Video Deepfake Classification Using Particle Swarm Optimization-based Evolving Ensemble Models

Li Zhang[1], Dezong Zhao[2], Chee Peng Lim[3], Houshyar Asadi[3], Haoqian Huang[4], Yonghong Yu[5], and Rong Gao[6]

[1]Department of Computer Science
Royal Holloway, University of London
Surrey, TW20 0EX, UK

[2]James Watt School of Engineering
University of Glasgow
Glasgow, G12 8QQ, UK

[3]Institute for Intelligent Systems Research and Innovation
Deakin University
Waurn Ponds, VIC 3216, Australia

[4]College of Energy and Electrical Engineering
Hohai University
Nanjing, 210098, China

[5]College of Tongda
Nanjing University of Posts and Telecommunications
Nanjing, 210023, China

[6]School of Computer Science
Hubei University of Technology
Wuhan, 430068, China

Email: li.zhang@rhul.ac.uk; dezong.zhao@glasgow.ac.uk; chee.lim@deakin.edu.au; houshyar.asadi@deakin.edu.au; hqhuang@hhu.edu.cn; yuyh@njupt.edu.cn; gaorong@hbut.edu.cn

Abstract.
The recent breakthrough of deep learning based generative models has led to the escalated generation of photo-realistic synthetic videos with significant visual quality. Automated reliable detection of such forged videos requires the extraction of fine-grained discriminative spatial-temporal cues. To tackle such challenges, we propose weighted and evolving ensemble models comprising 3D Convolutional Neural Networks (CNNs) and CNN-Recurrent Neural Networks (RNNs) with Particle Swarm Optimization (PSO) based network topology and hyper-parameter optimization for video authenticity classification. A new PSO algorithm is proposed, which embeds Muller's method and fixed-point iteration based leader enhancement, reinforcement learning-based optimal search action selection, a petal spiral simulated search mechanism, and cross-breed elite signal generation based on adaptive geometric surfaces. The PSO variant optimizes the RNN topologies in CNN-RNN, as well as key learning configurations of 3D CNNs, with the attempt to extract effective discriminative spatial-temporal cues. Both weighted and evolving ensemble strategies are used for ensemble formulation with aforementioned optimized networks as base classifiers. In particular, the proposed PSO algorithm is used to identify optimal subsets of optimized base networks for dynamic ensemble generation to balance between ensemble complexity and performance. Evaluated using several well-known synthetic video datasets, our approach outperforms existing studies and various ensemble models devised by other search methods with statistical significance for video authenticity classification. The proposed PSO model also illustrates statistical superiority over a number of search methods for solving optimization problems pertaining to a variety of artificial landscapes with diverse geometrical layouts.

Keywords: Video Deepfake Classification, Hybrid Deep Neural Network, 3D Convolutional Neural Network, Evolutionary Algorithm, Evolving Ensemble Classifier.

1. INTRODUCTION

Deep learning techniques have demonstrated significant advancement in video and signal processing. Deep generative methods (e.g., the Generative Adversarial Network (GAN) and its variants) show impressive capabilities in generating synthetic images, videos and audios with realistic forgeries. Example deepfake generation techniques include facial expression and identity manipulation, scene editing and static/animated content generation [1, 2]. These methods are capable of generating photo-realistic synthetic videos by enacting facial expressions from one person to another, swapping faces, inserting/deleting background scenes and synthesizing novel static views or animations [1, 2]. Besides that, medical deepfakes have also become prevailing, e.g., manipulated benign and malignant tumour images with respect to lung conditions [3]. Owing to the high quality photo-realistic forgeries in synthetic videos, deepfake detection is a challenging task for human observers, which may pose significant security and privacy threats. Because of the fast progression of the aforementioned as well as new deep learning generative models, the availability of manipulated videos has been significantly escalated [4-9]. As such, automated and accurate identification of video forgery is essential in tackling the above challenges.

