree Seed Algorithm were analysed. The variant methods included the combination of Tree Seed Algorithm with other swarm intelligence algorithms such as Artificial Bee Colony (ABC) [36] and SCA. Improvement strategies such as Levy and Gaussian distributions were also utilized to enhance flexibility of Tree Seed Algorithm. Moreover, the effectiveness of Tree Seed Algorithm was also ascertained by handling a variety of real-world optimization problems such as feature selection and image compression. A variant method of Tunicate Swarm Algorithm was studied by Gharehchopogh [37], which included Quantum Rotation Gate (QRG) and mutation operators based on Cauchy, Gaussian and Levy distributions to increase search robustness of the original method. In particular, besides using QRG, their work explored the effectiveness of the combinations of any two out of the three mutation operators as well as the integration of all three random walk strategies. The superiority of the full model integrating all mutation operators along with QRG was ascertained by solving a set of 52 unimodal, multimodal, composition and hybrid test functions, as well as several other engineering optimization problems.

A new FA variant was developed by Shahri et al. [24] by incorporating a brightness expectation value and a generalized weighted average of a random brightness. It exploited an adaptive absorption coefficient and an adaptive randomization search step to better balance the search between intensification and diversification. The population fitness variance was used to adjust these adaptive search parameters after a number of iterations, which was calculated using the difference between the fitness of each firefly and the mean fitness of the overall swarm, divided by a dynamic normalization factor. Owing to the adaptive adjustment of the search parameters based on the fitness variations during the search process, their method showed better capabilities in overcoming local optima traps in comparison with FA for solving several benchmark functions as well as multi-objective blasting engineering problems.

Motivated by foraging behaviours of social spiders, Social Spider Optimizer (SSO) [38] first generates a vibration intensity of each spider whereby a better fitness score aligns with a stronger vibration. The strongest vibration intensity generated by other spiders and sensed by the current spider is extracted. A randomly generated binary mask is used to select either this new best vibration intensity or another vibration intensity generated by a random individual in each dimension for the construction of new personal leader signal. This new elite leader signal is used to guide a random walk operation for position updating. Boundary checkings are also performed after position updates. SSO shows competitive performance as compared with a number of state-of-the-art search algorithms for tackling diverse numerical optimization problems. Besides the above, there are also other swarm intelligence algorithms developed for handling feature selection, hyperparameter search, deep neural architecture generation with respect to image segmentation/classification [19]

[39] [40] [41], human action recognition [42] and environmental sound classification [18], as well as solving other engineering and mathematical optimization problems [43] [44] [45] [46] [47] [48] [49].

3 The Proposed Methods for Deepfake Detection

The proposed deepfake detection system consists of three key steps, i.e. (1) data preprocessing for the extraction of cropped facial regions, (2) the proposed PSO-based hyperparameter optimization during network training stage, and (3) model establishment using the selected optimal settings and subsequent evaluation using unseen test samples. In particular, transfer learning with EfficientNet as the backbone as well as a hybrid EfficientNet-GRU model is studied in conjunction with PSO-based hyperparameter search for synthetic video classification. We introduce each key stage below.

3.1 Data Preprocessing

The initial stage of the training pipeline involves extracting and pre-processing the first 150 frames of each video. The Python OpenCV library was used to extract the image frames, and then the faces on each frame were processed through the Multi-task cascaded CNN (MTCNN) face detector [50] for cropping. After that, the face crops were organised into folders and saved as image files within the file system. In particular, the cropped facial regions from the real videos are augmented during training by flipping them horizontally to increase real sample sizes. Figure 1 shows the detailed preprocessing pipeline for face cropping.

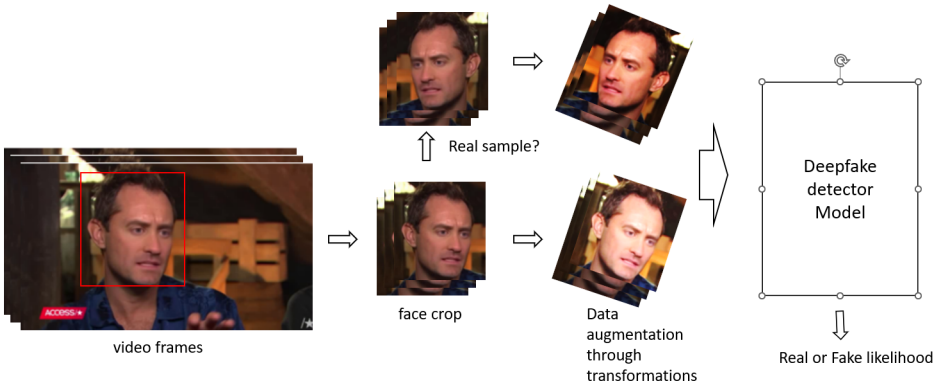

Fig. 1 The preprocessing pipeline for face cropping

Proposed by Zhang et al. [50], the MTCNN model is used for face detection. Specifically, the model is able to perform face classification, facial region

bounding box generation and facial alignment. MTCNN firstly deploys a proposal CNN to perform binary (face and non-face) classification, and generate a number of candidate bounding box regression vectors. The nonmaximum suppression (NMS) method is used to merge highly overlapped bounding boxes. A second CNN model is subsequently utilized to further refine the bounding box regression results by rejecting remaining candidate false positives. A third comparatively deeper CNN is used in this stage to determine the final bounding box output as well as generate a set of facial landmarks indicating positions of both eye centres, left and right mouth corners and the nose tip. The MTCNN model outperformed other face detection benchmarks while maintaining efficient computational cost.

In this research, we employ MTCNN to perform real-time facial bounding box regression for all sampled frames extracted from the video, without using associated facial landmark outputs. The detected facial regions determined by the bounding box regression vectors are cropped out for subsequent classification. Owing to the fact that a region of interest containing the entire face is tracked through the overall video using bounding boxes, the false positives of face classification and localization are greatly reduced. This in turn significantly improves real and manipulated video classification performance. Figure 2 shows the results for face detection and cropping for a sample video.

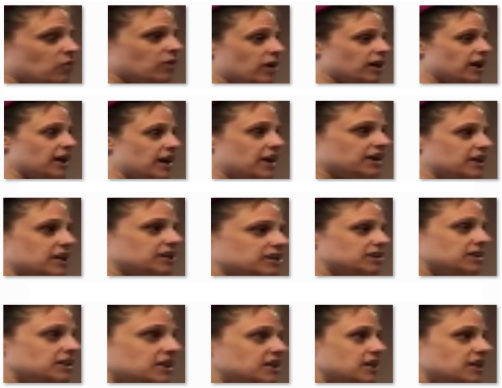

Fig. 2 Example outputs for face detection and cropping for a sample video clip

We use two well-known video deepfake datasets, i.e. Celeb-DFv2 and DFDC, for model evaluation. Precisely, the Celeb-DFv2 dataset [51] consists of 590 genuine and 5,639 fake videos. The official split of the dataset shows 5,711 and 518 videos for training/validation and test, respectively. We adopt this official split in our experiment.

The DFDC dataset [52] has a total of 23,654 real and 104,500 synthetic videos. We extract a subset of 1,016 original and 8,425 tampered videos in our experiments. A test set of 206 real and 1,636 fake videos is used for testing

with the remaining videos for training and validation in our experiment. We further split the training and validation sets using a ratio of 80-20.

Besides the evaluation of each of the above datasets, we also generate a customised dataset by combining the above two datasets with a video face recognition database, i.e. YouTube Faces Database [53], for model evaluation. The YouTube Faces Database is designed for video face recognition, and consists of 3,425 videos from 1,595 subjects, with an average of 181.3 frames per video. It is used to increase the genuine video sample sizes to balance the large numbers of fake instances provided by Celeb-DFv2 and DFDC. To be specific, at the training stage, all real videos from the official training set in Celeb-DFv2, a comparatively larger number of real videos from DFDC, as well as 1,618 videos from the Youtube Faces Database with more than 50 frames, were combined together to construct the customised genuine training video set. In addition, a balanced number of fake videos are drawn from the official training set of Celeb-DFv2 and our DFDC subset in order to obtain a ratio of approximately 50%-50% between fake and real videos, in the constructed training set. We further split the combined training set by a ratio of 80-20 for training and validation respectively. The real and fake samples from the official test set of Celeb-DFv2 and a comparatively larger DFDC unseen test set are used for model evaluation in this experiment.

Table 1 shows the detailed training/validation and test sample sizes for each dataset.

Table 1 Data split of each dataset

	Training/Validation			Test		
Dataset	Real	Fake	Total	Real	Fake	Total
Celeb-DFv2	412	5,299	5,711	178	340	518
DFDC	810	6,789	7,599	206	1,636	1,842
Combined	3,683	3,739	7,422	605	2,770	3,375

The final step was to send the data to the Pytorch dataloader so that the models could be trained and validated for each experimental setting. During the training process, the transformations serve to supplement the data by changing each frame from real videos at each epoch with a random component. This is accomplished through the use of augmentation. For the training dataset, we initially used random rotation with 20 degrees and gaussian blur. The images were initially resized and then normalised before being used in either the training/validation sets or the test set. For the oversampling of frames from real videos, the RandomHorizontalFlip() function is utilized.

3.2 Model 1 - Transfer Learning Using CNN

We firstly employ transfer learning using a CNN model with the EfficientNet architecture for deepfake detection. Figure 3 shows the overall dataflow using transfer learning for synthetic video classification.

EfficientNet was designed with the goal of scaling CNNs more efficiently than other deep networks proposed previously [54]. Since its inception, this CNN architecture has shown to be among those that achieve the highest performance when tested against various image classification benchmarks.

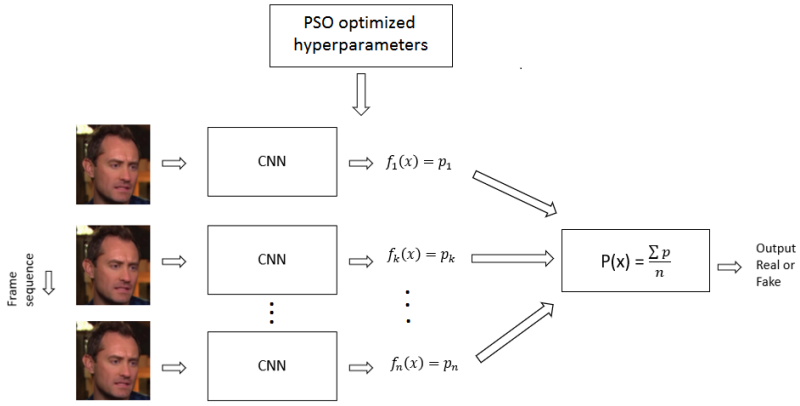

Fig. 3 Classification of real and deepfake videos using EfficientNet-B0

EfficientNet makes use of a compound scaling strategy, which involves scaling the network's width, resolution, and depth uniformly to whatever degree is required to make optimal use of the computational resources available. A grid search is usually used to find the scaling constants [54].

The network design of EfficientNet makes use of mobile inverted bottleneck convolution (MBConv), which is analogous to MobileNetV2 convolutional block but slightly larger. In order to maximise precision and FLOPS, a neural architecture search was employed in the construction of the baseline model. After that, a family of EfficientNet models was obtained by scaling it up using such a strategy. Within the context of this research, the version known as EfficientNet-B0 was employed [54]. The overall architecture specifically used in this study including the fully connected layers is shown in Table 2 below. The pure EfficientNet was also used by the winning solution of the Deepfake Detection Tournament hosted by the DFDC dataset authors in 2019 [52].

The MBConv blocks consist of residual blocks like ResNet that connect the beginning of the block with the end using a skip connection. The difference from the original block from ResNet is that, regarding the number of channels, it follows a narrow-wide-narrow approach instead of the traditional wide-narrow-wide strategy [54].

Table 2 The EfficientNet-B0 Model Architecture [54]

Stage	Operator	Resolution	Channels	Layers
1	Conv3x3	224 × 224	32	1
2	MBConv1 k3x3	112 × 112	16	1
3	MBConv6 k3x3	112 × 112	24	2
4	MBConv6 k5x5	56 × 56	40	2
5	MBConv6 k3x3	28 × 28	80	3
6	MBConv6 k5x5	14 × 14	112	3
7	MBConv6 k5x5	14 × 14	192	4
8	MBConv6 k3x3	7 × 7	320	1
9	Conv1x1 & Pooling	7 × 7	1280	1
10	FC	1	256	1
11	FC with Dropout	1	128	1
12	FC	1	2	1

The EfficientNet-B0 model was initially trained using ImageNet. We further fine-tune the model using the training/validation sets of the frames of each deepfake dataset in our experiments. The fine-tuned model is used for the identification of fake/real videos. In addition, a new PSO variant is used to fine-tune network hyperparameter, i.e. learning rate, dropout rate, image size and number of frames, with the attempt to further enhance performance.

Specifically, a random swarm is firstly initialized in a search space of [0, 1]. Each particle has four dimensions to represent the four optimized hyperparameters. The proposed PSO search operations are used to guide the particle movement in the search space for hyperparameter search. We evaluate each particle's fitness by converting its position into valid network learning configurations, which are used to set up transfer learning process. The network performance on the validation set is used as the fitness measure of each particle. The most optimal solution identified by the proposed otpimizer is used as the recommended best learning configurations of EfficientNet-B0. The optimized EfficientNet-B0 model is then trained using the combined training set with larger numbers of epochs and tested with the respective test sets for deepfake detection.

3.3 Model 2 - Hybrid CNN-RNN

Besides using transfer learning for video forgery detection, motivated by [2] [55] [56][57], a hybrid CNN-RNN architecture is proposed in this research for distinguishing fake from real videos. Specifically, this hybrid model uses EfficientNet serialized with a GRU layer for spatial-temporal feature extraction to inform video classification. The proposed PSO algorithm is used for identifying optimal hyperparameters. Figure 4 shows the system dataflow. The detailed network architecture is shown in Table 3.

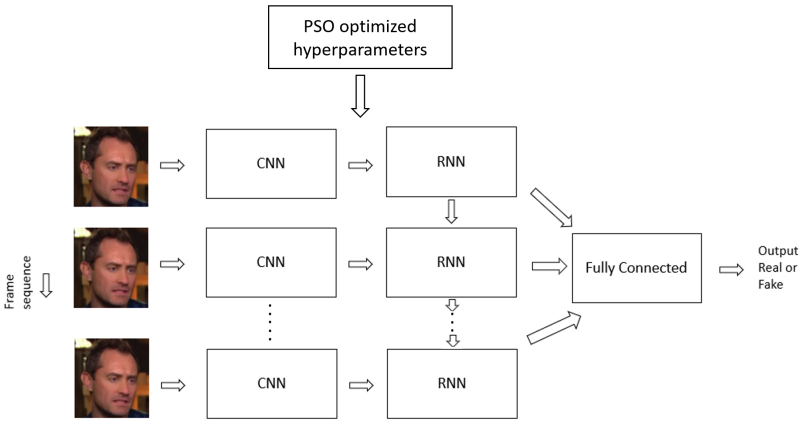

Fig. 4 Classification of real and deepfake videos using EfficientNet-GRU

As shown in Table 3, a latent representation of 1,280 dimension output extracted from the last convolutional layer of EfficientNet is taken as input from each frame and these features of each frame are concatenated to be passed on to the GRU layer. The GRU layer is used to take advantage of the spatial temporal features from the sequence of frames verifying if it is either a deepfake or a real video. The process is different from the pure CNN (i.e. the aforementioned EfficientNet) that takes the average of all frames for deepfake detection as in the transfer learning process. The GRU component has thus 1,280 latent dimensions and 1,280 hidden layers as the layer configurations in this study.

Table 3 Hybrid EfficientNet-GRU Model architecture

Stage	Operator	Resolution	Channels	Layers
1	Conv3x3	224 × 224	32	1
2	MBConv1 k3x3	112 × 112	16	1
3	MBConv6 k3x3	112 × 112	24	2
4	MBConv6 k5x5	56 × 56	40	2
5	MBConv6 k3x3	28 × 28	80	3
6	MBConv6 k5x5	14 × 14	112	3
7	MBConv6 k5x5	14 × 14	192	4
8	MBConv6 k3x3	7 × 7	320	1
9	Conv1x1 & Pooling	7 × 7	1280	1
10	GRU	1	1280	2
11	FC with Dropout	1	256	1
12	FC	1	128	1
13	FC	1	2	1

The optimizer used is ADAM that combines features of optimization algorithms such as RMSProp and ADAGrad. It is used to adjust the learning rate for each weight on the fly using exponential weighted moving average to get the first and second moments of the gradient estimates [56]. The loss function opted is cross-entropy loss. It gives two likelihoods for real and fake labels using a softmax function [56]. In addition, to improve discriminative feature learning, we fine-tune the weights of ImageNet pre-trained EfficientNet-B0 embedded in the proposed EfficientNet-GRU model using the combined training set with a small number of epochs (i.e. 5 epochs), before passing on features to the GRU layer. Moreover, the proposed PSO model is used to fine-tune hyperparameters of this hybrid network during the training stage, similar to the process discussed earlier for parameter search using transfer learning. Specifically, we optimize the learning rate, dropout rate, image size and number of video frames, owing to their significance to network performance.

3.4 The Proposed PSO Model for Hyperparameter Optimization

A new PSO variant is proposed for hyperparameter search for both EfficientNet-GRU and EfficientNet-B0 in this research. In order to tackle limitations of the original PSO algorithm, it incorporates nonlinear functions for composite leader generation and a reinforcement learning strategy for dynamically adjusting the search process. As such, different search actions led by different hybrid leaders and the global best solution are dynamically dispatched based on the reward schemes of the reinforcement learning algorithm. Figure 5 shows the overall proposed algorithm. The detailed search strategies are presented below.

Algorithm 1: Pseudo-Code of the Proposed PSO Algorithm

```
1.  Start
2.  Initialize a swarm of particles randomly;
3.  Evaluate all the particles;
4.  Sort the population based on fitness values and identify gbest;
5.  While (!Stagnation) {
6.      //Customize each particle's search action using reinforcement learning
7.      For (each particle in the swarm) do {
8.          Generate two composite leaders using Equations (3)-(6) and Equations (3), (7) and
            (5)-(6), respectively;
9.          Use Q-learning to select the swarm leader or any of the above newly generated
            composite leaders for position update based on Equations (1)-(2), and (8) and (2),
            respectively;
10.         Update the pbest if the new position is fitter;
11.     } End For
12.     Sort the overall swarm based on the fitness scores and identify gbest;
13. } Until (Stagnation)
14. Output gbest;
15. End
```

Fig. 5 Data flow of the proposed PSO algorithm

3.4.1 Composite Leader Generation

As indicated in existing studies, the original PSO model is likely to be trapped in local optima because of the adoption of a single swarm leader to lead the search process. Therefore, composite leaders are produced by incorporating the global best solution and a distant second leader based on the adaptive weighting factors generated using nonlinear formulae. Equation 3 shows the operation for composite leader generation, where the remote second leader, *sbest*, is obtained by selecting the most distant particle to the swarm leader among the top 5 ranking solutions.

$$composite(k) = w_a \cdot gbest(k) + w_b \cdot sbest(k) \tag{3}$$

where w_a and w_b are the adaptive weighting factors which are used to weigh the effects of the swarm leader and the second leader for composite signal generation. Two sets of nonlinear functions are introduced for weighting coefficient generation.

Equations 4-6 define the first set of formulae for adaptive weighting coefficient production.

$$r = ((\frac{cos(0.5u)}{2})^{\frac{1}{2}} + (\frac{sin(0.5u)}{2})^{\frac{1}{2}})^{-2} \tag{4}$$

$$x = rcos(u) \tag{5}$$

$$y = rsin(u) \tag{6}$$

where $u = [0{:}0.001{:}\pi]$ with x and y denoting the coordinates of the produced 2D points. The above equations generate increasing and decreasing subgraphs, as shown in blue and orange lines respectively in Figure 6. Each comprises 1,571 unique 2D points. We subsequently extract *maximum_iteration* number of values from 1,571 unique y-axis values in the increasing branch with an interval of $\frac{i}{maximum\_iteration}$. These extracted increasing values are used as the weighting factor w_a for the swarm leader. Similarly, *maximum_iteration* number of values are also extracted from 1,571 unique y-axis values in the decreasing subgraph with an interval of $\frac{i}{maximum\_iteration}$. They are subsequently assigned to the weighting coefficient w_b for the second leader. Each pair of increasing w_a and decreasing w_b parameters is utilized for producing a composite leader in each iteration. The adoption of such increasing and decreasing coefficients strengthens the effects of the swarm leader and reduces the influence of the second leader as iteration increases. As such, the algorithm encourages global exploration and intensifies local exploitation at the beginning and end of the search process, respectively.

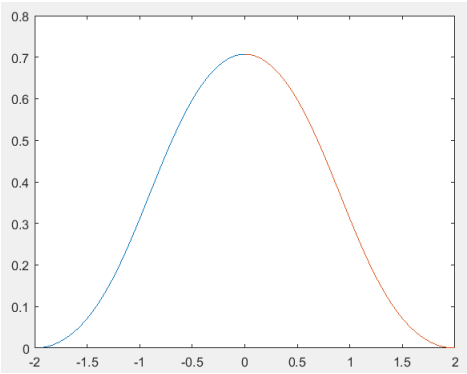

Fig. 6 The resulting increasing and decreasing subgraphs as defined in Equations 4-6

Besides the above, another set of adaptive increasing and decreasing coefficients is also generated using Equation 7 and Equations 5-6, for composite leader generation to increase search flexibility. The resulting increasing and decreasing sub-contours defined by Equation 7 are illustrated in Figure 7 with each containing 1,571 unique 2D points. We also extract $maximum\_iteration$ number of values from 1,571 unique y-axis values in the increasing branch with an interval of $\frac{i}{maximum\_iteration}$, and assign them as the increasing weight coefficient w_a for the swarm leader. The same process is also applied to the decreasing sub-contour for the generation of the weight factor w_b for the second leader. These new sets of w_a and w_b are then utilized for producing composite leaders.

The difference between these new sub-contours defined in Equation 7 and the subgraphs defined by Equation 4 is that these new sub-contours generate larger weighting factors in comparison with those yielded by the previous subgraphs, therefore diversifying the production of the combined leaders.

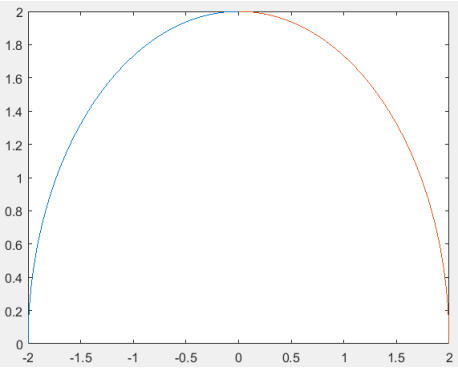

Fig. 7 The resulting increasing and decreasing subgraphs as defined in Equation 7 and Equations 5-6

$$r = ((\frac{cos(0.5u)}{2})^2 + (\frac{sin(0.5u)}{2})^2)^{-\frac{1}{2}} \tag{7}$$

Each composite leader is then used to replace the global best solution in Equation 1 for velocity production with respect to hyperparameter search, as shown in Equation 8. Such composite leaders are able to explore the search space more thoroughly and show enhanced capabilities in tackling stagnation.

$$v_i(k+1) = wv_i(k) + c_1r_1(x_i^{pbest}(k) - x_i(k)) + c_2r_2(composite(k) - x_i(k)) \tag{8}$$

3.4.2 Reinforcement Learning based Optimal Search Action Selection

Owing to the employment of the composite leader generation process, a total of three search operations led by the swarm leader and the aforementioned yielded two composite leaders are constructed. A reinforcement learning algorithm is subsequently used to identify the optimal selection of different leader signals for hyperparameter search. Specifically, in each iteration, each particle is guided by either a composite leader or the global best solution recommended by the Q-learning algorithm [58].

The Q-learning algorithm [58] employs a Bellman equation defined in Equation 9 to identify a sequence of optimal search actions. In reinforcement learning, an agent perceives the environment by learning from punishment and reward signals through trial-and-error. The ultimate goal of the reinforcement learning scheme is to yield a set of optimal search operations that maximize the cumulative reward. Such an expected cumulative reward score for a state-action combination denoted as the Q-value is updated using Equation 9, in the Q-learning algorithm. These Q-values are stored in a Q-table pertaining to each state-action pair.

$$Q^{new}(s_t, a_t) = (1-\theta) \cdot Q(s_t, a_t) + \theta \cdot (r_t + \beta \cdot \max_a Q(s_{t+1}, a)) \tag{9}$$

where θ is the learning rate and β is the discount coefficient. At each time t, the agent performs an action a_t in state s_t resulting in a new state s_{t+1}. Besides the current Q-value $Q(s_t, a_t)$, the new Q-value, $Q^{new}(s_t, a_t)$, is generated based on two additional components, i.e. an immediate reward r_t and a future reward $\max_a Q(s_{t+1}, a)$. After performing a selected search action a_t, the network with the new configuration decoded from the new position is used to test the sampled validation set of the combined dataset, whereby the cross-entropy loss of the sampled validation set is used as the fitness score. If this new fitness score is better than the previous fitness of the particle, an

immediate reward '1' is used for r_t, otherwise '-1' is dispatched. The future reward $\max_a Q(s_{t+1}, a)$ is produced by identifying the action that leads to the maximum reward in the new state s_{t+1}.

Each particle constructs a 3-by-3 Q-table with the rows and columns denoting the states and actions, respectively. Such a Q-table is used to determine the selection of optimal search actions led by either any of the composite leaders or the global best solution. Therefore in each iteration, each particle is assigned with different leader signals to increase search robustness. In comparison with random selection of the search actions as in most existing PSO variants, the Q-learning algorithm produces a sequence of optimal search actions based on the reward principles imposed by the Bellman equation.

The proposed PSO model equipped with composite leader generation and Q-learning based search operation dispatch shows enhanced search capabilities in tackling stagnation in our empirical studies. The hyperparameter search is conducted as follows. Because of the large training and validation sample sizes of the combined dataset, subsets of the training and validation sets are employed for optimal hyperparameter selection. Each element of the particle represents a hyperparameter to be optimized. The optimal hyperparameters recommended by each particle are used to set up a customized deep network. It is subsequently trained and evaluated using the sampled training and validation sets of the combined dataset, respectively. The cross-entropy loss of the sampled validation set is used as the fitness score of each particle. The final optimized network is constructed using the configurations extracted from the global best solution. This final optimized network is trained with a much larger number of training epochs (i.e. 30 epochs) using the overall training set of the combined dataset and tested using Celeb-DFv2, DFDC and combined datasets, respectively. We introduce evaluation details in the following section.

4 Evaluation and Results

We evaluate the transfer learning and hybrid networks with manual and automatic hyperparameter optimization using Celeb-DFv2, DFDC and the combined datasets, respectively. Firstly, for the manual and PSO-based parameter selection, we use the training and validation sets of the combined dataset, since the combined dataset has a mixed data source which may lead to better representative capabilities. The optimized learning configurations are subsequently used to set up each model. Each optimized network is then trained using the combined training set and evaluated using test sets of the Celeb-DFv2, DFDC and combined datasets, respectively. The experimental studies are elaborated in detail below.

4.1 Manual Hyperparameter Selection

In the initial experiments, the models were trained with hyperparameter searched manually. The process entails individually experimenting with a range of hyperparameters selected. The following hyperparameters are optimized:

- Learning rate
- Dropout rate
- Image Size - The size measured by height x width of the input image will influence the result because of the number of pixels processed by the CNN. The ranges evaluated are from 100 to 130 pixels because of the trade-off between performance and cost.
- Number of Frames per Video - This metric influences the result because of the size of the sequence of frames extracted. The maximum limit considered is 50 frames because of comparatively smaller or similar maximum frame settings adopted in existing studies [59] [60] [61].

The training and validation sets of the combined dataset are used for hyperparameter search. For each of the aforementioned hyperparameters, a set of three values was chosen in order to manually fine-tune the model and identify the configuration with the lowest loss. The process was repeated for each one of the four hyperparameters. The values that composed the set were illustrated in Table 4.

Table 4 Hyperparameters searched manually.

Hyperparameter	Ranges
learning rate	$1 \times 10^{-5}, 1 \times 10^{-4}, 1 \times 10^{-3}$
dropout rate	$0.2, 0.3, 0.4$
image size	$100, 112, 130$
frames	$30, 40, 50$

A total of 5 epochs were run to obtain the loss value for each hyperparameter set, as this was the number of epochs that demonstrated satisfactory stability in the preliminary results. To choose the optimal ones in each of them, the algorithm was run through their possible values, while the other hyperparameters remained constant. The parameter setting with the smallest loss error was then selected. These manually selected optimized settings are used to set up each network, which is further tested with test sets of Celeb-DFv2, DFDC and the combined datasets, respectively.

4.1.1 Manual Parameter Search for EfficientNet-B0

As mentioned above, the training and validation sets of the combined dataset are used for hyperparameter search. For the EfficientNet-B0 architecture using transfer learning, the best hyperparameters obtained through this manual process were shown in Table 5.

4.1.2 Manual Parameter Search for EfficientNet-GRU

Similarly, Table 6 comprised the hyperparameters that were determined to be the most optimal ones for the EfficientNet-GRU architecture for the combined dataset.

Table 5 The best hyperparameters identified using manual selection for transfer learning using EfficientNet-B0.

Hyperparameter	Values
learning rate	1×10^{-4}
dropout rate	0.3
image size	112×112 px
frames	30

Both the EfficientNet and EfficientNet-GRU networks equipped with manually selected best model configurations are subsequently evaluated using test sets of Celeb-DFv2, DFDC and the combined datasets, respectively. The detailed evaluation results for both networks with manually identified optimal settings are provided in Section 4.2.

Table 6 The best hyperparameters identified using manual selection for the EfficientNet-GRU network.

Hyperparameter	Values
learning rate	1×10^{-4}
dropout rate	0.3
image size	112×112 px
frames	40

We have also carried out experiments to determine the best search range of the number of frames for automated hyperparameter search. Existing studies such as Wang et al. [62] and Zhao et al. [15] employed 30 frames, while Zheng et al. [59], Shiohara and Yamasaki [60] and Zhao et al. [61] adopted 32 frames for video inference, for evaluating several video deepfake datasets, such as Celeb-DFv2, DFDC and FaceForensics++ (FF++). These studies resized the cropped facial images to larger image resolutions such as 224x224, 256x256 and 380x380, in their experiments. We identify the optimal search range of the frame settings using the combined training set for model training and the official Celeb-DFv2 test set for model testing. We manually set up the frame settings in the range of [10, 100], with the following fixed learning configurations, i.e. learning rate=0.0001, dropout rate=0.3 and image size=112, for both EfficientNet-B0 and EfficientNet-GRU. For both long and short videos, the target number of frames is randomly sampled from each video. The detailed evaluation results, i.e. accuracy rates and Area Under the Curve (AUC) scores, are shown in Tables 7-8.

As indicated in Tables 7-8, experimental results for the EfficientNet-B0 model shows improvements when the frame setting increases from 10 to 30 using the Celeb-DFv2 test set. When further increasing of the number of frames to 50 above, the training cost increases significantly, and the network

Table 7 Experiments using EfficientNet-B0 with different numbers of frames for the Celeb-DFv2 test set

No. of frames	Accuracy	AUC
10	0.7703	0.7086
20	0.7915	0.7421
30	0.8263	0.7780
40	0.8127	0.7610
50	0.7896	0.7393
60	0.7761	0.7264
70	0.7413	0.6825
80	0.7568	0.6929
100	0.7413	0.6745

Table 8 Experiments using EfficientNet-GRU with different numbers of frames for the Celeb-DFv2 test set

No. of frames	Accuracy	AUC
10	0.7741	0.7142
20	0.8089	0.7621
30	0.8224	0.7750
40	0.8417	0.7938
50	0.7992	0.7534
60	0.7896	0.7380
70	0.7703	0.7139
80	0.7780	0.7078
100	0.7625	0.6893

is increasingly becoming overfitting, owing to the capture of irrelevant noise between frames, lowering its performance as indicated in both accuracy rates and AUC scores. A similar case is also observed for the evaluation using EfficientNet-GRU using the Celeb-DFv2 test set. The model shows enhanced performance when using the frame settings ranging from 10 to 40. When further increasing the frame settings to 50 above, both accuracy rates and AUC scores are reduced, because of the extraction of noisy redundant details from video frames. A similar observation is also obtained when using the DFDC and combined test sets. Therefore in order to generate robust networks and balance well between computational cost and performance, we employ the frame setting range of [10-50] for automated hyperparameter search for both networks.

4.2 Automatic Hyperparameter Search Using the Proposed PSO Model

Besides manual hyperparameter selection, automatic hyperparameter search is also performed. We employ the proposed PSO model, as well as 8 classical search methods and 4 PSO variant algorithms, for hyperparameter search,

including PSO, ABC [36], Salp Swarm Algorithm (SSA) [44], SSO [38], Barebones PSO (BBPSO) [63], Flower Pollination Algorithm (FPA) [64], FA [32], Dragonfly Algorithm (DA) [65], Genetic PSO (GPSO) [40], PSO with sine coefficients (SPSO) [20], a BBPSO variant with attractiveness and evade actions (BBPSOV) [49], and PSO with adaptive sine, circle and spiral coefficients (ACPSO) [66]. We adopt variable settings of the above algorithms from their original studies in our experiments.

The following experimental settings are adopted. For each search method, a total of 10 search agents is created and the maximum number of 20 generations is used for hyperparameter search. All the search methods perform the same number (i.e. 200) of function evaluations. A set of 5 runs was conducted for each search method. For both EfficientNet-B0 and EfficientNet-GRU, Table 9 shows the search ranges of different hyperparameters. These search ranges are obtained via trial-and-error as discussed in Section 4.1. A set of five evaluation metrics, i.e. the mean precision, recall, accuracy, AUC scores and the Wilcoxon rank sum (RS) test, is used for performance comparison. The details of the experimental studies are presented as follows.

Table 9 Ranges of hyperparameters optimized by each search method

Hyperparameter	Ranges
learning rate	$1 \times 10^{-5} - 1 \times 10^{-3}$
dropout rate	$0.1 - 0.9$
image size	$100 - 128$
frames	$10 - 50$

Again the training and validation sets of the combined dataset are used for the proposed PSO-based hyperparameter search, owing to their representative capabilities. Because of the large sample sizes of this combined video deepfake dataset, in order to reduce the high computational cost of hyperparameter search, the training and validation sets of the combined dataset were sampled for 5% and 25%, respectively. The validation cross-entropy loss function was used as the objective function for evaluating each particle. The experiments ranged from 10 to 20 hours for each of the two models using the sampled training and validation sets for each hyperparameter search.

4.2.1 Automated Hyperparameter Search for EfficientNet-B0

We conduct automated hyperparameter search for EfficientNet-B0 using different search methods based on the combined training set. The variable configurations of different search methods are taken from existing studies. Additionally for each search method, we adopt a swarm size of 10 and a maximum number of generations of 20. A set of 5 runs was used for hyperparameter search using transfer learning based on EfficientNet-B0. The established final

networks with identified optimal settings by each search method are trained with 30 epochs using the combined training set and tested using the three test sets, respectively.

Table 10 Performance comparison for optimized EfficientNet-B0 models using the Celeb-DFv2 test set

Model	Acc.	Prec.	Recall	AUC	RS
Prop. PSO	0.9247	0.9101	0.9824	0.8985	n/a
PSO	0.8996	0.8830	0.9765	0.8646	9.74E-03
ABC	0.9015	0.8874	0.9735	0.8688	9.74E-03
BBPSO	0.8629	0.8549	0.9529	0.8220	9.74E-03
FPA	0.8900	0.8753	0.9706	0.8533	9.74E-03
SSA	0.8687	0.8579	0.9588	0.8277	9.74E-03
SSO	0.8919	0.8797	0.9677	0.8574	9.74E-03
FA	0.8668	0.8575	0.9559	0.8263	9.74E-03
DA	0.8687	0.8523	0.9677	0.8237	9.74E-03
SPSO	0.8880	0.8770	0.9647	0.8531	9.74E-03
GPSO	0.9131	0.8976	0.9794	0.8830	9.74E-03
BBPSOV	0.8726	0.8605	0.9618	0.8320	9.74E-03
ACPSO	0.8803	0.8639	0.9706	0.8392	9.74E-03
Manual	0.8263	0.8255	0.9324	0.7780	9.74E-03

Table 11 Performance comparison for optimized EfficientNet-B0 models using the DFDC test set

Model	Acc.	Prec.	Recall	AUC	RS
Prop. PSO	0.9414	0.9848	0.9487	0.9161	n/a
PSO	0.8865	0.9741	0.8961	0.8534	2.16E-03
ABC	0.8941	0.9768	0.9022	0.8661	2.16E-03
BBPSO	0.8686	0.9610	0.8881	0.8009	2.16E-03
FPA	0.8833	0.9721	0.8943	0.8452	2.16E-03
SSA	0.8778	0.9682	0.8918	0.8294	2.16E-03
SSO	0.8844	0.9734	0.8943	0.8500	2.16E-03
FA	0.8757	0.9668	0.8906	0.8239	2.16E-03
DA	0.8724	0.9642	0.8894	0.8136	2.16E-03
SPSO	0.8822	0.9714	0.8936	0.8425	2.16E-03
GPSO	0.9050	0.9784	0.9132	0.8765	2.16E-03
BBPSOV	0.8800	0.9689	0.8936	0.8327	2.16E-03
ACPSO	0.8817	0.9702	0.8943	0.8379	2.16E-03
Manual	0.8475	0.9484	0.8759	0.7486	2.16E-03

The detailed evaluation results, i.e. the mean precision, recall, accuracy and AUC scores, as well as the Wilcoxon rank sum test results, for the Celeb-DFv2, DFDC and combined datasets are shown in Tables 10-12. The AUC score is used as the summary measure of the overall model performance. Specifically, a

Table 12 Performance comparison for optimized EfficientNet-B0 models using the combined test set

Model	Acc.	Prec.	Recall	AUC	RS
Prop. PSO	0.9576	0.9852	0.9628	0.9484	n/a
PSO	0.9262	0.9758	0.9332	0.9137	2.16E-03
ABC	0.9292	0.9774	0.9354	0.9181	2.16E-03
BBPSO	0.9218	0.9569	0.9473	0.8761	2.16E-03
FPA	0.9224	0.9736	0.9307	0.9075	2.16E-03
SSA	0.9156	0.9708	0.9249	0.8988	2.16E-03
SSO	0.9233	0.9743	0.9311	0.9093	2.16E-03
FA	0.9289	0.9634	0.9495	0.8921	2.16E-03
DA	0.9239	0.9591	0.9477	0.8813	2.16E-03
SPSO	0.9197	0.9731	0.9278	0.9052	2.16E-03
GPSO	0.9319	0.9793	0.9368	0.9230	2.16E-03
BBPSOV	0.9164	0.9716	0.9253	0.9007	2.16E-03
ACPSO	0.9176	0.9723	0.9260	0.9027	2.16E-03
Manual	0.9049	0.9464	0.9372	0.8471	2.16E-03

higher AUC score correlates with a better classifier. The network with a higher AUC score typically has better capabilities in distinguishing between fake and real instances. In addition, the Wilcoxon rank sum test is also performed based on the AUC scores over 5 runs to indicate the statistical significance of the proposed model over baseline search methods. As depicted in Tables 10-12, the proposed PSO-based EfficientNet-B0 models achieve better results than those with optimal learning parameters obtained using other classical and advanced search methods for all three datasets, in terms of mean precision, recall, accuracy and AUC scores as well as statistical test results. In addition, models with settings yielded by GPSO and ABC show better results than those of the networks optimized by all other baseline methods across the datasets.