In parallel, deep neural networks such as 3D Convolutional Neural Networks (CNNs) and CNN-Recurrent Neural Network (RNN) show great efficiency in tackling video classification pertaining to human action recognition [10-13]. In particular, 3D CNNs such as Inflated 3D ConvNet (I3D) [10-12] and MC3 [14] have great superiority in spatial-temporal feature extraction [7]. To take advantage of such pre-trained 3D CNNs on human action recognition, we conduct transfer learning using I3D and MC3 for video authenticity identification. Owing to the dominance of learning configurations, such as the learning rate, learning rate drop factor and regularization coefficient, to network performance as well as model capabilities in undertaking under-fitting and over-fitting problems [10, 14, 15], automated hyper-parameter optimization is desirable. In particular, with respect to CNN-RNN, since the types and topologies of RNN models, e.g. Gated Recurrent Unit (GRU), Long Short-Term Memory (LSTM), and bidirectional LSTM (BiLSTM) as well as their configurations [16-19], play important roles in extracting effective temporal cues, the identification of such key network structures that best adapt the RNN decoder to different synthetic video classification tasks is essential.

Therefore, in this research, we exploit weighted and evolving ensemble models incorporating 3D CNNs and CNN-RNN with Particle Swarm Optimization (PSO)-based optimal network structure and hyper-parameter selection for video authenticity classification. Specifically, Inceptionv3-RNN is utilized as the encoder-decoder architecture while I3D and MC3 are used as the 3D CNNs for synthetic video identification, owing to their superior capabilities in spatial-temporal dynamic extraction in solving video classification tasks [11, 12, 13]. Moreover, a PSO variant is proposed to optimize the architectures of Inceptionv3-RNN as well as key hyper-parameters of I3D and MC3 with the attempt to extract fine-grained discriminative spatial-temporal cues. Precisely, we optimize the network type (i.e. GRU, LSTM, and BiLSTM) and structure (i.e. number of hidden units) of the RNN decoder in Inceptionv3-RNN. In addition, optimal learning parameters (i.e., the learning rate, learning rate drop factor and regularization coefficient) in I3D and MC3 are automatically identified using the proposed PSO method to tackle the laborious constraints of manual parameter selection. The new PSO algorithm embeds Muller's method and fixed-point iteration based leader enhancement, a petal spiral simulated search mechanism, 3D geometric landscape inspired cross-breed leader generation, and reinforcement learning (RL)-motivated sequential search deployment, to optimize the aforementioned hyper-parameters and RNN types and structures pertaining to video forgery identification.

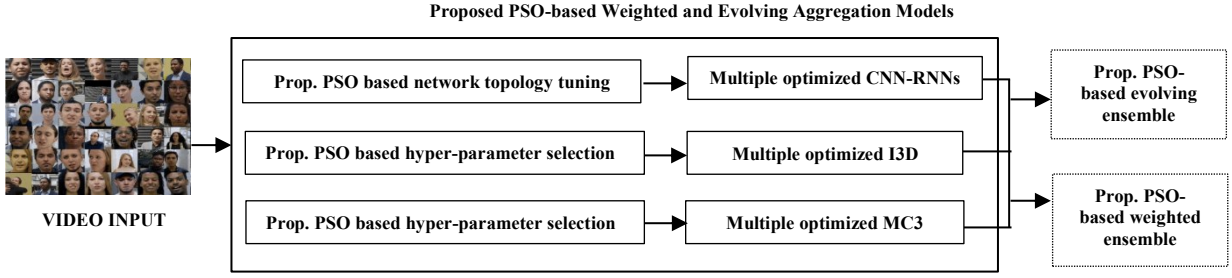

Figure 1 The proposed PSO-based weighted and evolving ensemble models integrating optimized I3D, MC3 and CNN-RNN for video authenticity classification

Moreover, two strategies are used for ensemble model formulation, i.e. a weighted ensemble scheme and an optimization-based evolving ensemble strategy. The former is able to effectively tackle class imbalanced classification problems. In the latter, optimal subsets of optimized Inceptionv3-RNN, I3D and MC3 networks are selected by the proposed PSO algorithm for ensemble model formulation. Figure 1 depicts the proposed PSO-based weighted and evolving ensemble models by integrating optimized Inceptionv3-RNN, I3D and MC3 networks for synthetic video classification.

A number of novel aspects of this research are listed, as follows.