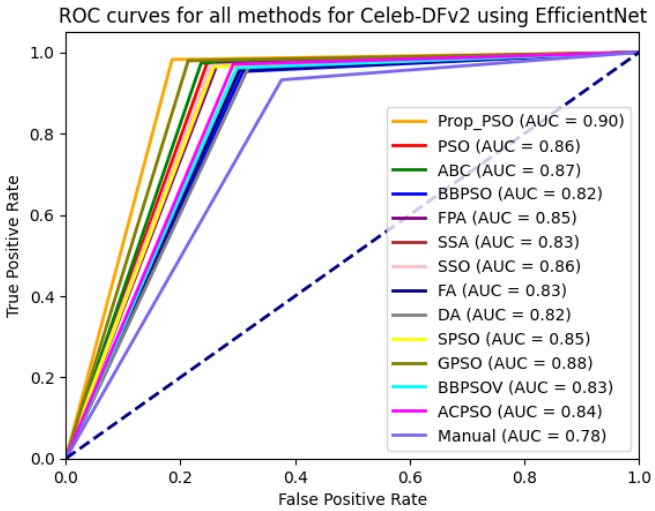

Fig. 8 ROC curves for Celeb-DFv2 using EfficientNet-B0 models with manual and optimal hyperparameters identified by all search methods

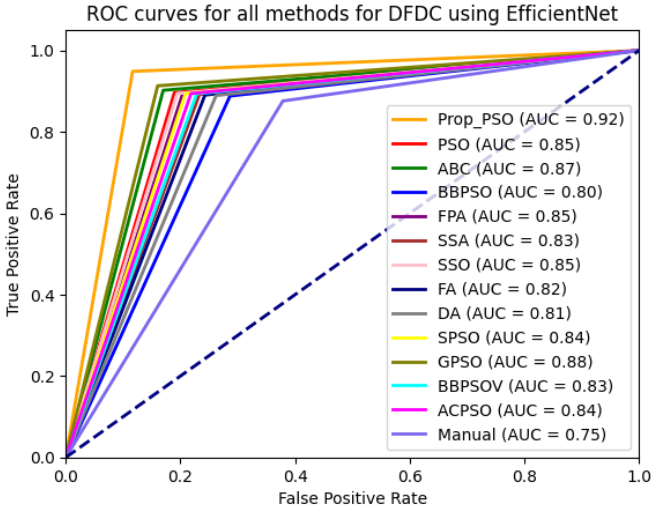

Fig. 9 ROC curves for DFDC using EfficientNet-B0 models with manual and optimal hyperparameters identified by all search methods

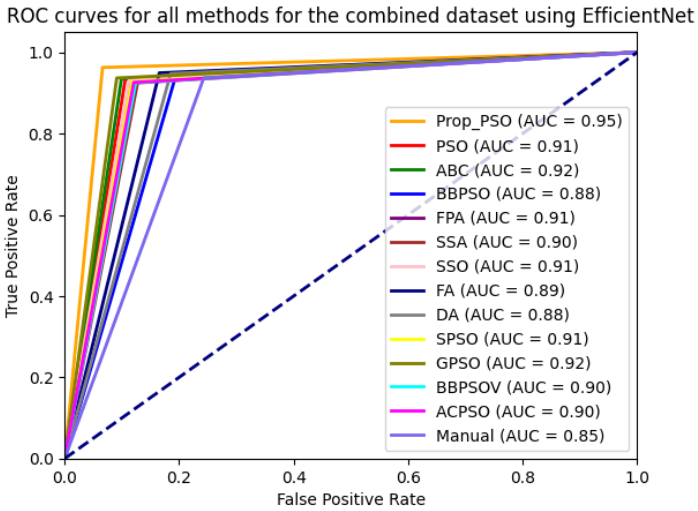

Fig. 10 ROC curves for the combined dataset using EfficientNet-B0 models with manual and optimal hyperparameters identified by all search methods

Figures 8-10 illustrate the Receiver Operating Characteristic (ROC) curves of the devised models by all search methods for the three datasets, respectively. The discriminative capabilities of the proposed PSO-optimized EfficientNet-B0 models are also indicated by the AUC score comparison, as depicted in Figures 8-10. Our optimized EfficientNet-B0 models obtain the best AUC scores in all test cases. The superiority of the proposed PSO-based EfficientNet-B0 models is further ascertained by the Wilcoxon rank sum test results based on the AUC scores (see the last columns in Tables 10-12), which are all lower than 0.05 for all three test sets. This indicates that our optimized EfficientNet-B0 models outperform those devised by other search methods with a statistical significance.

The mean optimized hyperparameters over 5 runs for EfficientNet-B0 using the sampled combined dataset are provided in Table 13. These yielded hyperparameters by different search methods are analysed to justify performance variations of the optimized networks. In addition, we also visualize the effects of different optimized dropout rates in Figure 11, which shows accuracy rates of the three test sets (in the y-axis) along with the dropout hyperparameters (in the x-axis) identified by each search method using the sampled combined training set. We use a specific shape and colour symbol to represent each search method.

As indicated in Table 13, the proposed PSO, GPSO and ABC-based EfficientNet-B0 models outperform those with learning configurations obtained by other search methods for the three test sets. As shown in Table 13 and Figure 11, these models are equipped with moderate mean learning rates and moderate or slightly higher mean dropout rates. Such settings are able to deploy steady magnitude updates to the learning mechanism and produce

Table 13 The mean results of the optimal hyperparameters identified using each search method for EfficientNet-B0

Model	Learning rate	Dropout	Size	No. of frames
Prop. PSO	0.0001810	0.5633	119	36
PSO	0.0003550	0.79	125	40
ABC	0.0001203	0.6512	121	35
BBPSO	0.0002273	0.1026	119	28
FPA	0.0002795	0.3566	117	35
SSA	0.0004570	0.2692	116	29
SSO	0.0003918	0.5266	118	38
FA	0.0004500	0.2454	114	32
DA	0.0005000	0.3635	115	32
SPSO	0.0002420	0.3326	116	33
GPSO	0.0001538	0.4671	126	37
BBPSOV	0.0002951	0.2985	115	34
ACPSO	0.0003626	0.3563	117	32
Manual	0.0001	0.3	112	30

efficient sparse network representations with effective discriminative feature learning capabilities to minimize redundancy. Significantly large or small settings of dropout rates are identified by PSO and BBPSO respectively as indicated in Figure 11. These may lead to the switching off of too many or too few neurons which may in turn result in the extraction of inadequate or noisy spatial features. Moreover, as illustrated in Table 13, large mean learning rates are produced by PSO, SSA, SSO, FA, DA and ACPSO, which may generate large learning magnitudes to cause fluctuations in loss space. A small learning rate is used in combination with a small dropout rate for the manual setting, resulting in inadequate gradient descent updates with redundant network topologies, limiting its performance.

As discussed in Section 4.1.2, moderate settings of image solutions and number of frames may lead to effective representations of video inputs while having an optimal computational cost. In contrast, significant large number of frames and image solutions may result in high computational cost as well as network overfitting by capturing noisy irrelevant details. In our experiments, 30-40 frames are recommended in most cases which are able to capture sufficient RGB and motion details as ascertained by the empirical results. Image resolutions of 115-126 are also mostly selected to balance between performance and computational cost.

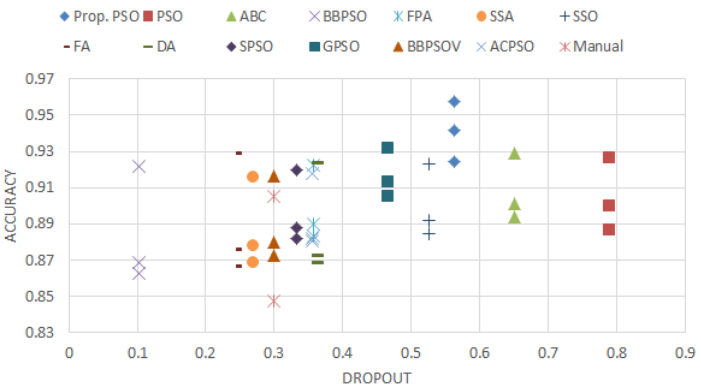

Fig. 11 Accuracy rates of the three test sets (in the y-axis) for optimized EfficientNet-B0 along with the dropout hyperparameters (in the x-axis) identified by each search method based on the sampled combined dataset (The results of the three test sets, i.e. the Celeb-DFv2, DFDC and combined test sets, for each search method are represented by a unique shape and colour symbol.)

4.2.2 Automated Hyperparameter Search for EfficientNet-GRU

We also employ the same experimental settings for hyperparameter search using the EfficientNet-GRU model. A set of 5 runs is performed for hyperparameter search for each search algorithm. The optimized settings obtained by each search method are used to establish the final models, which are trained using the training set of the combined dataset with a larger number of epochs. These optimized networks are then evaluated using test sets of Celeb-DFv2, DFDC and the combined datasets, respectively. Moreover, the Wilcoxon rank sum test is also performed based on the AUC scores over 5 runs to indicate the statistical significance of our optimized model over those with optimal settings yielded by other search methods. The detailed evaluation and statistical test results for our optimized EfficientNet-GRU models against other devised networks are provided in Tables 14-16.

As shown in Tables 14-16, the proposed PSO-based EfficientNet-GRU models obtain better performances than those of the counterparts generated by all the baseline search methods in terms of the all evaluation metrics (i.e. precision, recall, accuracy, AUC and Wilcoxon rank sum test results) for all three test sets. In addition, models with hyperparameters produced by ACPSO, GPSO, and SSO obtain better mean accuracy rates and AUC scores than the results of those with learning configurations selected by other baselines in most test cases. The ROC curves of the optimized EfficientNet-GRU models derived by different search methods for the three test sets are illustrated in Figures 12-14, where our optimized models show better AUC scores than those of the networks yielded by other search methods for all test scenarios. The significance of our optimized EfficientNet-GRU models is further ascertained by the

Table 14 Performance comparison for optimized EfficientNet-GRU models using the Celeb-DFv2 test set

Model	Acc.	Prec.	Recall	AUC	RS
Prop. PSO	0.9382	0.9208	0.9918	0.9141	n/a
PSO	0.9054	0.8839	0.9853	0.8691	7.94E-03
ABC	0.8784	0.8635	0.9676	0.8378	7.94E-03
BBPSO	0.8822	0.8681	0.9677	0.8434	7.94E-03
FPA	0.8938	0.8740	0.9794	0.8549	7.94E-03
SSA	0.8861	0.8707	0.9706	0.8477	7.94E-03
SSO	0.9131	0.9019	0.9735	0.8856	7.94E-03
FA	0.8977	0.8806	0.9765	0.8618	7.94E-03
DA	0.8996	0.8750	0.9882	0.8593	7.94E-03
SPSO	0.9116	0.8927	0.9853	0.8775	7.94E-03
GPSO	0.9189	0.9049	0.9794	0.8914	7.94E-03
BBPSOV	0.9093	0.9081	0.9588	0.8867	7.94E-03
ACPSO	0.9209	0.9052	0.9824	0.8929	7.94E-03
Manual	0.8417	0.8342	0.9471	0.7938	7.94E-03

Table 15 Performance comparison for optimized EfficientNet-GRU models using the DFDC test set

Model	Acc.	Prec.	Recall	AUC	RS
Prop. PSO	0.9517	0.9886	0.9566	0.9346	n/a
PSO	0.8849	0.9804	0.8881	0.8737	7.94E-03
ABC	0.8806	0.9784	0.8851	0.8649	7.94E-03
BBPSO	0.8833	0.9804	0.8863	0.8728	7.94E-03
FPA	0.8936	0.9800	0.8985	0.8765	7.94E-03
SSA	0.8931	0.9806	0.8973	0.8783	7.94E-03
SSO	0.9034	0.9841	0.9059	0.8947	7.94E-03
FA	0.8860	0.9741	0.8955	0.8531	7.94E-03
DA	0.8952	0.9676	0.9126	0.8349	7.94E-03
SPSO	0.9023	0.9834	0.9053	0.8919	7.94E-03
GPSO	0.9083	0.9854	0.9102	0.9017	7.94E-03
BBPSOV	0.8947	0.9788	0.9010	0.8728	7.94E-03
ACPSO	0.9370	0.9792	0.9493	0.8945	7.94E-03
Manual	0.8675	0.9703	0.8778	0.8321	7.94E-03

Wilcoxon rank sum statistical test results. As illustrated in the last columns in Tables 14-16, all the statistical test results are lower than 0.05 for the three test sets, which indicate the statistical superiority of our optimized networks against those yielded by other search methods.

Table 16 Performance comparison for optimized EfficientNet-GRU models using the combined test set

Model	Acc.	Prec.	Recall	AUC	RS
Prop. PSO	0.9695	0.989	0.9737	0.9620	n/a
PSO	0.9307	0.9818	0.9329	0.9268	7.94E-03
ABC	0.9239	0.9846	0.9217	0.9278	7.94E-03
BBPSO	0.9253	0.9839	0.9242	0.9274	7.94E-03
FPA	0.9304	0.9818	0.9325	0.9266	7.94E-03
SSA	0.9265	0.9835	0.9260	0.9275	7.94E-03
SSO	0.9381	0.9860	0.9383	0.9377	7.94E-03
FA	0.9253	0.9730	0.9350	0.9080	7.94E-03
DA	0.9342	0.9684	0.9509	0.9044	7.94E-03
SPSO	0.9348	0.9826	0.9372	0.9306	7.94E-03
GPSO	0.9363	0.9845	0.9372	0.9347	7.94E-03
BBPSOV	0.9310	0.9753	0.9397	0.9153	7.94E-03
ACPSO	0.9352	0.9793	0.9408	0.9249	7.94E-03
Manual	0.9126	0.9714	0.9206	0.8983	7.94E-03

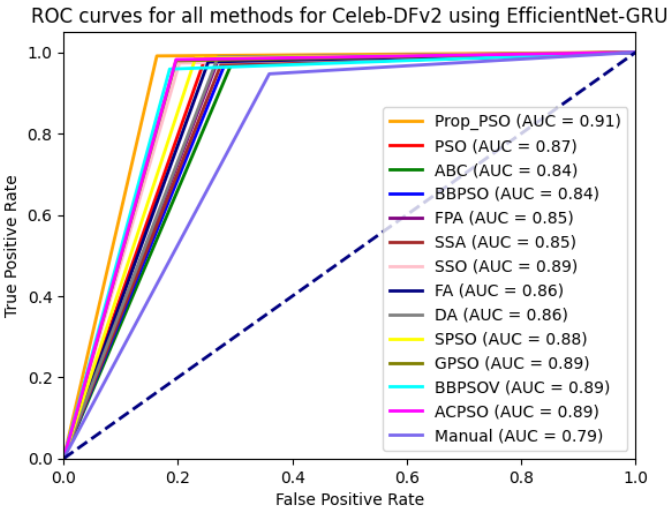

Fig. 12 ROC curves for Celeb-DFv2 using EfficientNet-GRU models with manual and optimal hyperparameters identified by all the search methods

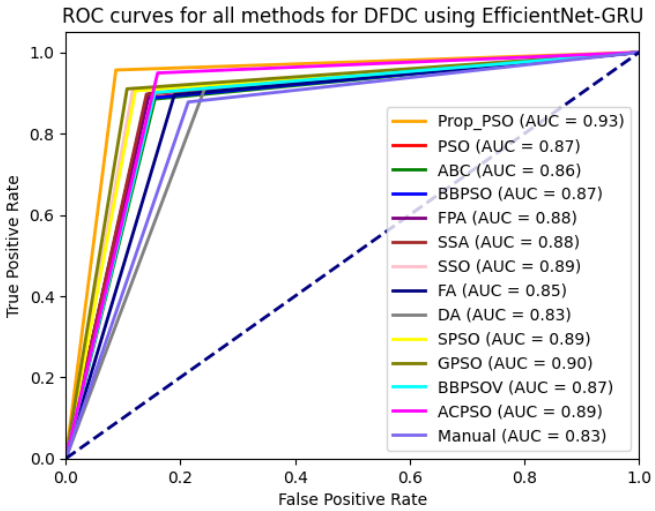

Fig. 13 ROC curves for DFDC using EfficientNet-GRU models with manual and optimal hyperparameters identified by all the search methods

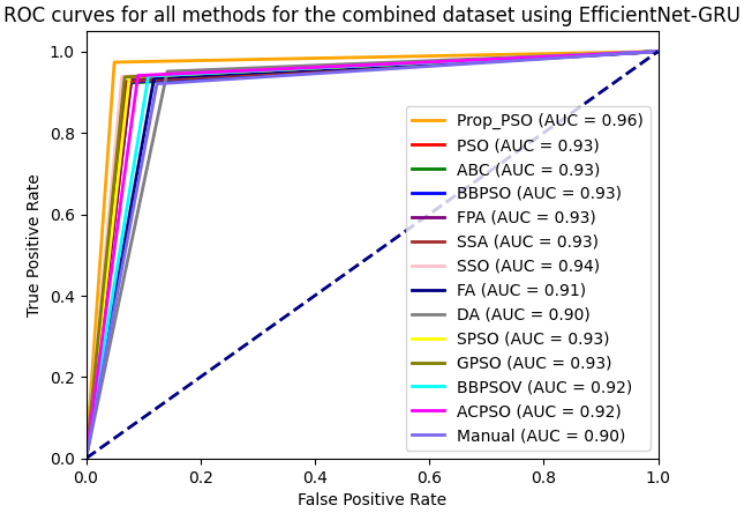

Fig. 14 ROC curves for the combined dataset using EfficientNet-GRU models with manual and optimal hyperparameters identified by all the search methods

The mean hyperparameters obtained by each search method over 5 runs using the sampled training and validation sets of the combined dataset are shown in Table 17. Some further analysis of these optimized hyperparameters is provided below to explain model performance variations.

Table 17 The mean results of the optimal hyperparameters identified using each search method for EfficientNet-GRU

Model	Learning rate	Dropout	Size	No. of Frames
Prop. PSO	0.0002333	0.5237	118	37
PSO	0.0001080	0.3832	117	37
ABC	0.0003520	0.6224	105	27
BBPSO	0.0003312	0.6897	113	30
FPA	0.0001108	0.3237	116	36
SSA	0.0003292	0.3816	106	29
SSO	0.0001848	0.4641	121	39
FA	0.0004300	0.3341	116	35
DA	0.0005000	0.3272	115	32
SPSO	0.0002499	0.3687	119	38
GPSO	0.0002258	0.4335	125	34
BBPSOV	0.0003010	0.6057	117	39
ACPSO	0.0001766	0.4478	123	35
Manual	0.0001	0.3	112	40

As indicated in Table 17 and Tables 14-16 pertaining to selected hyperparameter settings and detailed evaluation results respectively, the proposed PSO algorithm, ACPSO, GPSO and SSO have obtained comparatively moderate mean learning rate and dropout rate configurations over a set of 5 runs, in comparison with those obtained by other search methods. Such moderate mean learning rates show great efficiency in extracting knowledge in a new domain by deploying reasonable magnitudes of learning updates. The identified mean dropout rate settings reduce redundancy by switching off reasonable numbers of neurons, while generating effective discriminative video representations. Among the baseline methods, ABC, BBPSO, SSA, FA, DA, and BBPSOV-based networks are equipped with large learning rates, resulting in the employment of large magnitudes for the learning updates to cause instability. ABC, BBPSO, and BBPSOV also produce large mean dropout rates which lead to the elimination of large numbers of neurons, thus resulting in discarding important spatial-temporal cues. Moreover, smaller mean learning rates are identified by PSO and FPA, as well as adopted in the manual setting, which may yield insufficient momentum for network upgrading towards global optima, lowering network performance.

In particular, Figure 15 shows the accuracy rates of the EfficientNet-GRU models for three test sets (in the y-axis) with manual and optimal dropout rates identified by each search method (in the x-axis). As mentioned above, high accuracy rates are correlated with the moderate settings of dropout rates, which are preferred by the proposed PSO algorithm, ACPSO, GPSO and SSO. Significantly larger dropout rate configurations, such as those obtained by ABC, BBPSO, and BBPSOV, reduce the model performance by constraining network representations considerately, while excessively small settings of the dropout rates, recommended by FPA, FA, and DA, may lead to redundant network structures with limited flexibilities in tackling overfitting.

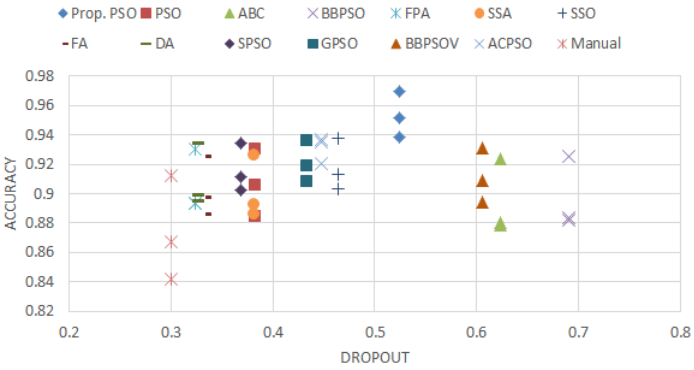

Fig. 15 Accuracy rates of the three test sets (in the y-axis) for optimized EfficientNet-GRU along with the dropout hyperparameters (in the x-axis) identified by each search method based on the sampled combined dataset (The results of the three test sets, i.e. the Celeb-DFv2, DFDC and combined test sets, for each search method are represented by a unique shape and colour symbol.)

Similar to the findings of the previous experiments using EfficientNet-B0, most search methods select 30-39 frames leading to the capture of sufficient spatial-temporal patterns using EfficientNet-GRU, while avoiding overfitting. Image resolutions of 113-125 are mostly identified to achieve reliable classification performance while maintaining efficient computational cost.

In short, when the hyperparameters obtained using the proposed PSO method are used in network evaluation, both optimized EfficientNet-B0 and EfficientNet-GRU models show improvement over those with manual and optimal settings yielded by other search methods. This is owing to the efficient search capabilities of the proposed PSO algorithm by integrating composite leaders and reinforcement learning-based search strategy selection in identifying optimal hyperparameters in a multi-dimensional complex search space with challenging high intra-class and low inter-class variations.

We analyze the search behaviours of each algorithm below. ABC explores the search space by following randomly selected leader individuals, therefore shows a slow convergence rate to reach global optimality. SSA simulates the salp chain behaviours by adopting the mean position of a neighbouring follower salp and the current individual for movement update. Owing to the adoption of neighbouring solution for search exploration, SSA is more likely to converge prematurely and requires a significant number of iterations to obtain competitive performance. Also a random threshold is used in SSA to determine the respective random walk action for updating the leading salp, instead of using a more informative selection scheme, therefore limiting its performance.

SSO generates a dynamic leader signal for the position update of each search agent by using the strongest and randomly selected vibration intensities, but the model only employs a single random walk mechanism for position update. FA employs neighbouring fitter solutions for search exploration while

DA adopts separation, alignment, cohesion, attraction and evading actions for movement update. But both algorithms use single search strategies for search space exploration. Similarly, PSO, BBPSO, SPSO with sine coefficients and GPSO with genetic operators also mainly employ monotonous search operations guided by the swarm leader for position update. When the single search actions in the aforementioned models become stagnant, there are no substitute search operations available to reactivate sudden movements of the search agents to mitigate premature stagnation.

To overcome the above limitations, BBPSOV and ACPSO adopt two or multiple search mechanisms to better manage local optima traps. For example, attractiveness and evading action are exploited in BBPSOV, while ACPSO employs three subswarms guided by the PSO operations with adaptive sine, circle and spiral coefficients respectively. FPA uses search actions led by either the randomly selected individuals or the swarm leader with Levy search coefficients. The multiple search actions in the above algorithms are mostly randomly selected or performed in sequential orders without the guidance of more informative selection principles.

In comparison with the above search algorithms, a reinforcement learning algorithm is employed in the proposed PSO variant to generate a more informed strategy to identify the optimal selection of different search actions for each particle. Such Q-learning based search deployment governed by Bellman equation empowers the search process with bespoke particle behaviours to explore the search space effectively while accelerating convergence. On top of it, search operations guided by multiple composite leaders yielded by distinctive nonlinear functions are also used to divert the search process if the action led by the swarm leader becomes stagnant. The above analysis has been further evidenced by the evaluation and statistical test results in our empirical studies.

4.3 Comparison with Other Hybrid Networks and 3D CNNs

We conduct performance comparison between our optimized EfficientNet-B0 and EfficientNet-GRU models and other networks including ResNet50-GRU, ResNet101-GRU, GoogLeNet-GRU, as well as 3D CNNs such as Inflated-3D (I3D), a Mixed Convolution Network (MC3), 3D ResNeXt101, and 3D ResNeXt50, using the Celeb-DFv2, DFDC, and combined datasets, respectively. These baseline state-of-the-art networks are selected because of their significant discriminative capabilities in video classification [67] [60] [9] [42]. Similar to our work, for the hybrid networks, i.e. ResNet50-GRU, ResNet101-GRU, and GoogLeNet-GRU, the respective CNN (ResNet50, ResNet101 and GoogLeNet) models are pretrained using ImageNet and their successive GRU models are trained from scratch using our combined deepfake training set.

In addition, for 3D CNNs, 3D convolutions instead of 2D convolutions are used for feature learning, except for MC3 where mixed convolutions are used.

Precisely, MC3 employs 3D convolutions in first two groups and 2D convolutions from group 3 onwards [68]. 3D ResNeXt101 and 3D ResNeXt50 are variants of 3D ResNet, which introduce a cardinality parameter to control the number of parallel paths within each residual block [69]. All the above 3D CNNs are pre-trained using a large human action dataset, i.e. Kinetics, consisting of 306,245 videos from 400 classes, then fine-tuned using the combined deepfake training set for real/manipulated video classification.

All the selected hybrid and 3D CNN baseline networks are equipped with the following learning settings, i.e. learning rate=0.0001, dropout rate=0.3, image size=112, and number of frames=40. Tables 18-20 show the detailed evaluation results, while Figures 16-18 illustrate respective ROC curves, for the Celeb-DFv2, DFDC, and combined test sets, respectively.

As indicated in Tables 18-20 and Figures 16-18, the proposed PSO-optimized EfficientNet-B0 and EfficientNet-GRU models show competitive performances as compared with those of the aforementioned three hybrid and four 3D CNN models, across datasets. The proposed optimizer employs multiple composite elite signals and Q-learning based search operation allocation with bespoke search behaviours of each particle, to boost model capabilities in tackling stagnation. Our optimized networks are thus equipped with better learning settings and show enhanced capabilities in spatial-temporal feature learning for video forgery classification. In addition, ResNet101-GRU shows better results than those of ResNet50-GRU and GoogLeNet-GRU, because of its more effective feature learning capabilities using deeper residual blocks. Among the 3D CNNs, I3D illustrates more discriminative capabilities by inflating all the filters and pooling kernels in a 2D architecture through the insertion of an additional temporal dimension, thus achieves the most reliable performance. It outperforms MC3, 3D ResNeXt101, and 3D ResNeXt50 for most test scenarios.

Table 18 Performance comparison with hybrid networks and 3D CNNs for the Celeb-DFv2 test set

Model	Acc.	Prec.	Recall	AUC
Prop. PSO-based Effnet-GRU	0.9382	0.9208	0.9912	0.9141
Prop. PSO-based Effnet	0.9247	0.9101	0.9824	0.8985
ResNet50-GRU	0.7876	0.7556	1.0000	0.6910
ResNet101-GRU	0.8494	0.8359	0.9588	0.7996
GoogLeNet-GRU	0.8012	0.7699	0.9941	0.7134
I3D	0.8494	0.8639	0.9147	0.8197
MC3	0.8514	0.8433	0.9500	0.8065
3D ResNeXt101	0.8378	0.8249	0.9559	0.7841
3D ResNeXt50	0.7703	0.7826	0.9000	0.7112
Manual Effnet-GRU	0.8417	0.8342	0.9471	0.7938
Manual Effnet	0.8263	0.8255	0.9324	0.7780

Table 19 Performance comparison with hybrid networks and 3D CNNs for the DFDC test set

Model	Acc.	Prec.	Recall	AUC
Prop. PSO-based Effnet-GRU	0.9517	0.9886	0.9566	0.9346
Prop. PSO-based Effnet	0.9414	0.9848	0.9487	0.9161
ResNet50-GRU	0.9289	0.9547	0.9658	0.8008
ResNet101-GRU	0.8127	0.9806	0.8050	0.8394
GoogLeNet-GRU	0.9164	0.9603	0.9450	0.8172
I3D	0.8969	0.9726	0.9095	0.8528
MC3	0.8865	0.9672	0.9028	0.8300
3D ResNeXt101	0.8882	0.9643	0.9077	0.8204
3D ResNeXt50	0.8806	0.9603	0.9028	0.8033
Manual Effnet-GRU	0.8675	0.9703	0.8778	0.8321
Manual Effnet	0.8475	0.9484	0.8759	0.7486

Table 20 Performance comparison with hybrid networks and 3D CNNs for the combined test set

Model	Acc.	Prec.	Recall	AUC
Prop. PSO-based Effnet-GRU	0.9695	0.9890	0.9737	0.9620
Prop. PSO-based Effnet	0.9576	0.9852	0.9628	0.9484
ResNet50-GRU	0.9209	0.9606	0.9422	0.8827
ResNet101-GRU	0.8865	0.9846	0.8755	0.9063
GoogLeNet-GRU	0.9339	0.9626	0.9567	0.8932
I3D	0.9369	0.9765	0.9459	0.9209
MC3	0.9393	0.9655	0.9603	0.9016
3D ResNeXt101	0.9470	0.9592	0.9769	0.8934
3D ResNeXt50	0.9120	0.9541	0.9379	0.8656
Manual Effnet-GRU	0.9126	0.9714	0.9206	0.8983
Manual Effnet	0.9049	0.9464	0.9372	0.8471

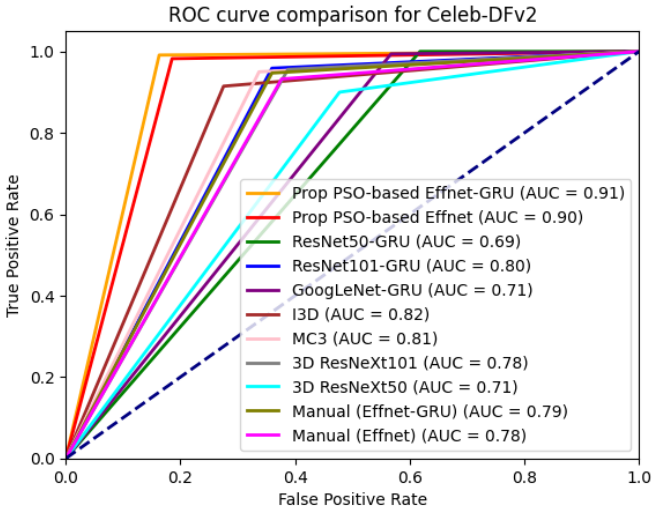

Fig. 16 ROC curve comparison between our optimized models and hybrid networks and 3D CNNs for the Celeb-DFv2 test set

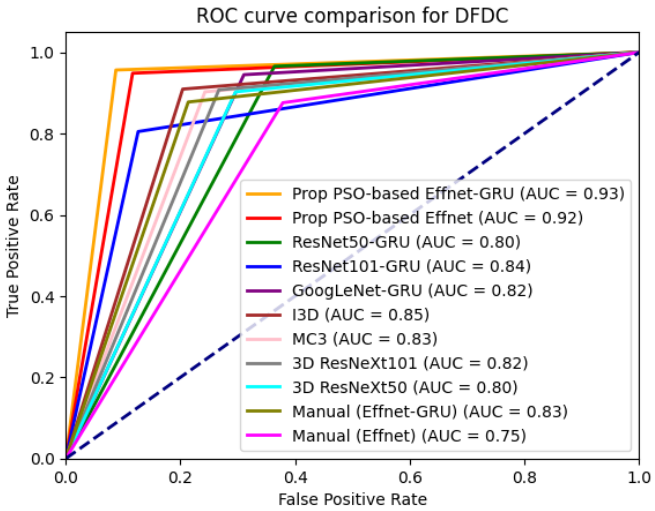

Fig. 17 ROC curve comparison between our optimized models and hybrid networks and 3D CNNs for the DFDC test set

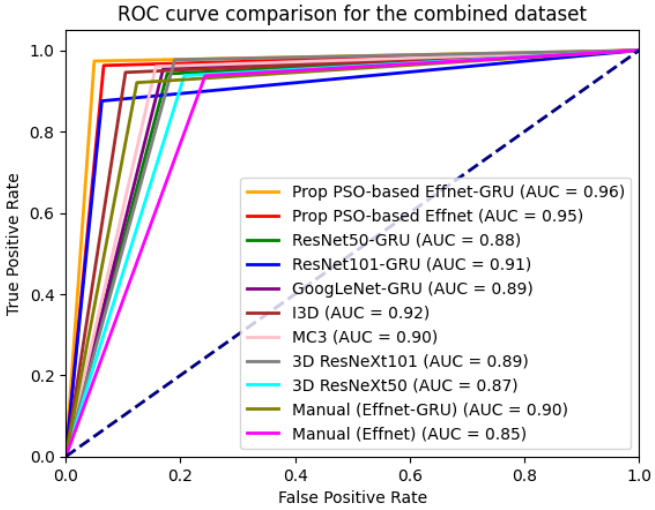

Fig. 18 ROC curve comparison between our optimized models and hybrid networks and 3D CNNs for the combined test set

The confusion matrices of the proposed PSO-optimized EfficientNet-B0 and EfficientNet-GRU models with respect to the Celeb-DFv2, DFDC and combined test sets are provided in Figures 19 and 20, respectively. The built-in scikit-learn packages in the Python library are used to generate these confusion

matrix results, along with those for other evaluation metrics (i.e. accuracy, precision, recall and AUC scores) in this research.

	Fake	Real
Fake	334	6
Real	33	145

	Fake	Real
Fake	1552	84
Real	24	182

	Fake	Real
Fake	2667	103
Real	40	565

Fig. 19 Confusion matrices of the proposed PSO-optimized EfficientNet-B0 for the Celeb-DFv2 (left), DFDC (middle) and combined (right) test sets, respectively

	Fake	Real
Fake	337	3
Real	29	149

	Fake	Real
Fake	1565	71
Real	18	188

	Fake	Real
Fake	2697	73
Real	30	575

Fig. 20 Confusion matrices of the proposed PSO-optimized EfficientNet-GRU for the Celeb-DFv2 (left), DFDC (middle) and combined (right) test sets, respectively

Since all the search methods employ the same number of function evaluations for hyperparameter search, with deep learning-based fitness function evaluation as the most costly process, these methods have a similar overall cost for optimal parameter selection at the training stage. We provide the average cost of each algorithm with one function evaluation over 5 runs for computational efficiency comparison. Such a mean cost for each trial is calculated by averaging the cost for hyperparameter search by the number of function evaluations performed. This includes the cost of dedicated search operations embedded in each algorithm along with one fitness evaluation of the recommended hyperparameters using either EfficientNet-B0 or EfficientNet-GRU. The cost variations are mainly caused by different search principles operated in the search methods. Table 21 depicts the detailed cost comparison.

We conduct the computational cost comparison using a NVIDIA RTX 3090 consumer GPU. As indicated in Table 21, the proposed model shows moderate mean computational costs per function evaluation over 5 runs for both networks. ABC, SSA, FA, FPA, and BBPSO have lower mean computational costs owing to their comparatively simpler search strategies by following randomly selected (ABC), neighbouring (SSA and FA) and global best (BBPSO and FPA) solutions, respectively. DA employs a search action combining separation, alignment, cohesion, attraction and evading mechanisms, also showing relatively light computational costs. SSO performs vibration intensity-based actions with dynamic leader generation, resulting in slightly higher but reasonable costs. In contrast, BBPSOV and ACPSO have the highest computational costs due to diverse embedded search actions, e.g. evading/attraction-inspired mechanisms in BBPSOV, and three subswarms with adaptive sine, circle and

Table 21 The mean computational costs (in seconds) of each search method with one function evaluation for hyperparameter search

Model	EfficientNet-B0	EfficientNet-GRU
Prop. PSO	268.1408	275.4205
PSO	184.0033	205.9014
ABC	80.0875	83.0909
BBPSO	132.3337	160.4870
FPA	147.4523	180.2467
SSA	77.3659	136.9845
SSO	184.8069	188.9568
FA	142.7953	153.1096
DA	149.0886	173.9974
SPSO	238.7975	265.3025
GPSO	243.6370	257.8900
BBPSOV	269.7608	279.9356
ACPSO	360.4934	378.6980

spiral search coefficients in ACPSO. The proposed model, GPSO, and SPSO show moderate costs because of the deployment of reinforcement learning-based action selection and hybrid leader generation in our model, crossover and mutation operations in GPSO, and adaptive sine-based search coefficients in SPSO, respectively. The optimal configurations identified by each method in the training process are used to construct the final network at the test stage for performance comparison.

We also compare our optimized networks with other existing studies for Celeb-DFv2 and DFDC datasets. The selected existing studies were evaluated using the official Celeb-DFv2 test set and the DFDC subset, respectively, as performed in our experiments. But since these related studies were trained using a variety of deepfake training databases for evaluating both datasets, they are used for loose performance comparison. Tables 22 and 23 illustrate the performance comparison with state-of-the-art existing studies using the Celeb-DFv2 and DFDC datasets, respectively.