1. Owing to the great efficiency in extracting spatial-temporal cues of Inceptionv3-RNN, I3D and MC3, we propose weighted and evolving ensemble models integrating these deep networks with PSO-based network topology and hyper-parameter fine-tuning for video authenticity classification. A new PSO variant is proposed to optimize different types of RNN layers (i.e. LSTM, BiLSTM and GRU) and the number of hidden neurons of the selected recurrent layer. On the other hand, the proposed PSO algorithm is also employed to identify optimal settings of the learning rate, learning rate drop factor and regularization coefficient in I3D and MC3, respectively. Diverse optimized I3D, MC3 and CNN-RNN models are subsequently used to construct the base classifier pool. Besides using a weighted ensemble scheme, the proposed PSO variant is used to conduct ensemble model formulation by selecting optimal subsets of these optimized classifiers to balance between ensemble complexity and performance.
2. The new PSO variant overcomes limitations of the original PSO operation [20] by exploiting fixed-point iteration and Muller's method for swarm leader enhancement, reinforcement learning-inspired search strategy selection, cross-breed elite signal generation based on adaptive 3D geometric landscapes, as well as a petal spiral-based intensification. Specifically, the aforementioned mathematical root finding mechanisms for leader enhancement employ recursive division strategies to better estimate global minima to guide local exploitation of the swarm leader. A total of four sets of adaptive 3D geometrical landscape oriented crossover formulae are proposed to generate diverse cross-breed leaders to enhance global exploration. In addition, a petal simulated spiral search action is utilized to exploit local jumps by using a petal-spiral search trajectory. To optimize search behaviours of each particle, a reinforcement learning algorithm is used to identify the optimal sequential deployment of the above local and global search operations. Overall, these search strategies coordinate with one another to increase search territories, diversify elite signal generation, steer effective guided search action selection and intensification of the swarm leader, to mitigate local optima traps.
3. In this research, we employ two schemes for ensemble network construction, i.e. (1) a weighted scheme, i.e. ensemble scheme 1 (as discussed in Section 5.1), and (2) an evolving ensemble generation scheme devised by each optimization algorithm, i.e. ensemble scheme 2 (as depicted in Section 5.2). The former is able to effectively tackle class imbalanced classification problems, while the latter is able to eliminate weak or redundant base classifiers to minimize ensemble complexity while maximizing performance. We employ the weighted ensemble scheme 1 to formulate a pool of models with the same types of base networks, while the evolving ensemble scheme 2 using optimization algorithms is used to devise a pool of classifiers integrating different types of base networks. In both schemes, the aggregation of the proposed PSO-optimized CNN-RNN and 3D CNN models with different learning mechanisms, network topologies and parameter configurations further improves the ensemble performance. In particular, for evolving ensemble scheme 2, moderate numbers of base classifiers are selected to achieve a balance between ensemble complexity and performance, as compared with those devised by other search methods.
4. Evaluated using three well-known video datasets, i.e., Celeb-DFv2 [1], FaceForensics++ [2] and Deepfakes [2], for both weighted and evolving ensemble schemes, the proposed PSO-based ensemble models outperform those (with optimized learning settings or optimal base classifier subsets) yielded by other search methods with a statistical significance. Our ensemble networks also show better performance than those of existing state-of-the-art methods for video authenticity classification. The proposed PSO variant also illustrates statistically better performance than those of a number of classical and advanced search methods for solving numerical benchmark functions with challenging artificial landscapes.

The remaining sections are arranged as follows. In Section 2, we introduce state-of-the-art deep learning methods and PSO variants for solving synthetic video classification and industrial optimization problems, respectively. Section 3 elaborates the proposed PSO variant with numerical analysis methods, petal-driven local exploitation, synthetic signals for global exploration, as well as reinforcement learning-based search action selection. We conduct hyper-parameter and network topology optimization for 3D CNN and CNN-RNN models as well as weighted and dynamic ensemble formulation in Sections 4 and 5, respectively. Performance

comparison between the proposed PSO algorithm and other search methods for ensemble model construction with respect to deepfake detection is provided in Section 6. Finally, research findings and future inspirations are summarized in Section 7.

2. RELATED WORK

We analyse a variety of state-of-the-art deepfake video classification methods as well as swarm intelligence and evolutionary algorithms for solving diverse industrial and real-world optimization problems in this section.