We used the official split to evaluate the Celeb-DFv2 dataset. Specifically, for the official Celeb-DFv2 test set, the selected existing studies shown in Table 22 obtained accuracy rates ranging from 0.8074 to 0.8989 and AUC scores ranging from 0.696 to 0.9003. Our transfer learning using EfficientNet-B0 with the proposed PSO-based hyperparameter fine-tuning obtained a competitive benchmark with an accuracy rate of 0.9247 and an AUC score of 0.8985, while EfficientNet-GRU with the proposed PSO-based hyperparameter selection achieved a better accuracy rate of 0.9382 and a better AUC result of 0.9141. Both of our models outperform most of these related studies by a sufficient margin.

Table 23 shows the comparison between our optimized networks and existing studies for the DFDC test set. Again our optimized EfficientNet-GRU and EfficientNet-B0 models with the proposed PSO-based hyperparameter fine-tuning obtain more reliable performance in comparison with those of existing

Table 22 Performance comparison for the Celeb-DFv2 test set

Model	Methodology	Accuracy	AUC
Hu [13]	Two Stream	0.8074	-
Demir and Ciftci [14]	Biological Signals (sequence-based)	0.8576	-
Demir and Ciftci [14]	Biological Signals (video-based)	0.8835	-
Kandasamy et al. [70]	VGG19	0.8843	-
Kandasamy et al. [70]	ResNet	0.8932	-
Rossler et al. [3]	XceptionNet-Max	0.8989	-
Haliassos et al. [71]	LipForensics	-	0.824
Liu et al. [72]	SPSL(Xception as the backbone)	-	0.7688
Wang et al. [62]	MC-LCR	-	0.7161
Zheng et al. [59]	Temporal Coherence	-	0.869
Wang et al. [73]	CNN-aug	-	0.756
Nguyen et al. [74]	Multi-task	-	0.757
Chai et al. [75]	PatchForensics	-	0.696
Masi et al. [76]	Two-branch LSTM	-	0.7665
Tolosana et al. [77]	Facial element extraction	-	0.836
Zhao et al. [61]	PCL + I2G (ResNet-34 as the backbone)	-	0.9003
This research	Prop. PSO-based EfficientNet-GRU	0.9382	0.9141
This research	Prop. PSO-based EfficientNet	0.9247	0.8985

studies. Specifically, the EfficientNet-B0 model with the proposed PSO-based hyperparameter selection achieves a competitive mean accuracy rate of 0.9414 and AUC score of 0.9161, and our optimized EfficientNet-GRU obtains a better mean accuracy rate of 0.9517 and a better AUC score of 0.9346.

In short, owing to the efficient search capabilities of the proposed PSO model guided by composite leaders and optimized search strategies governed by the fitness evaluations, our optimized networks outperform existing studies for both Celeb-DFv2 and DFDC datasets and show great efficiency in tackling challenging manipulated video classification tasks. They can be deployed as effective substitute methods for deepfake content identification. In addition, our work also highlights the importance of hyperparameter selection in deep learning networks and the potential use of evolutionary algorithms in such tasks for video deepfake classification.

5 Visualization Using Gradient-weighted Class Activation Mapping

Various evolutionary and sensitivity-based methods are proposed for feature selection [83] [19] [84] [85]. Owing to the large feature dimensions extracted from multiple video frames, in this research, we visualize the contributions of different convolutional and spatial features using class-discriminative heatmaps, to indicate the effectiveness of our optimized networks for discriminative feature extraction. Specifically, we generate heatmaps using gradient-weighted class activation mapping (Grad-CAM) [86] for both proposed EfficientNet serialized with GRU, as well as the proposed PSO-optimized

Table 23 Performance comparison for the DFDC test set

Model	Methodology	Accuracy	AUC
Li [78]	XceptionNet + MIL	0.8378	-
Li [78]	XceptionNet + S-MIL-T	0.8511	-
Zhang et al. [79]	TD-3DCNN	0.8264	-
Wang et al. [62]	MC-LCR	0.702	0.7134
Güera and Delp [80]	RNN	0.6242	0.669
Hu et al. [81]	FInfer	0.6945	0.7039
Li et al. [82]	Face X-ray	-	0.655
Zheng et al. [59]	Temporal Coherence	-	0.74
Wang et al. [73]	CNN-aug	-	0.721
Shiohara and Yamasaki [60]	Self-blended data synthesis	-	0.7242
Song et al. [67]	CD-Net (Xception as the backbone)	-	0.783
Zhao et al. [61]	PCL + I2G (ResNet-34 as the backbone)	-	0.6752
This research	Prop. PSO-based EfficientNet-GRU	0.9517	0.9346
This research	Prop. PSO-based EfficientNet	0.9414	0.9161

EfficientNet to indicate their effectiveness in feature learning. In particular, as indicated earlier, in the proposed EfficientNet-GRU model, all the convolutional layers of the ImageNet pre-trained EfficientNet are slightly fine-tuned using the combined training set with a small number of epochs (e.g. 5 epochs) with the attempt to improve their discriminative feature learning capabilities in the target domain.

First of all, since this research focuses on detection and classification of face swapping and facial re-enactment, MTCNN-based facial cropping is performed to extract the facial regions and eliminate background noise. Besides that, for both EfficientNet-B0 and EfficienNet-GRU models, as discussed in Sections 4.2.1 and 4.2.2, the proposed PSO and other search methods are used to optimize the number of video frames and image resolution sizes, along with learning and dropout rates, to maintain optimal cost while eliminating irrelevant noisy features to avoid overfitting.

To indicate the effectiveness of the proposed PSO-based EfficientNet model and the EfficientNet embedded in EfficientNet-GRU for feature learning, Grad-CAM [86] heatmaps with different color schemes are used to visualize the importance of the extracted features from the cropped facial regions. Grad-CAM first calculates the gradient of the prediction score for a target class with respect to the extracted feature maps in the last convolutional layer. Then the global average pooling is applied to gradients calculated above to generate weightings of respective feature maps for a target class. The yielded importance weightings subsequently multiply with respective feature maps. A summation operation followed by a ReLU activation function is performed on these weighted results to produce the Grad-CAM heatmaps. These localization maps indicate feature significance to class prediction using different colour schemes with deep red indicating the most significant/relevant characteristics and deep blue as the least influential factors for categorization. Therefore, such class-discriminative heatmaps are used in this research to

visualize which image regions have the most influence to synthetic/original video classification. In comparison with class activation mapping (CAM), the Grad-CAM method can be applied to any CNN architectures even without re-training [86]. In addition, as mentioned above, in this research, to improve discriminative feature learning, we fine-tune all the convolutional layers of the ImageNet pre-trained EfficientNet-B0 embedded in the proposed EfficientNet-GRU model using the combined training set with a small number of epochs (i.e. 5 epochs), before feature extraction. Therefore, we generate Grad-CAM heatmaps for this EfficientNet-B0 model with light-weight fine-tuning embedded in the EfficientNet-GRU, to indicate its effectiveness in feature learning. Besides that, we also generate Grad-CAM heatmaps for the proposed PSO-optimized EfficientNet-B0 to demonstrate its capabilities in discriminative feature representation.

Figure 21 shows example original video frames (the first row), respective manipulated video frames (the second row), and Grad-CAM heatmaps extracted from lightly-tuned EfficientNet-B0 embedded in the EfficientNet-GRU (the third row), as well as the heatmaps generated by the proposed PSO-optimized EfficientNet-B0 (the fourth row), with respect to the manipulated image frames in the second row. Since these example video frames are taken from Celeb-DFv2, the face swap attack is performed in these deepfake examples. The inspection of the synthetic image frames in the second row in Figure 21 against the real image frames in the first row reveals the presence of vagueness and blurry in the eye, nose, mouth or overall facial regions, as well as shape alterations of eye, nose, and mouth elements. As indicated in existing studies, inconsistent shadowing/lighting/colour tone over faces, unnatural/inconsistent teeth/mouth/eye movements, misaligned teeth, distortions in eyebrows, facial hair and facial borders, double chins, non-circled pupils, and other spatial inconsistency, are key factors for identifying manipulated images against real ones.

As visualized by Grad-CAM heatmaps shown in the third and fourth rows in Figure 21, the most significant factors extracted by both the lightly-tuned EfficientNet (embedded in EfficientNet-GRU) and the proposed PSO-optimized EfficientNet model are mostly derived from these facial abnormality regions such as eye and mouth/teeth regions. These dominating features are represented by deep red heatmaps, emphasizing their importance to manipulated class prediction. As an example, the heatmaps extracted using the lightly-tuned EfficientNet model (embedded in EfficientNet-GRU) in the third row show high correlations to those manipulated facial regions such as blurred eye regions, and inconsistent shadowing/lighting/colour tone over faces. Built upon this, as shown in the last row in Figure 21, the proposed PSO-optimized EfficientNet-B0 method with substantial re-training is able to even better capture such abnormalities and strengthen the extraction of such important characteristics. For instance, in most cases, significant factors with respect to shadowy eye regions, unnatural pupils and iris borders, are extracted by our optimized network. In addition, distortions in facial borders, misaligned teeth,

and double chins, which also have vital influence to authenticity classification, are identified as well.

As indicated in the results in Figure 21, both the EfficientNet with light-weight fine-tuning and the proposed PSO-optimized EfficientNet-B0 model show great efficiency in capturing important discriminative features playing significant roles in synthetic video identification. In addition, the proposed PSO-optimized EfficientNet-B0 network adopts such extracted heatmaps for frame-level fake/real image classification. A mean average ensemble scheme is used to combine the frame-level prediction based on a sequence of frames to determine final video classification outcome. Moreover, in the proposed EfficientNet-GRU model, the most important spatial features in the heatmaps extracted using EfficientNet-B0 with light-weight fine-tuning are further strengthened by combining a sequence of such class-discriminative Grad-CAM maps from multiple frames. Such a sequence of discriminative heatmaps is then passed on to the GRU model for temporal feature learning to inform final video authenticity identification.

Besides the above, as indicated in Sections 4.2.1 and 4.2.2, the proposed PSO model is also used to optimize the number of frames and image resolution settings for both EfficientNet-GRU and EfficientNet-B0 networks, in order to extract salient features without capturing too much noisy irrelevant/contradictory details to avoid overfitting. As an example, for EfficientNet-B0, a selection of 30-40 frames and image resolutions of 115-126 is recommended by the proposed PSO and other search methods owing to a good balance between performance and computational cost. Similarly for EfficientNet-GRU, the proposed PSO model and other search methods recommend the optimal number of 30-39 frames with image resolutions of 113-125 for model training and test to better tackle overfitting.

These optimized frame and image resolution settings, along with the effectiveness of the extracted spatial-temporal features by the EfficientNet-B0 with light-weight fine-tuning and proposed PSO-optimized EfficientNet-B0 models as evidenced in Grad-CAM maps, lead to the capture of discriminative RGB and motion cues to achieve reliable video classification. Our optimized EfficientNet-B0 networks also outperform those generated by other search methods as indicated by the empirical and statistical test results in Tables 10-12. The efficiency of our optimized EfficientNet-GRU models is also ascertained by the experimental and statistical test results as shown in Tables 14-16.

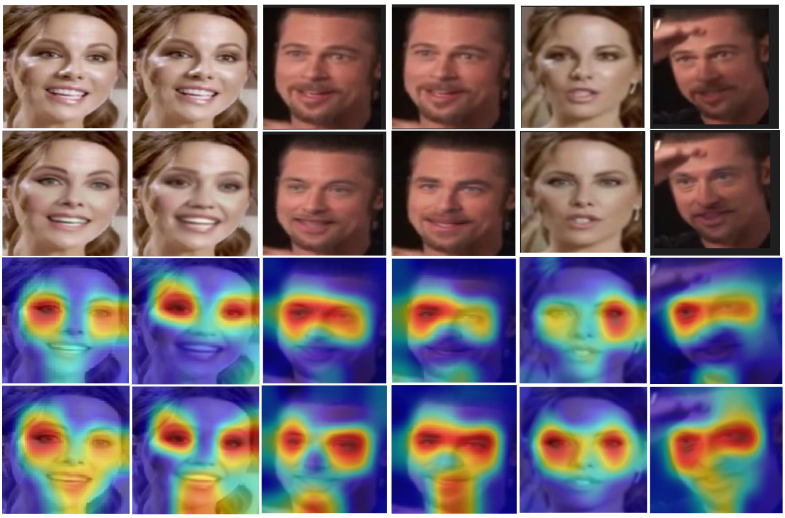

Fig. 21 Example Grad-CAM heatmaps generated for manipulated samples (row 1: the original video frames, row 2: the respective manipulated frames, row 3: Grad-CAM heatmaps generated by the EfficientNet model with light-weight fine-tuning for the manipulated frames (in row 2), and row 4: Grad-CAM heatmaps generated using the proposed PSO-optimized EfficientNet model for the manipulated frames (in row 2)

6 Uncertainty Analysis

To indicate model effectiveness, we also conduct uncertainty analysis. We employ the Monte Carlo dropout (MCD) method for uncertainty quantification in this research. Specifically, we employ the MCD method to measure epistemic uncertainty, which is usually caused by the lack of training data. In other words, with more training data, such model uncertainty can be reduced. Before introducing MCD, we briefly discuss the traditional dropout method, which is only applied in the training stage by switching off some randomly selected neurons. And there is no dropout operation applied during test with all neurons enabled. The dropout operation provides flexibility in model training, thus helps tackle overfitting. In contrast, in the MCD method, the dropout is enabled during testing. This results in different dropout masks to be deployed during the forward passes for result calculation. These generated new architectures can be regarded as Monte Carlo samples. As such, by using dropout during testing, each test sample will be evaluated using different model architectures and the result distributions of these test samples are used in this research for computing different uncertainty metrics.

Since we focus on a classification problem for video authenticity identification, as suggested by existing studies [87][88][89], we employ the predictive entropy and mutual information for uncertainty analysis. Equations 10-11 define the formulae of the predictive entropy [87].

$$Entropy = -\sum_{c=1}^{C}(u_c)log(u_c) \tag{10}$$

$$u_c = \frac{1}{T}\sum_{i=1}^{T}P_c^i \tag{11}$$

where C denotes the number of predicted classes with u_c representing the class-wise mean prediction probability. In addition, T denotes the number of MCD forward passes employed in our experiments.

The mutual information is formulated in Equation 12.

$$MI = -\sum_{c=1}^{C}(u_c)log(u_c) + \frac{1}{T}\Sigma_{c=1}^{C}\Sigma_{t=1}^{T}P((y \in c)\Big|(x, w_t))log(P((y \in c)\Big|(x, w_t))) \tag{12}$$

where MI indicates mutual information, while $P((y \in c)|(x, w_t))$ represents the softmax score for the input sample x belonging to class c with model parameters w_t.

We employ all test video samples, i.e. 3,375 videos, from the combined test set for uncertainty analysis. Each sample is tested $T = 20$ times using the MCD method. Table 24 shows the mean predictive entropy (MPE) and mutual information scores over all test videos with respect to the EfficientNet and EfficientNet-GRU models with optimized settings obtained using the proposed PSO model, respectively. Specifically, MPE_fake, MPE_real, and MPE_all in Table 24 denote the mean predictive entropy scores for the fake, real and both classes, with Mutual Info indicating the mean mutual information score for both classes, over all test samples.

As indicated in Table 24, both optimized networks obtain low entropy and mutual information scores, which indicate that the models have high certainty about predictions. In addition, both mean predictive entropy and mutual information scores of the proposed PSO-based EfficientNet-GRU method are lower than those of the proposed PSO-based EfficientNet model. This shows that the optimized EfficientNet-GRU model has better discriminative capabilities for distinguishing synthetic and real videos with lower uncertainty.

For each optimized network, the mean predictive entropy scores for both manipulated and real classes are also provided in Table 24. The mean predictive entropy scores for the manipulated class are lower than those of the genuine video class for both networks. Figures 22 and 23 also show the detailed uncertainty estimation distributions in terms of the predictive entropy for both fake

and real classes for the PSO-devised EfficientNet-GRU and EfficientNet models, respectively. As indicated in Table 24 and Figures 22-23, the manipulated videos are classified with higher certainty than those of the original videos by both optimized networks. The combined dataset construction may help explain the above observations. Since in the employed deepfake datasets (i.e. Celeb-DFv2 and DFDC), there are usually much larger numbers of synthetic videos than those of the genuine ones, real video samples from the YouTube Faces Database are also borrowed to increase the real class sample sizes and balance class distributions when constructing the combined dataset. Therefore, the trained networks have been encountered with a variety of manipulated instances and show comparatively lower uncertainty in recognizing fake videos in comparison with the original ones.

Overall, the mean predictive entropy and mutual information scores indicate the effectiveness of both of the proposed PSO-optimized networks for classifying manipulated and real videos with reasonably high certainty.

Table 24 The mean predictive entropy and mutual information scores for the proposed PSO-optimized networks

Model	MPE_fake	MPE_real	MPE_all	Mutual Info
Prop. PSO-based Effnet-GRU	0.06335	0.10060	0.16400	0.05633
Prop. PSO-based Effnet	0.07149	0.11009	0.18159	0.06355

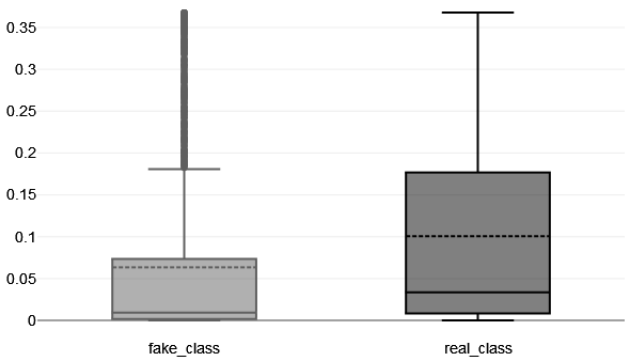

Fig. 22 Uncertainty estimation distributions in terms of the predictive entropy for both manipulated and real classes for the proposed PSO-optimized EfficientNet-GRU model using the combined test set

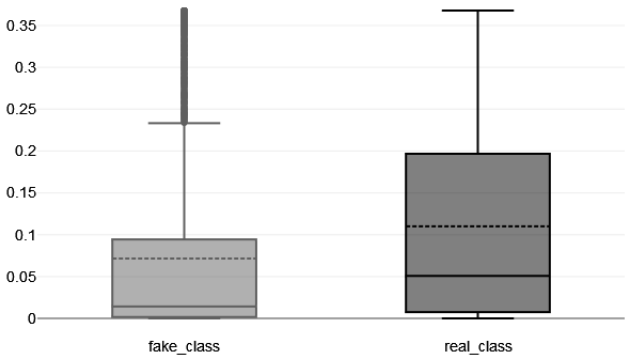

Fig. 23 Uncertainty estimation distributions in terms of the predictive entropy for both manipulated and real classes for the proposed PSO-optimized EfficientNet-B0 model using the combined test set

7 Evaluation Using Benchmark Functions

To further indicate the effectiveness of the proposed PSO algorithm, we employ unimodal and multimodal benchmark functions with varied search spaces and artificial landscapes for evaluation. Multimodal functions such as Rastrigin, Griewank, Ackley, and Powell, as well as unimodal functions including Rotated Hyper-Ellipsoid (Rothyp), Dixon-Price (Dixon), Sphere, Rosenbrock, Zakharov, Sum of Different Powers (Sumpow), and Sum Squares (Sumsqu), are evaluated in our experiments. The experiments are conducted using a maximum number of function evaluations of 25,000 (population=50 and iterations=500), and a dimension of 30. This maximum number of function evaluations (i.e. 25,000) is conducted by all search methods to ensure a fair comparison. The mean results along with maximum, minimum and standard deviation performances over a set of 30 runs for solving these benchmark functions are presented in Table 25. The Wilcoxon rank sum test results shown in Table 26 are used to indicate the significance of our results against those of the baseline methods.

As shown in Table 25, our model outperforms all the baseline methods for 9 out of 11 benchmark functions, while SSA and ABC obtain the best results for Griewank and Rosenbrock, respectively, with the proposed model achieving the second best results for these two test functions. The statistical superiority of the proposed model is also further evidenced in the rank sum test results shown in Table 26. Specifically, our model obtains statistically better results than those of the baseline methods for most test functions, except that SSA and ABC obtain statistically better results for Griewank and Rosenbrock than those of the proposed model, respectively.

Table 25 Evaluation results for the benchmark functions with dimension=30

		Prop_PSO	PSO	ABC	SSA	SSO	FA	DA	BBPSO	FPA	BBPSOV	SPSO	GPSO	ACPSO
Ackley	mean	**7.43E-15**	1.09E+01	5.91E-02	2.74E+00	2.13E+01	3.02E-02	1.93E+01	1.17E+01	1.22E+01	1.95E+01	1.30E+01	1.84E+01	1.12E+01
	min	**4.00E-15**	9.22E+00	5.91E-02	2.74E+00	2.13E+01	3.02E-02	1.93E+01	6.62E-01	1.09E+01	1.95E+01	1.30E+01	1.84E+01	4.21E+00
	max	**7.55E-15**	1.10E+01	5.91E-02	2.74E+00	2.13E+01	3.02E-02	1.93E+01	1.77E+01	1.42E+01	1.95E+01	1.30E+01	1.84E+01	1.79E+01
	std	6.49E-16	3.22E-01	**0.00E+00**	4.52E-16	3.61E-15	**0.00E+00**	**0.00E+00**	3.62E+00	7.78E-01	3.61E-15	7.23E-15	1.45E-14	5.21E+00
Dixon	mean	**1.29E-01**	1.26E+05	6.67E-01	7.29E-01	6.78E+06	1.41E+00	1.18E+03	4.30E+04	4.92E+03	2.09E+06	1.89E+04	3.58E+05	1.54E+05
	min	**1.29E-01**	1.09E+05	6.67E-01	7.29E-01	6.78E+06	1.41E+00	1.18E+03	6.67E-01	1.27E+03	2.09E+06	1.89E+04	3.58E+05	1.21E+00
	max	**1.29E-01**	1.27E+05	6.67E-01	7.29E-01	6.78E+06	1.41E+00	1.18E+03	3.52E+05	8.57E+03	2.09E+06	1.89E+04	3.58E+05	6.86E+05
	std	5.65E-17	3.24E+03	2.00E-11	5.65E-16	1.89E-09	2.26E-16	**0.00E+00**	1.05E+05	1.83E+03	9.47E-10	1.11E-11	5.92E-11	2.28E+05
Griewank	mean	2.28E-03	3.47E+00	2.94E-03	**6.25E-05**	1.30E+03	5.08E-03	9.40E+00	1.81E+01	2.91E+01	5.02E+02	6.12E+01	2.80E+02	4.66E+01
	min	2.28E-03	3.20E+00	2.94E-03	**6.25E-05**	1.30E+03	5.08E-03	9.40E+00	2.22E-03	1.84E+01	5.02E+02	6.12E+01	2.80E+02	2.53E-02
	max	2.28E-03	3.48E+00	2.94E-03	**6.25E-05**	1.30E+03	5.08E-03	9.40E+00	9.05E+01	3.70E+01	5.02E+02	6.12E+01	2.80E+02	3.18E+02
	std	1.76E-18	5.09E-02	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	5.42E-15	3.67E+01	5.29E+00	2.31E-13	4.34E-14	**0.00E+00**	1.04E+02
Rastrigin	mean	**8.21E-04**	1.19E+02	6.07E+00	5.77E+01	6.91E+02	3.69E+01	9.24E+01	1.31E+02	2.00E+02	4.07E+02	2.21E+02	3.54E+02	2.86E+02
	min	**8.21E-04**	7.14E+01	6.07E+00	5.77E+01	6.91E+02	3.69E+01	9.24E+01	3.19E+01	1.66E+02	4.07E+02	2.21E+02	3.54E+02	1.31E+02
	max	**8.21E-04**	1.21E+02	6.07E+00	5.77E+01	6.91E+02	3.69E+01	9.24E+01	2.21E+02	2.27E+02	4.07E+02	2.21E+02	3.54E+02	3.58E+02
	std	**3.31E-19**	9.01E+00	9.03E-16	3.61E-14	3.47E-13	2.17E-14	4.34E-14	5.15E+01	1.11E+01	5.78E-14	1.45E-13	1.73E-13	5.69E+01
Rothyp	mean	**1.05E-273**	1.18E+04	1.29E-04	1.09E+01	9.80E+05	1.64E+00	1.62E+04	2.42E+04	1.40E+04	3.88E+05	2.86E+04	1.98E+05	1.75E+04
	min	**1.01E-273**	6.31E+02	1.29E-04	1.09E+01	9.80E+05	1.64E+00	1.62E+04	2.34E-04	8.45E+03	3.88E+05	2.86E+04	1.98E+05	3.02E-05
	max	**2.16E-273**	1.22E+04	1.29E-04	1.09E+01	9.80E+05	1.64E+00	1.62E+04	1.14E+05	1.76E+04	3.88E+05	2.86E+04	1.98E+05	1.96E+05
	std	**0.00E+00**	2.11E+03	5.51E-20	3.61E-15	1.18E-10	**0.00E+00**	9.25E-12	3.36E+04	2.13E+03	1.18E-10	1.85E-11	8.88E-11	5.34E+04
Rosenbrock	mean	2.36E+01	1.13E+04	**7.35E-01**	2.23E+02	6.21E+06	2.87E+01	5.37E+03	5.93E+04	6.13E+03	1.65E+06	2.86E+04	3.12E+05	2.48E+04
	min	2.33E+01	2.32E+03	**7.35E-01**	2.23E+02	6.21E+06	2.87E+01	5.37E+03	9.59E+00	2.20E+03	1.65E+06	2.86E+04	3.12E+05	6.00E+00
	max	2.36E+01	1.16E+04	**7.35E-01**	2.23E+02	6.21E+06	2.87E+01	5.37E+03	2.23E+05	1.04E+04	1.65E+06	2.86E+04	3.12E+05	2.66E+05
	std	5.07E-02	1.69E+03	**0.00E+00**	5.78E-14	9.47E-10	1.08E-14	**0.00E+00**	6.23E+04	2.07E+03	**0.00E+00**	1.48E-11	**0.00E+00**	6.94E+04
Sphere	mean	**1.10E-275**	1.28E+00	3.97E-07	9.52E-11	3.78E+02	2.02E-03	3.38E+00	8.74E+00	8.55E+00	1.83E+02	1.89E+01	1.22E+02	1.32E+01
	min	**4.63E-277**	7.95E-01	3.97E-07	9.52E-11	3.78E+02	2.02E-03	3.38E+00	7.45E-08	6.04E+00	1.83E+02	1.89E+01	1.22E+02	2.83E-07
	max	**3.16E-274**	1.30E+00	3.97E-07	9.52E-11	3.78E+02	2.02E-03	3.38E+00	5.24E+01	1.14E+01	1.83E+02	1.89E+01	1.22E+02	1.00E+02
	std	**0.00E+00**	9.21E-02	1.62E-22	**0.00E+00**	5.78E-14	**0.00E+00**	1.36E-15	1.59E+01	1.23E+00	5.78E-14	**0.00E+00**	**0.00E+00**	2.96E+01
Sumpow	mean	**0.00E+00**	2.84E-07	6.04E-14	4.61E-07	5.32E+00	2.24E-07	6.52E-05	1.85E-19	8.88E-05	4.43E-01	3.61E-04	2.08E-02	3.02E-02
	min	**0.00E+00**	1.70E-07	6.04E-14	4.61E-07	5.32E+00	2.24E-07	6.52E-05	4.77E-30	7.65E-06	4.43E-01	3.61E-04	2.08E-02	6.53E-22
	max	**0.00E+00**	2.88E-07	6.04E-14	4.61E-07	5.32E+00	2.24E-07	6.52E-05	3.96E-18	2.70E-04	4.43E-01	3.61E-04	2.08E-02	9.86E-02
	std	**0.00E+00**	2.15E-08	**0.00E+00**	1.62E-22	**0.00E+00**	**0.00E+00**	1.38E-20	7.28E-19	6.93E-05	**0.00E+00**	1.65E-19	1.06E-17	3.27E-02
Zakharov	mean	**1.72E-03**	1.90E+02	4.52E+00	1.52E+02	1.40E+03	2.86E+01	2.66E+02	1.83E+02	2.65E+02	7.08E+02	3.12E+02	4.87E+02	4.05E+02
	min	**1.72E-03**	1.51E+02	4.52E+00	1.52E+02	1.40E+03	2.86E+01	2.66E+02	9.29E+01	2.36E+02	7.08E+02	3.12E+02	4.87E+02	3.01E+02
	max	**1.72E-03**	1.91E+02	4.52E+00	1.52E+02	1.40E+03	2.86E+01	2.66E+02	2.67E+02	3.01E+02	7.08E+02	3.12E+02	4.87E+02	4.95E+02
	std	1.10E-18	7.40E+00	**0.00E+00**	2.89E-14	6.94E-13	**0.00E+00**	1.16E-13	4.85E+01	1.59E+01	3.47E-13	**0.00E+00**	5.78E-14	4.76E+01
Sumsqu	mean	**1.15E-278**	1.83E+01	7.64E-06	3.44E-03	6.06E+03	3.86E-01	1.08E+01	1.69E+02	1.08E+02	2.23E+03	2.29E+02	1.15E+03	2.28E+02
	min	**3.42E-279**	2.70E+00	7.64E-06	3.44E-03	6.06E+03	3.86E-01	1.08E+01	1.10E-06	7.43E+01	2.23E+03	2.29E+02	1.15E+03	6.51E-06
	max	**2.46E-277**	1.88E+01	7.64E-06	3.44E-03	6.06E+03	3.86E-01	1.08E+01	8.91E+02	1.67E+02	2.23E+03	2.29E+02	1.15E+03	1.39E+03
	std	**0.00E+00**	2.95E+00	6.89E-21	1.32E-18	2.78E-12	1.13E-16	1.81E-15	1.82E+02	1.93E+01	9.25E-13	0.00E+00	4.63E-13	4.70E+02
Powell	mean	**2.00E-07**	1.37E+02	5.26E-02	3.84E+00	1.12E+05	1.53E+00	8.65E+01	1.08E+03	8.39E+01	9.96E+03	3.02E+02	5.50E+03	5.61E+02
	min	**7.61E-11**	1.34E+02	5.26E-02	3.84E+00	1.12E+05	1.53E+00	8.65E+01	8.45E-02	3.35E+01	9.96E+03	3.02E+02	5.50E+03	6.86E-04
	max	**2.07E-07**	2.13E+02	5.26E-02	3.84E+00	1.12E+05	1.53E+00	8.65E+01	6.39E+03	1.22E+02	9.96E+03	3.02E+02	5.50E+03	4.75E+03
	std	3.78E-08	1.45E+01	3.53E-17	**0.00E+00**	5.92E-11	**0.00E+00**	**0.00E+00**	1.53E+03	2.05E+01	**0.00E+00**	1.16E-13	4.63E-12	1.18E+03

Table 26 The Wilcoxon rank sum test results over 30 runs for dimension=30

	PSO	ABC	SSA	SSO	FA	DA	BBPSO	FPA	BBPSOV	SPSO	GPSO	ACPSO
Ackley	4.29E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	1.72E-12	1.72E-12	2.71E-14	2.71E-14	2.71E-14	1.72E-12
Dixon	2.71E-14	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.21E-12
Griewank	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	3.36E-11	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.21E-12
Rastrigin	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.21E-12
Rothyp	4.29E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	1.72E-12	1.72E-12	2.71E-14	2.71E-14	2.71E-14	1.72E-12
Rosenbrock	4.29E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	4.56E-11	1.72E-12	2.71E-14	2.71E-14	2.71E-14	2.59E-06
Sphere	4.29E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	1.72E-12	1.72E-12	2.71E-14	2.71E-14	2.71E-14	1.72E-12
Sumpow	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.21E-12
Zakharov	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.21E-12
Sumsqu	4.29E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	1.72E-12	1.72E-12	2.71E-14	2.71E-14	2.71E-14	1.72E-12
Powell	4.29E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	1.72E-12	1.72E-12	2.71E-14	2.71E-14	2.71E-14	1.72E-12

Besides the Wilcoxon rank sum test, the non-parametric Friedman test is also conducted. It tests the null hypothesis that the results of all the treatment methods have identical distributions or otherwise based on a Chi-square approximation. Table 27 shows the mean rankings of all the search methods over 30 runs based on the mean results of all test functions shown in Table 25 using the Friedman test. The mean ranking result of each algorithm is obtained by averaging the rankings of the mean results of all benchmark functions. As indicated in Table 27, the proposed model has the highest mean ranking in comparison with those of all the baseline methods. ABC and FA also show competitive rankings against other baseline methods. The p-value obtained using the Friedman test illustrated in Table 28 is lower than 0.001. It further ascertains that our results are better than those of all other search methods with a statistical significance.

The empirical results indicate that the proposed model has a fast convergence rate in most test cases. Moreover, it shows better capabilities in tackling

Table 27 The mean ranking results of all search methods based on the Friedman test for benchmark functions with dimension=30

Algorithms	Mean Ranking
Prop. PSO	1.18
PSO	6
ABC	2.27
SSA	3.55
SSO	13
FA	3.36
DA	6.45
BBPSO	7.36
FPA	6.91
BBPSOV	12
SPSO	9.09
GPSO	10.82
ACPSO	9

Table 28 The statistical results of the Friedman test for benchmark functions with dimension=30

Chi-Square	p-Value	Hypothesis
120.94	< 0.001	Rejected

local optima traps by locating the minimum global optima in most test cases. Example mean convergence curves over 30 runs generated using the logarithm scale with a base of 10 during the course of 500 iterations with respect to the Powell, Ackley and Sphere functions are provided in Figures 24-26, respectively. The visualization results indicate that the proposed model navigates various search spaces with a fast convergence speed. Owing to the adoption of diverse composite leaders and optimal search action selection reinforced by Q-learning, as demonstrated in the visualized convergence curves, the proposed optimizer also shows better capabilities in tackling local optima traps as compared with those of other baseline methods. A similar trend is also obtained for other benchmark functions.

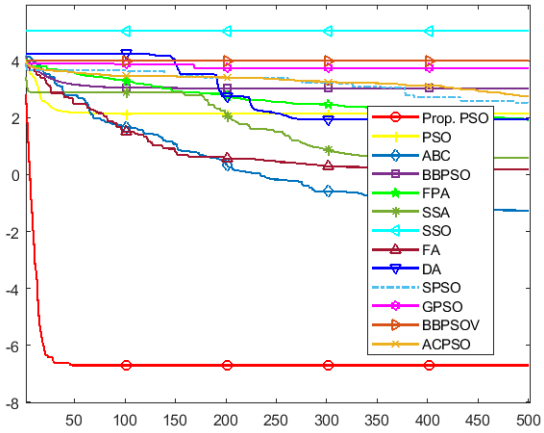

Fig. 24 Mean convergence rate comparison in the log scale over 30 runs for the Powell function with dimension=30

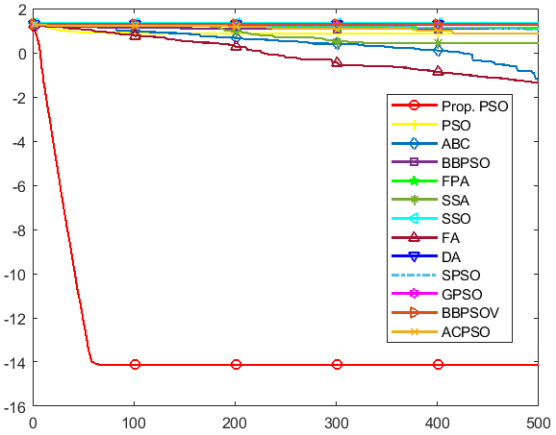

Fig. 25 Mean convergence rate comparison in the log scale over 30 runs for the Ackley function with dimension=30

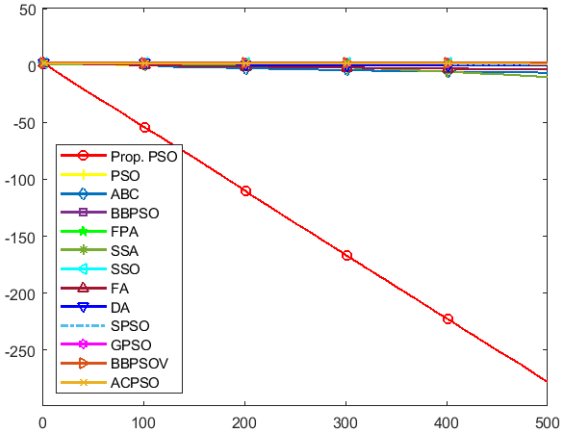

Fig. 26 Mean convergence rate comparison in the log scale over 30 runs for the Sphere function with dimension=30

To further test model effectiveness, we have also evaluated the proposed model using the benchmark functions with a dimension of 50. The experiments are performed using the following settings, i.e. population=50, iteration=1,000 and trial=30. A maximum number of 50,000 function evaluations is used by all search methods. The detailed evaluation and Wilcoxon rank sum statistical test results are provided in Tables 29-30.