2.1 Deepfake Detection

The escalated synthetic video and audio generation poses significant cybersecurity threats for a number of applications, e.g., robotic navigation, auto-pilot, social media, and medical diagnosis. As an example, fake video and audio clips could be used to conduct personal attacks via social media and influence political elections, public views and medical diagnosis and treatment [6]. Accurate synthetic video classification therefore is crucial in detecting such cyber-attacks. A number of related studies were developed in recent years for fake/real video classification. Zhang et al. [15] utilized a new 3D CNN model, which embedded a 3D inception module and temporal dropout, for synthetic video classification. The network adopted inconsistent subtle spatial-temporal hints between video frames to distinguish fake from real videos. Specifically, the inception module was used to extract multi-scale spatial features and better capture the inconsistent signals in the video sequences, while the dropout function performed random sampling to randomly remove some frames from the raw video volumes for deepfake classification. Their network effectiveness was tested using three benchmark synthetic video datasets. The model effectiveness could be attributed to the extraction of temporal inconsistencies through a 3D inception module and the adoption of a temporal dropout scheme to better preserve local and global temporal information. Nonetheless, their work only evaluated the aforementioned strategies on a single 3D CNN architecture, without generating more diversified video presentations by integrating with other 3D CNN models, which could limit their model performance. Optimization algorithms could be used to identify optimal settings of their proposed temporal dropout strategy as well as other key hyper-parameters to further increase network robustness.

In addition, since biological signals embedded in the original videos were difficult to preserve in synthetic contents, Ciftci et al. [7] exploited a scheme to capture such biological cues to distinguish fake from real videos. Precisely, their work extracted biological signals from both real and synthetic videos pertaining to the facial regions (i.e., left and right cheeks and the upper nose region). Then signal transformations to time, frequency, time-frequency domains were performed respectively based on these extracted biological hints, to calculate spatial-temporal consistencies and correlations for pairwise classification. The respective biological properties and indicator maps were subsequently constructed and used to train the Support Vector Machine (SVM) and CNN to identify fake from authentic videos. Ensembling of the outputs of each video segment via a majority voting mechanism was used to determine the final fake/real video prediction. Their work exploited the advantage of a complex biological signal extraction process to learn features from the green colour channel in image frames for spatial-temporal consistency identification. Different SVM classifiers were trained using different sets of biological signals. But the complementary nature of these extracted features could be further strengthened using an evolutionary algorithm-based ensemble construction scheme instead of using a traditional majority voting method. This could result in the identification of effective and dynamic sets of base classifiers to improve robustness. In addition, instead of using an ImageNet pre-trained deep 2D CNNs or 3D CNNs, a comparatively shallow 2D CNN model with three convolutional blocks was used for video authenticity classification in their work, which could show limited capabilities in learning high-dimensional features from multiple video frames.

Chintha et al. [4] employed a Convolutional Recurrent Neural Network (CRNN) with XceptionNet as the spatial feature encoder and BiLSTM as the temporal decoder for authentic video classification. Cross-entropy loss, Kullback-Leibler (KL) divergence loss, and their combinations were studied in their experiments for visual deepfake classification. In addition, a CRNN, i.e. 1D convolutions combined with BiLSTM layers, as well as a residual network with WideBlocks, is utilized for authentic audio identification. Evaluated using a number of benchmark video (FaceForensics++ and Celeb-DFv1) and audio (ASVSpoof 2019) datasets, their networks obtained improved performance as compared with those of existing studies. The appealing aspect of their work was the employment of both visual and auditory inputs for video authenticity classification. Two sets of video and audio-based networks were developed respectively to tackle deepfake detection. However, the configurations of the hidden layer structures (e.g. the number of hidden units) of the BiLSTM models in both XceptionNet-BiLSTM and CRNN for video and audio classification respectively, were pre-determined and fixed, which could be dynamically optimized to better adapt to different video/audio classification tasks.