Table 29 Evaluation results for the benchmark functions with dimension=50

		Prop. PSO	PSO	ABC	SSA	SSO	FA	DA	BBPSO	FPA	BBPSOV	SPSO	GPSO	ACPSO
Ackley	mean	**7.55E-15**	1.39E+01	4.07E-02	2.32E+00	2.11E+01	3.04E-02	1.96E+01	1.56E+01	1.31E+01	2.04E+01	1.40E+01	1.86E+01	1.39E+01
	min	**7.55E-15**	1.25E+01	4.07E-02	2.32E+00	2.11E+01	3.04E-02	1.96E+01	1.28E+01	1.17E+01	2.04E+01	1.40E+01	1.86E+01	1.23E+01
	max	**7.55E-15**	1.39E+01	4.07E-02	2.32E+00	2.11E+01	3.04E-02	1.96E+01	1.77E+01	1.42E+01	2.04E+01	1.40E+01	1.86E+01	1.48E+01
	std	**0.00E+00**	2.62E-01	**0.00E+00**	4.52E-16	7.23E-15	**0.00E+00**	1.45E-14	1.32E+00	7.26E-01	7.23E-15	**0.00E+00**	1.08E-14	4.87E-01
Dixon	mean	**6.67E-01**	7.19E+05	1.33E+00	7.36E-01	1.67E+07	2.96E+00	4.19E+04	1.54E+05	2.35E+03	5.79E+06	1.58E+05	3.10E+06	1.26E+05
	min	**6.67E-01**	6.21E+05	1.33E+00	7.36E-01	1.67E+07	2.96E+00	4.19E+04	8.77E+00	1.21E+03	5.79E+06	1.58E+05	3.10E+06	8.84E+04
	max	**6.67E-01**	7.23E+05	1.33E+00	7.36E-01	1.67E+07	2.96E+00	4.19E+04	6.27E+05	4.24E+03	5.79E+06	1.58E+05	3.10E+06	2.00E+05
	std	9.62E-12	1.86E+04	4.52E-16	**1.13E-16**	3.79E-09	1.36E-15	7.40E-12	2.00E+05	7.14E+02	1.89E-09	5.92E-11	4.74E-10	2.62E+04
Griewank	mean	1.93E-03	1.17E+02	**1.72E-05**	2.24E-05	1.95E+03	5.69E-03	1.73E+01	8.16E+01	1.77E+01	1.14E+03	1.68E+02	6.71E+02	1.48E+02
	min	1.93E-03	1.07E+02	**1.72E-05**	2.24E-05	1.95E+03	5.69E-03	1.73E+01	5.88E-03	1.33E+01	1.14E+03	1.68E+02	6.71E+02	1.18E+02
	max	1.93E-03	1.18E+02	**1.72E-05**	2.24E-05	1.95E+03	5.69E-03	1.73E+01	3.61E+02	2.42E+01	1.14E+03	1.68E+02	6.71E+02	1.89E+02
	std	**0.00E+00**	1.91E+00	**0.00E+00**	**0.00E+00**	9.25E-13	**0.00E+00**	1.08E-14	9.57E+01	2.27E+00	9.25E-13	5.78E-14	1.16E-13	1.56E+01
Rastrigin	mean	**1.87E-03**	2.49E+02	8.42E+00	4.88E+01	1.07E+03	2.89E+01	1.92E+02	2.65E+02	2.95E+02	7.95E+02	4.41E+02	6.07E+02	4.47E+02
	min	**1.87E-03**	2.49E+02	8.42E+00	4.88E+01	1.07E+03	2.89E+01	1.92E+02	1.24E+02	2.50E+02	7.95E+02	4.41E+02	6.07E+02	3.93E+02
	max	**1.87E-03**	2.52E+02	8.42E+00	4.88E+01	1.07E+03	2.89E+01	1.92E+02	3.52E+02	3.34E+02	7.95E+02	4.41E+02	6.07E+02	5.07E+02
	std	**0.00E+00**	6.16E-01	3.61E-15	7.23E-15	4.63E-13	3.61E-15	2.89E-14	5.21E+01	1.82E+01	1.16E-13	2.89E-13	4.63E-13	2.73E+01
Rothyp	mean	**0.00E+00**	3.45E+05	7.21E-06	8.40E+01	2.36E+06	1.81E+01	2.56E+04	8.89E+04	1.66E+04	1.36E+06	1.41E+05	6.61E+05	6.60E+05
	min	**0.00E+00**	3.44E+05	7.21E-06	8.40E+01	2.36E+06	1.81E+01	2.56E+04	1.65E-01	1.33E+04	1.36E+06	1.41E+05	6.61E+05	3.84E+05
	max	**0.00E+00**	3.76E+05	7.21E-06	8.40E+01	2.36E+06	1.81E+01	2.56E+04	2.75E+05	2.35E+04	1.36E+06	1.41E+05	6.61E+05	7.95E+05
	std	**0.00E+00**	5.90E+03	3.45E-21	1.45E-14	9.47E-10	**0.00E+00**	1.11E-11	7.96E+04	2.54E+03	4.74E-10	8.88E-11	1.18E-10	9.64E+04
Rosenbrock	mean	4.48E+01	3.42E+05	**2.33E-01**	1.34E+02	9.16E+06	4.71E+01	9.06E+04	2.79E+05	2.46E+04	2.96E+06	1.02E+05	1.37E+06	8.63E+04
	min	4.48E+01	3.39E+05	**2.33E-01**	1.34E+02	9.16E+06	4.71E+01	9.06E+04	5.72E+02	1.56E+04	2.96E+06	1.02E+05	1.37E+06	4.92E+04
	max	4.48E+01	4.29E+05	**2.33E-01**	1.34E+02	9.16E+06	4.71E+01	9.06E+04	1.11E+06	3.71E+04	2.96E+06	1.02E+05	1.37E+06	1.51E+05
	std	4.24E-03	1.65E+04	8.47E-17	8.67E-14	3.79E-09	2.89E-14	5.92E-11	1.98E+05	5.72E+03	9.47E-10	**0.00E+00**	**0.00E+00**	2.30E+04
Sphere	mean	**0.00E+00**	1.74E+01	1.30E-07	1.52E-10	5.68E+02	7.32E-04	4.22E+00	1.58E+01	5.50E+00	3.43E+02	4.85E+01	1.24E+02	4.19E+01
	min	**0.00E+00**	1.64E+01	1.30E-07	1.52E-10	5.68E+02	7.32E-04	4.22E+00	4.16E-06	4.46E+00	3.43E+02	4.85E+01	1.24E+02	3.21E+01
	max	**0.00E+00**	4.81E+01	1.30E-07	1.52E-10	5.68E+02	7.32E-04	4.22E+00	7.86E+01	6.67E+00	3.43E+02	4.85E+01	1.24E+02	5.34E+01
	std	**0.00E+00**	5.80E+00	1.08E-22	2.63E-26	1.16E-13	2.21E-19	1.81E-15	2.34E+01	6.36E-01	5.78E-14	7.23E-15	2.89E-14	5.56E+00
Sumpow	mean	**0.00E+00**	7.49E-08	4.29E-16	3.16E-08	6.54E+00	1.18E-07	1.16E-04	4.76E-18	8.20E-07	5.75E-01	3.08E-04	2.42E-01	3.12E-04
	min	**0.00E+00**	1.77E-08	4.29E-16	3.16E-08	6.54E+00	1.18E-07	1.16E-04	1.23E-26	7.11E-08	5.75E-01	3.08E-04	2.42E-01	3.97E-05
	max	**0.00E+00**	1.73E-06	4.29E-16	3.16E-08	6.54E+00	1.18E-07	1.16E-04	6.33E-17	4.01E-06	5.75E-01	3.08E-04	2.42E-01	1.16E-03
	std	**0.00E+00**	3.13E-07	1.50E-31	1.35E-23	**0.00E+00**	4.04E-23	6.89E-20	1.57E-17	7.92E-07	1.13E-16	**0.00E+00**	8.47E-17	2.58E-04
Zakharov	mean	**3.62E-03**	5.47E+02	9.33E+00	3.28E+02	2.23E+03	5.19E+01	3.83E+02	3.79E+02	4.46E+02	1.36E+03	5.76E+02	8.92E+02	5.92E+02
	min	**3.62E-03**	3.59E+02	9.33E+00	3.28E+02	2.23E+03	5.19E+01	3.83E+02	2.27E+02	3.79E+02	1.36E+03	5.76E+02	8.92E+02	5.11E+02
	max	**3.62E-03**	5.53E+02	9.33E+00	3.28E+02	2.23E+03	5.19E+01	3.83E+02	5.23E+02	5.11E+02	1.36E+03	5.76E+02	8.92E+02	6.88E+02
	std	**0.00E+00**	3.55E+01	**0.00E+00**	1.73E-13	**0.00E+00**	**0.00E+00**	1.73E-13	8.30E+01	2.60E+01	**0.00E+00**	3.47E-13	1.16E-13	4.32E+01
Sumsqu	mean	**0.00E+00**	7.72E+02	3.93E-07	1.81E+00	1.56E+04	1.96E+00	5.82E+00	7.59E+02	1.15E+02	8.27E+03	9.65E+02	3.52E+03	9.49E+02
	min	**0.00E+00**	3.08E+02	3.93E-07	1.81E+00	1.56E+04	1.96E+00	5.82E+00	9.96E-02	7.68E+01	8.27E+03	9.65E+02	3.52E+03	7.54E+02
	max	**0.00E+00**	7.88E+02	3.93E-07	1.81E+00	1.56E+04	1.96E+00	5.82E+00	2.12E+03	1.68E+02	8.27E+03	9.65E+02	3.52E+03	1.14E+03
	std	**0.00E+00**	8.78E+01	1.08E-22	9.03E-16	1.11E-11	1.13E-15	1.81E-15	5.57E+02	2.36E+01	1.85E-12	6.94E-13	2.31E-12	9.74E+01
Powell	mean	**6.82E-08**	4.52E+02	8.26E-02	5.10E+00	1.62E+05	3.92E+00	3.18E+02	2.78E+03	8.84E+01	2.45E+04	1.25E+03	1.34E+04	1.31E+03
	min	**9.76E-09**	3.15E+02	8.26E-02	5.10E+00	1.62E+05	3.92E+00	3.18E+02	7.03E-01	5.37E+01	2.45E+04	1.25E+03	1.34E+04	8.98E+02
	max	**7.02E-08**	4.57E+02	8.26E-02	5.10E+00	1.62E+05	3.92E+00	3.18E+02	9.90E+03	1.40E+02	2.45E+04	1.25E+03	1.34E+04	1.84E+03
	std	1.10E-08	2.59E+01	1.41E-17	1.81E-15	**0.00E+00**	1.81E-15	5.78E-14	2.15E+03	2.12E+01	**0.00E+00**	9.25E-13	3.70E-12	2.22E+02

Table 30 The Wilcoxon rank sum test results over 30 runs with dimension=50

	PSO	ABC	SSA	SSO	FA	DA	BBPSO	FPA	BBPSOV	SPSO	GPSO	ACPSO
Ackley	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.69E-14
Dixon	4.29E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	1.72E-12	1.72E-12	2.71E-14	2.71E-14	2.71E-14	2.71E-14
Griewank	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.69E-14
Rastrigin	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.69E-14
Rothyp	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.69E-14
Rosenbrock	4.29E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	1.72E-12	1.72E-12	2.71E-14	2.71E-14	2.71E-14	3.02E-11
Sphere	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.69E-14
Sumpow	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.69E-14
Zakharov	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.69E-14
Sumsqu	2.71E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.69E-14	1.21E-12	1.21E-12	1.69E-14	1.69E-14	1.69E-14	1.69E-14
Powell	4.29E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	2.71E-14	1.72E-12	1.72E-12	2.71E-14	2.71E-14	2.71E-14	2.71E-14

As shown in Tables 29-30, the proposed model achieves statistically better results than those of all the baseline methods for most numerical optimization problems. In particular, it achieves the most optimal global minima of '0' for Rotated Hyper-Ellipsoid, Sphere, Sum of Different Powers, and Sum Square functions. The exceptions are for Rosenbrock where ABC obtains the best global minima and outperforms the proposed model with statistical significance. In addition, for Griewank, ABC and SSA obtain statistical better performances than those of the proposed model. For these two test functions, i.e. Rosenbrock and Griewank, the proposed model achieves the second and third best results, respectively. These numerical function results also indicate the proposed model with composite leaders and Q-learning based search action optimization possesses better capabilities in tackling local optima traps and achieves the best global minima in most test cases.

The significance of the proposed model is also further ascertained by the Friedman test. As indicated in Tables 31-32, the proposed model dominates the highest mean ranking for solving benchmark functions with dimension=50 as compared with those of other search methods. The p-value from the Friedman test is also lower than 0.05, which indicates that the proposed model is better than all the baseline methods with a statistical significance.

Table 31 The mean ranking results of all search methods based on the Friedman test for benchmark functions with dimension=50

Algorithms	Mean Ranking
Prop. PSO	1.27
PSO	7.77
ABC	2.18
SSA	3.36
SSO	13
FA	3.55
DA	6.36
BBPSO	7.09
FPA	5.91
BBPSOV	12
SPSO	8.91
GPSO	10.91
ACPSO	8.68

Table 32 The statistical results of the Friedman test for benchmark functions with dimension=50

Chi-Square	p-Value	Hypothesis
120.52	< 0.001	Rejected

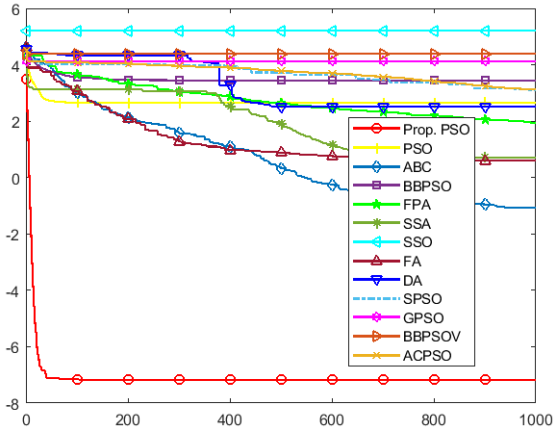

Fig. 27 Mean convergence rate comparison in the log scale over 30 runs for the Powell function with dimension=50

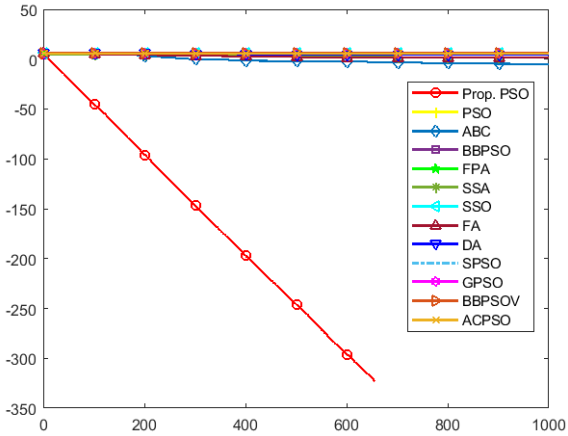

Fig. 28 Mean convergence rate comparison in the log scale over 30 runs for the Rotated Hyper-Ellipsoid function with dimension=50

Figures 27 and 28 depict the mean convergence curves of all search methods over 30 runs during the course of 1,000 iterations with respect to the Powell and Rotated Hyper-Ellipsoid functions. As mentioned earlier, these mean convergence graphs are generated using the logarithm scale with a base of 10 for these example test functions. As indicated in Figures 27-28, the proposed model

illustrates sufficient capabilities in navigating through complex search spaces and shows great competence in tackling local optima traps. For the Rotated Hyper-Ellipsoid function, the proposed model achieves the global minimum of '0' since iteration 657 based on the results over 30 runs. Owing to the fact that $\log_{10} 0 = -\infty$, the convergence curve of our optimizer is shown until iteration 656. These convergence graphs again illustrate the model's fastest convergence rates and its competitive capabilities in achieving the most optimal global minima, in comparison with those of all the baseline search methods. A similar trend is also observed for the proposed optimizer for other benchmark functions.

8 Conclusion

In this research, we have proposed transfer learning and hybrid deep networks with PSO-based optimal hyperparameter selection for undertaking deepfake classification. A new PSO model is proposed for optimal hyperparameter search by integrating composite leader generation and reinforcement learning based search operation adjustment. The preprocessing of face cropping is also conducted to extract the facial regions and eliminate background noise. Evaluated using several challenging deepfake datasets, the proposed PSO-optimized EfficientNet-B0 and EfficientNet-GRU models show enhanced performance. In particular, EfficientNet-GRU with optimal settings yielded by our proposed optimizer achieves the best benchmarks and outperforms existing studies significantly in different experimental settings. The proposed optimizer also achieves statistically better performance against those of other search methods in solving diverse unimodal and multimodal mathematical landscapes.

The next steps for this research could include further exploration of different loss functions and the integration with other optimization algorithms to further enhance performance. Additionally, incorporating more diverse and larger datasets for model training could also help improve deepfake detection accuracy. Another potential direction would be to investigate the use of other state-of-the-art models, such as transformer-based models and contrastive learning, for deepfake detection. Such explorations could provide valuable insights into the effectiveness of different approaches for handling the deepfake detection problem. Finally, reinforcement learning algorithms with continuous action space will also be studied to further enhance the proposed optimizer pertaining to search coefficient generation to further increase search diversity.

Declarations

Funding acknowledgement

This research was supported by StoryFutures project funded by Arts and Humanities Research Council (AHRC).

Conflicts of interest/Competing interests

The authors have no relevant financial or non-financial interests to disclose.

Availability of data and material

This research employs publicly available deepfake datasets for experimental studies.

Code availability

The authors will publish the code for the proposed work in a dedicated website after the acceptance of the paper.

Authors' contributions

Leandro Cunha: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Roles/Writing - original draft, Writing - review and editing.

Li Zhang: Conceptualization, Formal analysis, Investigation, Methodology, Resources, Supervision, Validation, Roles/Writing - original draft, Writing - review and editing.

Bilal Sowan: Supervision, Writing - review and editing.

Chee Peng Lim: Supervision, Validation, Writing - review and editing.

Yinghui Kong: Writing - review and editing.

Ethics approval

The proposed work has gained organizational ethical approval.

Consent to participate

The consent to participate has been obtained from all the co-authors for the proposed studies.

Consent for publication

The consent for publication has been obtained from all the co-authors for the proposed studies.

References

[1] Nightingale, S.J., Wade, K.A., Watson, D.G.: Can people identify original and manipulated photos of real-world scenes? Cognitive Research:

Principles and Implications **2**(1), 1–21 (2017). https://doi.org/10.1186/s41235-017-0067-2

[2] Sabir, E., Cheng, J., Jaiswal, A., AbdAlmageed, W., Masi, I., Natarajan, P.: Recurrent convolutional strategies for face manipulation detection in videos. Interfaces **3**(1), 80–87 (2019)

[3] Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., Nießner, M.: Faceforensics++: Learning to detect manipulated facial images. In: 2019 IEEE International Conference on Computer Vision, pp. 1–11 (2019). https://doi.org/10.1109/ICCV.2019.00009. IEEE

[4] Natsume, R., Yatagawa, T., Morishima, S.: Rsgan: face swapping and editing using face and hair representation in latent spaces, 1–2 (2018). https://doi.org/10.1145/3230744.3230818

[5] Thies, J., Zollhofer, M., Stamminger, M., Theobalt, C., Nießner, M.: Face2face: Real-time face capture and reenactment of rgb videos. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2387–2395 (2016). https://doi.org/10.1109/CVPR.2016.262. IEEE

[6] Kim, H., Garrido, P., Tewari, A., Xu, W., Thies, J., Niessner, M., Pérez, P., Richardt, C., Zollhöfer, M., Theobalt, C.: Deep video portraits. ACM Transactions on Graphics **37**(4), 1–14 (2018). https://doi.org/10.1145/3197517.3201283

[7] Liang, M., Hu, X.: Recurrent convolutional neural network for object recognition. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3367–3375 (2015). https://doi.org/10.1109/CVPR.2015.7298958. IEEE

[8] Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2625–2634 (2015). https://doi.org/10.1109/TPAMI.2016.2599174. IEEE

[9] Zhang, L., Lim, C.P., Yu, Y.: Intelligent human action recognition using an ensemble model of evolving deep networks with swarm-based optimization. Knowledge-Based Systems **220**, 106918 (2021). https://doi.org/10.1016/j.knosys.2021.106918

[10] Ahn, D., Kim, S., Hong, H., Ko, B.C.: STAR-Transformer: A Spatio-temporal Cross Attention Transformer for Human Action Recognition. In: 2023 IEEE Winter Conference on Applications of Computer Vision, pp. 3330–3339 (2023). https://doi.org/10.1109/WACV56688.2023.00333. IEEE

[11] Slade, S., Zhang, L., Yu, Y., Lim, C.P.: An evolving ensemble model of multi-stream convolutional neural networks for human action recognition in still images. Neural Computing and Applications **34**(11), 9205–9231 (2022). https://doi.org/10.1007/s00521-022-06947-6

[12] Dasari, P., Zhang, L., Yu, Y., Huang, H., Gao, R.: Human Action Recognition Using Hybrid Deep Evolving Neural Networks. In: 2022 International Joint Conference on Neural Networks, pp. 1–8 (2022). https://doi.org/10.1109/IJCNN55064.2022.9892025. IEEE

[13] Hu, J., Liao, X., Wang, W., Qin, Z.: Detecting compressed deepfake videos in social networks using frame-temporality two-stream convolutional network. IEEE Transactions on Circuits and Systems for Video Technology **32**(3), 1089–1102 (2021). https://doi.org/10.1109/TCSVT.2021.3074259

[14] Demir, I., Ciftci, U.A.: Where do deep fakes look? synthetic face detection via gaze tracking. In: 2021 ACM Symposium on Eye Tracking Research and Applications, pp. 1–11 (2021). https://doi.org/10.1145/3448017.3457387. ACM

[15] Zhao, H., Zhou, W., Chen, D., Wei, T., Zhang, W., Yu, N.: Multi-attentional deepfake detection. In: 2021 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2185–2194 (2021). https://doi.org/10.1109/CVPR46437.2021.00222. IEEE

[16] Kennedy, J., Eberhart, R.: Particle swarm optimization. In: 1995 International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995). https://doi.org/10.1109/ICNN.1995.488968. IEEE

[17] Yamasaki, T., Honma, T., Aizawa, K.: Efficient optimization of convolutional neural networks using particle swarm optimization. In: 2017 IEEE Third International Conference on Multimedia Big Data (BigMM), pp. 70–73 (2017). https://doi.org/10.1109/BigMM.2017.69. IEEE

[18] Zhang, L., Lim, C.P., Yu, Y., Jiang, M.: Sound classification using evolving ensemble models and Particle Swarm Optimization. Applied Soft Computing **116**, 108322 (2022). https://doi.org/10.1016/j.asoc.2021.108322

[19] Tan, T.Y., Zhang, L., Lim, C.P.: Adaptive melanoma diagnosis using evolving clustering, ensemble and deep neural networks. Knowledge-Based Systems **187**, 104807 (2020). https://doi.org/10.1016/j.knosys.2019.06.015

[20] Fielding, B., Zhang, L.: Evolving image classification architectures with enhanced particle swarm optimisation. IEEE Access **6**, 68560–68575 (2018). https://doi.org/10.1109/ACCESS.2018.2880416

[21] Zhang, L., Liu, X., Guan, H.: Automtl: A programming framework for automating efficient multi-task learning. Advances in Neural Information Processing Systems **35**, 34216–34228 (2022)

[22] Stützle, T., López-Ibáñez, M.: Automated design of metaheuristic algorithms. Handbook of metaheuristics, 541–579 (2019). https://doi.org/10.1007/978-3-319-91086-4_17

[23] Mirfallah Lialestani, S.P., Parcerisa, D., Himi, M., Abbaszadeh Shahri, A.: Generating 3D geothermal maps in Catalonia, Spain using a hybrid adaptive multitask deep learning procedure. Energies **15**(13), 4602 (2022). https://doi.org/10.3390/en15134602

[24] Abbaszadeh Shahri, A., Khorsand Zak, M., Abbaszadeh Shahri, H.: A modified firefly algorithm applying on multi-objective radial-based function for blasting. Neural Computing and Applications, 1–17 (2022). https://doi.org/10.1007/s00521-021-06544-z

[25] Cheng, J., Liu, J., Kuang, H., Wang, J.: A fully automated multimodal MRI-based multi-task learning for glioma segmentation and IDH genotyping. IEEE Transactions on Medical Imaging **41**(6), 1520–1532 (2022). https://doi.org/10.1109/TMI.2022.3142321

[26] Cheng, M.-Y., Prayogo, D.: Symbiotic organisms search: a new metaheuristic optimization algorithm. Computers & Structures **139**, 98–112 (2014). https://doi.org/10.1016/j.compstruc.2014.03.007

[27] Gharehchopogh, F.S., Shayanfar, H., Gholizadeh, H.: A comprehensive survey on symbiotic organisms search algorithms. Artificial Intelligence Review **53**, 2265–2312 (2020). https://doi.org/10.1007/s10462-019-09733-4

[28] Dhiman, G., Kumar, V.: Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Advances in Engineering Software **114**, 48–70 (2017). https://doi.org/10.1016/j.advengsoft.2017.05.014

[29] Ghafori, S., Gharehchopogh, F.S.: Advances in spotted hyena optimizer: a comprehensive survey. Archives of Computational Methods in Engineering, 1–22 (2021). https://doi.org/10.1007/s11831-021-09624-4

[30] Xue, J., Shen, B.: A novel swarm intelligence optimization approach: sparrow search algorithm. Systems Science & Control Engineering **8**(1), 22–34 (2020). https://doi.org/10.1080/21642583.2019.1708830

[31] Gharehchopogh, F.S., Namazi, M., Ebrahimi, L., Abdollahzadeh, B.: Advances in sparrow search algorithm: a comprehensive survey. Archives

of Computational Methods in Engineering **30**(1), 427–455 (2023). https://doi.org/10.1007/s11831-022-09804-w

[32] Yang, X.-S., He, X.: Firefly algorithm: recent advances and applications. International Journal of Swarm Intelligence **1**(1), 36–50 (2013). https://doi.org/10.1504/IJSI.2013.055801

[33] Mirjalili, S.: SCA: a sine cosine algorithm for solving optimization problems. Knowledge-Based Systems **96**, 120–133 (2016). https://doi.org/10.1016/j.knosys.2015.12.022

[34] Kiran, M.S.: TSA: Tree-seed algorithm for continuous optimization. Expert Systems with Applications **42**(19), 6686–6698 (2015). https://doi.org/10.1016/j.eswa.2015.04.055

[35] Gharehchopogh, F.S.: Advances in Tree Seed Algorithm: A Comprehensive Survey. Archives of Computational Methods in Engineering **29**(1), 3281–3304 (2022). https://doi.org/10.1007/s11831-021-09698-0

[36] Karaboga, D.: Artificial bee colony algorithm. Scholarpedia **5**(3), 6915 (2010). https://doi.org/10.4249/scholarpedia.6915

[37] Gharehchopogh, F.S.: An improved tunicate swarm algorithm with best-random mutation strategy for global optimization problems. Journal of Bionic Engineering **19**(4), 1177–1202 (2022). https://doi.org/10.1007/s42235-022-00185-1

[38] James, J., Li, V.O.: A social spider algorithm for global optimization. Applied Soft Computing **30**, 614–627 (2015). https://doi.org/10.1016/j.asoc.2015.02.014

[39] Slade, S., Zhang, L., Huang, H., Asadi, H., Lim, C.P., Yu, Y., Zhao, D., Lin, H., Gao, R.: Neural Inference Search for Multiloss Segmentation Models. IEEE Transactions on Neural Networks and Learning Systems (2023). https://doi.org/10.1109/TNNLS.2023.3282799

[40] Chen, Q., Chen, Y., Jiang, W.: Genetic particle swarm optimization–based feature selection for very-high-resolution remotely sensed imagery object change detection. Sensors **16**(8), 1204 (2016). https://doi.org/10.3390/s16081204

[41] Pandit, D., Zhang, L., Chattopadhyay, S., Lim, C.P., Liu, C.: A scattering and repulsive swarm intelligence algorithm for solving global optimization problems. Knowledge-Based Systems **156**, 12–42 (2018). https://doi.org/10.1016/j.knosys.2018.05.002

[42] Zhang, L., Lim, C.P., Liu, C.: Enhanced Bare-Bones Particle Swarm Optimization based Evolving Deep Neural Networks. Expert Systems with Applications, 120642 (2023). https://doi.org/10.1016/j.eswa.2023.120642

[43] Zhang, L., Mistry, K., Neoh, S.C., Lim, C.P.: Intelligent facial emotion recognition using moth-firefly optimization. Knowledge-Based Systems **111**, 248–267 (2016). https://doi.org/10.1016/j.knosys.2016.08.018

[44] Mirjalili, S., Gandomi, A.H., Mirjalili, S.Z., Saremi, S., Faris, H., Mirjalili, S.M.: Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. Advances in Engineering Software **114**, 163–191 (2017). https://doi.org/10.1016/j.advengsoft.2017.07.002

[45] Tan, T.Y., Zhang, L., Lim, C.P., Fielding, B., Yu, Y., Anderson, E.: Evolving ensemble models for image segmentation using enhanced particle swarm optimization. IEEE Access **7**, 34004–34019 (2019). https://doi.org/10.1109/ACCESS.2019.2903015

[46] Tan, T.Y., Zhang, L., Neoh, S.C., Lim, C.P.: Intelligent skin cancer detection using enhanced particle swarm optimization. Knowledge-Based Systems **158**, 118–135 (2018). https://doi.org/10.1016/j.knosys.2018.05.042

[47] Xie, H., Zhang, L., Lim, C.P., Yu, Y., Liu, C., Liu, H., Walters, J.: Improving K-means clustering with enhanced Firefly Algorithms. Applied Soft Computing **84**, 105763 (2019). https://doi.org/10.1016/j.asoc.2019.105763

[48] Mistry, K., Zhang, L., Neoh, S.C., Lim, C.P., Fielding, B.: A micro-GA embedded PSO feature selection approach to intelligent facial emotion recognition. IEEE Transactions on Cybernetics **47**(6), 1496–1509 (2016). https://doi.org/10.1109/TCYB.2016.2549639

[49] Srisukkham, W., Zhang, L., Neoh, S.C., Todryk, S., Lim, C.P.: Intelligent leukaemia diagnosis with bare-bones PSO based feature optimization. Applied Soft Computing **56**, 405–419 (2017). https://doi.org/10.1016/j.asoc.2017.03.024

[50] Zhang, K., Zhang, Z., Li, Z., Qiao, Y.: Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters **23**(10), 1499–1503 (2016). https://doi.org/10.1109/LSP.2016.2603342

[51] Li, Y., Yang, X., Sun, P., Qi, H., Lyu, S.: Celeb-df: A large-scale challenging dataset for deepfake forensics. In: 2020 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3207–3216 (2020). https://doi.org/10.1109/CVPR42600.2020.00327. IEEE

[52] Dolhansky, B., Bitton, J., Pflaum, B., Lu, J., Howes, R., Wang, M., Ferrer, C.C.: The deepfake detection challenge (dfdc) dataset. arXiv preprint (2020). https://doi.org/10.48550/arXiv.2006.07397

[53] Wolf, L., Hassner, T., Maoz, I.: Face recognition in unconstrained videos with matched background similarity. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition, pp. 529–534 (2011). https://doi.org/10.1109/CVPR.2011.5995566. IEEE

[54] Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: 2019 International Conference on Machine Learning, pp. 6105–6114 (2019). PMLR

[55] Kinghorn, P., Zhang, L., Shao, L.: A region-based image caption generator with refined descriptions. Neurocomputing **272**, 416–424 (2018). https://doi.org/10.1016/j.neucom.2017.07.014

[56] Zhang, A., Lipton, Z.C., Li, M., Smola, A.J.: Dive into Deep Learning. arXiv preprint (2021). https://doi.org/10.48550/arXiv.2106.11342

[57] Kinghorn, P., Zhang, L., Shao, L.: A hierarchical and regional deep learning architecture for image description generation. Pattern Recognition Letters **119**, 77–85 (2019). https://doi.org/10.1016/j.patrec.2017.09.013

[58] Watkins, C.J., Dayan, P.: Q-learning. Machine Learning **8**, 279–292 (1992)

[59] Zheng, Y., Bao, J., Chen, D., Zeng, M., Wen, F.: Exploring temporal coherence for more general video face forgery detection. In: 2021 IEEE International Conference on Computer Vision, pp. 15044–15054 (2021). https://doi.org/10.1109/ICCV48922.2021.01477. IEEE

[60] Shiohara, K., Yamasaki, T.: Detecting deepfakes with self-blended images. In: 2022 IEEE Conference on Computer Vision and Pattern Recognition, pp. 18720–18729 (2022). https://doi.org/10.1109/CVPR52688.2022.01816. IEEE

[61] Zhao, T., Xu, X., Xu, M., Ding, H., Xiong, Y., Xia, W.: Learning self-consistency for deepfake detection. In: 2021 IEEE International Conference on Computer Vision, pp. 15023–15033 (2021). https://doi.org/10.1109/ICCV48922.2021.01475. IEEE

[62] Wang, G., Jiang, Q., Jin, X., Li, W., Cui, X.: MC-LCR: Multimodal contrastive classification by locally correlated representations for effective face forgery detection. Knowledge-Based Systems **250**, 109114 (2022). https://doi.org/10.1016/j.knosys.2022.109114

[63] Kennedy, J.: Bare bones particle swarms. In: 2003 IEEE Swarm Intelligence Symposium, pp. 80–87 (2003). https://doi.org/10.1109/SIS.2003.1202251. IEEE

[64] Yang, X.-S.: Flower pollination algorithm for global optimization. In: 2012 International Conference on Unconventional Computing and Natural Computation, pp. 240–249 (2012). https://doi.org/10.1007/978-3-642-32894-7_27. Springer

[65] Mirjalili, S.: Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Computing and Applications **27**, 1053–1073 (2016). https://doi.org/10.1007/s00521-015-1920-1

[66] Tan, T.Y., Zhang, L., Lim, C.P.: Intelligent skin cancer diagnosis using improved particle swarm optimization and deep learning models. Applied Soft Computing **84**, 105725 (2019). https://doi.org/10.1016/j.asoc.2019.105725

[67] Song, L., Fang, Z., Li, X., Dong, X., Jin, Z., Chen, Y., Lyu, S.: Adaptive face forgery detection in cross domain. In: 2022 European Conference on Computer Vision, pp. 467–484 (2022). https://doi.org/10.1007/978-3-031-19830-4_27. Springer

[68] Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, pp. 6450–6459 (2018). https://doi.org/10.1109/CVPR.2018.00675. IEEE

[69] Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1492–1500 (2017). https://doi.org/10.1109/CVPR.2017.634. IEEE

[70] Kandasamy, V., Hubálovskỳ, Š., Trojovskỳ, P.: Deep fake detection using a sparse auto encoder with a graph capsule dual graph CNN. PeerJ Computer Science **8**, 953 (2022). https://doi.org/10.7717/peerj-cs.953

[71] Haliassos, A., Vougioukas, K., Petridis, S., Pantic, M.: Lips don't lie: A generalisable and robust approach to face forgery detection. In: 2021 IEEE Conference on Computer Vision and Pattern Recognition, pp. 5039–5049 (2021). https://doi.org/10.1109/CVPR46437.2021.00500. IEEE

[72] Liu, H., Li, X., Zhou, W., Chen, Y., He, Y., Xue, H., Zhang, W., Yu, N.: Spatial-phase shallow learning: rethinking face forgery detection in frequency domain. In: 2021 IEEE Conference on Computer Vision and Pattern Recognition, pp. 772–781 (2021). https://doi.org/10.1109/

CVPR46437.2021.00083. IEEE

[73] Wang, S.-Y., Wang, O., Zhang, R., Owens, A., Efros, A.A.: CNN-generated images are surprisingly easy to spot... for now. In: 2020 IEEE Conference on Computer Vision and Pattern Recognition, pp. 8695–8704 (2020). https://doi.org/10.1109/CVPR42600.2020.00872. IEEE

[74] Nguyen, H.H., Fang, F., Yamagishi, J., Echizen, I.: Multi-task learning for detecting and segmenting manipulated facial images and videos. In: 2019 IEEE Conference on Biometrics Theory, Applications and Systems (BTAS), pp. 1–8 (2019). https://doi.org/10.1109/BTAS46853.2019.9185974. IEEE

[75] Chai, L., Bau, D., Lim, S.-N., Isola, P.: What makes fake images detectable? understanding properties that generalize. In: 2020 European Conference on Computer Vision, pp. 103–120 (2020). https://doi.org/10.1007/978-3-030-58574-7_7. Springer

[76] Masi, I., Killekar, A., Mascarenhas, R.M., Gurudatt, S.P., AbdAlmageed, W.: Two-branch recurrent network for isolating deepfakes in videos. In: 2020 European Conference on Computer Vision, pp. 667–684 (2020). https://doi.org/10.1007/978-3-030-58571-6_39. Springer

[77] Tolosana, R., Romero-Tapiador, S., Fierrez, J., Vera-Rodriguez, R.: Deepfakes evolution: Analysis of facial regions and fake detection performance. In: 2021 International Conference on Pattern Recognition, pp. 442–456 (2021). https://doi.org/10.1007/978-3-030-68821-9_38. Springer

[78] Li, X., Lang, Y., Chen, Y., Mao, X., He, Y., Wang, S., Xue, H., Lu, Q.: Sharp multiple instance learning for deepfake video detection. In: 2020 ACM International Conference on Multimedia, pp. 1864–1872 (2020). https://doi.org/10.1145/3394171.3414034. ACM

[79] Zhang, D., Li, C., Lin, F., Zeng, D., Ge, S.: Detecting Deepfake Videos with Temporal Dropout 3DCNN. In: 2021 International Joint Conference on Artificial Intelligence, pp. 1288–1294 (2021). https://doi.org/10.24963/ijcai.2021/178. IJCAI

[80] Güera, D., Delp, E.J.: Deepfake video detection using recurrent neural networks. In: 2018 IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 1–6 (2018). https://doi.org/10.1109/AVSS.2018.8639163. IEEE

[81] Hu, J., Liao, X., Liang, J., Zhou, W., Qin, Z.: Finfer: Frame inference-based deepfake detection for high-visual-quality videos. In: 2022 AAAI Conference on Artificial Intelligence, vol. 36, pp. 951–959 (2022). https://doi.org/10.1609/aaai.v36i1.19978. AAAI Press

[82] Li, L., Bao, J., Zhang, T., Yang, H., Chen, D., Wen, F., Guo, B.: Face x-ray for more general face forgery detection. In: 2020 IEEE Conference on Computer Vision and Pattern Recognition, pp. 5001–5010 (2020). https://doi.org/10.1109/CVPR42600.2020.00505. IEEE

[83] Naik, D.L., Kiran, R.: A novel sensitivity-based method for feature selection. Journal of Big Data **8**, 1–16 (2021). https://doi.org/10.1186/s40537-021-00515-w

[84] Xue, B., Zhang, M., Browne, W.N., Yao, X.: A survey on evolutionary computation approaches to feature selection. IEEE Transactions on Evolutionary Computation **20**(4), 606–626 (2015). https://doi.org/10.1109/TEVC.2015.2504420

[85] Neoh, S.C., Zhang, L., Mistry, K., Hossain, M.A., Lim, C.P., Aslam, N., Kinghorn, P.: Intelligent facial emotion recognition using a layered encoding cascade optimization model. Applied Soft Computing **34**, 72–93 (2015). https://doi.org/10.1016/j.asoc.2015.05.006

[86] Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: 2017 IEEE International Conference on Computer Vision, pp. 618–626 (2017). https://doi.org/10.1109/ICCV.2017.74. IEEE

[87] Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: 2016 International Conference on Machine Learning, pp. 1050–1059 (2016). PMLR

[88] Bórquez, S., Pezoa, R., Salinas, L., Torres, C.E.: Uncertainty estimation in the classification of histopathological images with HER2 overexpression using Monte Carlo Dropout. Biomedical Signal Processing and Control **85**, 104864 (2023). https://doi.org/10.1016/j.bspc.2023.104864

[89] Islam, M.F., Rahman, F.B., Zabeen, S., Islam, M.A., Hossain, M.S., Mehedi, M.H.K., Manab, M.A., Rasel, A.A.: RNN Variants vs Transformer Variants: Uncertainty in Text Classification with Monte Carlo Dropout. In: 2022 International Conference on Computer and Information Technology, pp. 7–12 (2022). https://doi.org/10.1109/ICCIT57492.2022.10055922. IEEE