Wang et al. [5] utilized a novel Siamese network for manipulated video classification. A pairwise input of the original and manipulated images was exploited to generate two segmentation maps. The localization consistency of the two maps was subsequently calculated using an invariance loss. The extracted feature maps and the segmentation maps were then used as the inputs to a mask-guided transformer to generate co-occurrence characteristics, which were then used to train a feedforward shallow neural network for video authenticity classification. Evaluated using FaceForensics++ and Celeb-DFv2, their network showed competitive performance for both within and cross-dataset evaluations. The advantage of their model lied on the employment of segmented features from the manipulated regions in conjunction with the feature maps extracted from the whole authentic image, which were used as inputs to a transformer for more informative feature representation generation. But their model operated as a frame-level method and could not be easily deployed for video classification tasks. In addition, it purely employed RGB image frames as inputs without the consideration of temporal details, which could limit their model flexibility.

Pu et al. [21] proposed a ResNet50-GRU encoder-decoder architecture for real and manipulated video classification. The extracted features from ResNet50-GRU were used for both video-level and image-level detection. For video classification, the extracted spatial-temporal patterns from the encoder-decoder architecture were passed on to the pooling, flatten and dense layers for authenticity classification, while for image-level classification, the extracted dynamic temporal features from each frame were used as the inputs to a fully connected layer for the classification of manipulated content at the frame-level. A joint loss function was used for network training. Their model showed improved capabilities in tackling imbalanced and cross-dataset evaluation. Their work combined image-level and video-level streams for video authenticity classification. Nonetheless, the network configurations of the GRU decoder in ResNet50-GRU were pre-defined with fix settings for evaluating different video deepfake datasets. The decoder network could benefit from the incorporation of evolutionary or reinforcement learning algorithms to generate dynamic adaptive settings to improve network robustness. Besides the adoption of a CNN-RNN architecture, their work could also take 3D CNNs into account to increase classifier diversity.

Motivated by the combination of the original and manipulated contents in a forged video, Shang et al. [22] exploited a dual relation network to identify pixel-wise and region-wise relations for authenticity classification. The model employed the Pixel-Wise Relation (PR) module to extract similarity information between pixels. The PR module also extracted feature representations of the original and forged regions using an attention layer, respectively. The Region-Wise Relation component was subsequently used to identify incoherent characteristics between the two extracted feature maps using multiple metrics. Such inconsistency comparison was used to inform manipulated video classification. The appealing aspect of their work was the integration of the aforementioned pixel and region-wise components for authenticity classification. But owing to inconsistency measurement at the pixel or region levels, the model showed limited capabilities in identifying completely synthesized fake images.

Wang et al. [23] conducted video authenticity detection by using a two-stream architecture fusing spatial and frequency-based subnetworks. Global and multi-scale shallow spatial features were extracted using the spatial stream, while frequency related features such as amplitude and phase properties were obtained using the frequency subnetwork. Early fusion of the two streams was performed by concatenating the yielded spatial and frequency characteristics, which were subsequently used to inform contrastive learning and video forgery detection. Their model yielded an impressive performance for evaluating several large-scale benchmark deepfake datasets. The work employed concatenated features from both spatial and frequency-based networks to increase robustness. However their model required additional pre-processing efforts in generating amplitude and phase information from the original image inputs based on the discrete Fourier transform (DFS) and could incur additional computational cost in comparison with those of existing 2D or 3D CNNs. Moreover, their model purely focused on frame-level classification, without considering sequential temporal details.

Chen et al. [24] developed Xception-ConvLSTM with attention mechanisms for video forgery detection. A new attention component was exploited with the attempt to preserve sufficient spatial-temporal cues before applying feature reduction of the CNN-RNN operations. In particular, their model extracted intra-frame and inter-frame correlations using spatial and temporal attention functions respectively. The yielded feature maps were subsequently used as the inputs to Xception-ConvLSTM for forged video classification. The ConvLSTM structure was also capable of extracting more refined spatial-temporal patterns. Their model showed enhanced classification accuracy rates when evaluated using sample videos extracted from several deepfake datasets (e.g., Celeb-DFv2 and Deepfake Detection Challenge (DFDC)). The key contribution of their work was the proposal of spatial-temporal attention mechanisms for feature learning. However, the model depicted limited robustness owing to the excessive extraction of intra- and inter-frame features using their attention models. In addition, the

key learning configurations of ConvLSTM, such as the dropout rate, number of hidden units and filter sizes and numbers, were pre-determined, which could actually be fine-tuned using swarm intelligence algorithms or reinforcement learning methods, in order to increase network robustness.

Moreover, capsule networks with features maps yielded by VGG19 were studied by Nguyen et al. [25], and ResNet50 trained with photo-realistic images generated using PGGAN was examined by Wang et al. [26] for evaluating model generalization capabilities using videos synthesized by other generative models. SCnet with embedding of a set of stacked convolutional layers was developed by Guo et al. [27] for video authenticity identification. A number of other exiting studies on video deepfake generation and classification have also been analysed in detail by Nguyen et al. [8]. Besides manipulated video classification, forged audio classification [28] has also been conducted using a variety of deep networks such as BiLSTM [29], Deep4SNet [30], MesoInception-4 [31], and Xception [31] for evaluating FakeAVCeleb [32] and ASV spoof 2019 [33] datasets.

Table 1 depicts the key methodologies of the aforementioned existing studies. As indicated in Table 1 and other existing works, the research gaps for deepfake detection are as follows. (1) 2D CNNs and CNN-RNN models have been commonly adopted in many existing works, while comparatively few studies have employed 3D CNNs. (2) Very few studies have combined optimization algorithms with 3D CNNs or CNN-RNNs. As an example, most methods have employed fixed RNN settings in CNN-RNN or CRNN models without considering optimizing the recurrent layer types and hidden unit configurations using evolutionary algorithms. (3) Early feature-level fusion and traditional majority voting methods are mainly adopted in existing studies. Dynamic ensemble model formulation using evolutionary algorithms is rarely exploited for video classification. (4) Some methods require additional pre-processing (e.g. frequency feature generation using DFS and spatial/temporal attention components for feature extraction), leading to additional computational cost and hindering their deployment in real-world settings.

To bridge research gaps, we employ three models, i.e. CNN-RNN (Inceptionv3-RNN) and two 3D CNNs (I3D and MC3) for deepfake classification in this study. A new optimization algorithm integrating PSO with the reinforcement learning (Q-learning) algorithm is proposed for recurrent layer configuration and hyper-parameter optimization for CNN-RNN and 3D CNNs, respectively. The optimization process enables these networks to adapt to different deepfake detection tasks effectively. These optimized networks are then used for ensemble model generation. Specifically, two new schemes are devised in this research for ensemble model formulation, i.e. (1) a weighted scheme, i.e. ensemble scheme 1 (as discussed in Section 5.1), and (2) an evolving ensemble generation scheme devised by each search algorithm, i.e. ensemble scheme 2 (as depicted in Section 5.2). The former is able to effectively tackle class imbalanced classification problems, while the latter is able to eliminate weak or redundant base classifiers and to minimize ensemble complexity while maximizing performance. To the best of our knowledge, our research on combining PSO variant with reinforcement learning for both hyper-parameter optimization and evolving ensemble formulation for video authenticity classification is new, therefore our contribution.

Table 1 Comparison of key methodologies between this research and existing studies

	Key methodologies	Optimization, ensemble or other strategies	Limitations and/or mitigation strategies
Zhang et al. [15]	Single 3D CNN with temporal dropout and a 3D inception module	-	No other 3D CNNs or hybrid networks used
Ciftci et al. [7]	Biological signal extraction + a shallow 2D CNN + SVM-based ensemble with a traditional majority voting method	Traditional majority voting ensemble	No dynamic/evolving ensemble model construction using evolutionary algorithms considered
Chintha et al. [4]	XceptionNet-BiLSTM with multiple losses for video classification + CRNN and CNN for audio classification	-	Fixed layer and neuron settings of the BiLSTMs in both CNN-RNN and CRNN without dynamic adaptation or optimization.
Wang et al. [5]	A Siamese network integrated with image segmenter and transformer-based feature learning + a feedforward shallow neural network for classification	A frame-based method purely based on RGB frame input	No consideration of temporal inputs and cannot be easily deployed for video classification tasks.
Pu et al. [21]	A ResNet50-GRU combined with image-level and video-level classifiers	-	Pre-defined fixed settings of the GRU decoder in ResNet50-GRU without optimization
Wang et al. [23]	A two-stream architecture fusing spatial and frequency-based subnetworks.	DFS used for additional amplitude and phase information extraction	A frame-based method without the consideration of temporal cues

Chen et al. [24]	Spatial-temporal attention mechanisms + Xception-ConvLSTM	Intra- and inter-frame correlations extracted using attention methods	Limited robustness owing to the excessive extraction of intra- and inter-frame features. No optimization applied to key settings of ConvLSTM.
Shang et al. [22]	A dual relation network to identify pixel-wise and region-wise relations	Pixel-wise and region-wise inconsistency identification	Limited capabilities in identifying completely synthesized fake images.
This research	CNN-RNN, I3D and MC3 with evolutionary algorithm-based hyper-parameter optimization and ensemble network construction	Diverse optimized CNN-RNNs and 3D CNNs are devised for weighted and evolving ensemble generation. Optimal hyper-parameter and base classifier selection performed using a newly proposed optimizer.	A new PSO variant combined with reinforcement learning is proposed for hyper-parameter optimization and weighted and evolving ensemble model construction.

2.2 Particle Swarm Optimization and Its Variant Methods

Evolutionary algorithms have been intensively deployed in a variety of industrial optimization problems, such as robot and drone navigation, path planning, job scheduling and energy efficiency applications. As a typical swarm intelligent algorithm, PSO [20] exploits the search space by using personal and global best solutions, as indicated in Equations (1)-(2).

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \tag{1}$$

$$v_{id}^{t+1} = w \times v_{id}^{t} + c_1 \times r_1 \times \left(p_{best\_id} - x_{id}^{t}\right) + c_2 \times r_2 \times \left(g_{best\_d} - x_{id}^{t}\right) \tag{2}$$

In Equation (1), the position and velocity of particle i in the d-th dimension and $t+1$-th iteration are denoted as x_{id}^{t+1} and v_{id}^{t+1}, respectively. The personal and global best experiences, i.e. p_{best_i} and g_{best}, along with their respective acceleration coefficients, c_1 and c_2, are used to dominate the search behaviours of the cognitive and social elements, respectively. The effect of the previous velocity to the new one is determined by the inertial weight w. To diversify the search steps of the swarm particles, random vectors, r_1 and r_2, are also used.

We use the random vectors r_1 and r_2 to diversify the search steps for each particle. They are generated using the same dimension as that of the swarm particles. Each element of r_1 and r_2 is assigned with a random value in the range of [0, 1], so that it randomizes the search step in each dimension to increase robustness. Precisely, r_1 is used to randomize the search step in each dimension for the particle to move towards the personal best solution, while r_2 randomizes the search step for moving the current particle towards the global best solution. Indicated by existing studies [34], instead of using a fix random value for all the elements for r_1 and r_2, respectively, the random vectors with a different random value in each dimension may help increase search diversity to mitigate local optima traps.

In addition, since PSO relies on the personal and global best solutions to navigate particles in the search space, it is likely to converge prematurely. Many existing studies have proposed variants of PSO as well as new swarm intelligence algorithms to overcome such limitations. For example, Zhang et al. [35] developed a PSO variant with automated subswarm topology and parameter adaptation. To increase search diversity, subswarms were automatically formed using the K-means (KM) clustering algorithm where the Calinski-Harabasz (CH) index was applied to determine the number of subswarms. Adaptive search coefficients were deployed to fine-tune the model behaviours, while Bayesian optimization was also exploited to optimize the boundaries of these search hyper-parameters. Besides the original PSO velocity update, particle velocity calculation was also affected by the subswarm leaders. Their model showed a better performance for solving standard and complex numerical optimization problems and hyper-parameter optimization for image enhancement than those of other existing PSO variants. The appealing aspects of their work were the CH index based automatic generation of the optimal number of subswarms, and parameter search boundary determination using Bayesian optimization for enabling a better trade-off of local and global search behaviours. While both the subswarm leaders and the global best solution were used for velocity update, these important optimal leader signals were not further enhanced using any random walk or other strategies. In addition, their work only employed a single search operation integrating the respective subswarm leader and the global best solution for velocity update, without the adoption of any alternative search mechanisms. If this single search action was trapped in local optima, there were no alternative search operations for activating the swarm to explore other regions to overcome stagnation.

Zhang et al. [36] studied a PSO model with mutation operators and adaptive search coefficients in combination with Monte Carlo search based error simulation for solving precision airdrop tasks. In their PSO algorithm, the

weighting of the previous velocity was adaptively adjusted to the fitness difference between each particle and the swarm leader, divided by the fitness difference between the mean position of the overall swarm and the swarm leader. The flexibility of the velocity formula was further enhanced using an additional mutation operator. Evaluated using challenging real-world cases, their model showed enhanced precision for airdrop assignment as compared with those of several existing PSO variants. The advantage of their studies was the adoption of the adaptive inertia weight coefficient fine-tuning the new velocity generation, as well as the additional random mutation operator for velocity diversification. Although such a random-based mutation operator could increase search diversity, it incurred additional search iterations to achieve convergence. Instead of using a random operator, more informative mathematical numerical analysis or Gaussian-based fitness estimation methods could be used to implement the mutation action and to accelerate convergence while increasing search territories. Similar to the above studies, a single search operation integrating the original PSO operation with the new random operator was used for velocity update. No additional alternative search mechanisms and multiple leaders were considered. When this single search strategy led by the swarm leader became stagnant, no additional momentum was injected via the execution of other search actions guided by other elite signals to drive the swarm out of stagnation.

Liang et al. [37] utilized an enhanced PSO algorithm in combination with Fuzzy C-Means (FCM) clustering for estimating air quality. Specifically, the weight matrix of FCM was optimized by their PSO method. In their PSO variant, the inertia weight was diversified in accordance with the fitness score of each particle and further randomized in each search dimension. To enhance local exploitation of each search agent, another particle was randomly selected from the same subswarm to conduct Differential Evolution (DE)-based position updating. In addition, the global search operation of their model was improved by using two subswarms where the subswarm leaders were used in conjunction with the personal best experiences for position updating. The switching between the original PSO operation and this new subswarm leader-based search operation was controlled by a random factor. Evaluated using several air pollution datasets, their hybrid model outperformed other search methods inspired FCMs in terms of several error metrics. The advantage of their model was the employment of the DE-based local search strategy to overcome limited local search capabilities of the PSO algorithm. However, the determination between the new subswarm leader-based global search action and the original PSO operation was purely conducted based on a random factor instead of using more informative strategies (e.g. reinforcement learning), therefore limiting their model performance. In addition, no additional operations were exploited for swarm or subswarm leader enhancement.

Liu et al. [38] exploited a PSO variant with both attraction and repulsion behaviours. Specifically, the velocity update of each particle was guided by the personal best experiences of the overall swarm. Each particle was attracted to the fitter personal best individuals and repelled from the less optimal personal best experiences. Next, the personal best solutions of the overall swarm were randomly paired. Information exchange was conducted between each pair of such randomly selected search agents. Precisely, a crossover operator was used to generate offspring of the selected personal best solutions, which were used to replace weaker parent chromosomes. Evaluated using a set of 22 hybrid and rotated benchmark functions, their algorithm outperformed several existing PSO variants. Their work leveraged information exchange between personal best individuals to generate new leader signals. Owing to the guidance of all the personal best experiences in the overall swarm for the position update of each particle, such a position update is computationally costly. In addition, the approach could lead to oscillatory behaviours of each particle by following all the personal best individuals of the entire swarm. Moreover, there was only a single search action used for position update without the consideration of additional position updating mechanisms. Their model could therefore highly likely be trapped in local optima when their main search operation became stagnant, owing to the unavailability of alternative search operations to activate sudden movements of the swarm.

Neuroscience inspired PSO was developed by Liu et al. [39]. Besides the traditional PSO velocity formula, an additional nervous guidance scheme was proposed for velocity updating. Randomly generated and immune orientation based leaders were used to lead this new search action. Such an additional term enabled the extension of the search territory with better capabilities in tackling early stagnation. The switching of this new velocity updating action and the original PSO operation was conducted based on a random factor. An immune orientation component was also used with a local optimal guiding signal to update the particle movement. Adaptive coefficients were also generated using an endocrine regulation mechanism to adapt the search process to several key search stages. Complex numerical optimization and real-world scenarios were utilized for model evaluation. Their work had the benefit of having an additional nervous guidance velocity updating formula in conjunction with the original PSO operation. Nevertheless, the activation of this new search action on top of the original PSO operation was based on a random factor without taking other informative strategies into account, such as

convergence checking or reinforcement learning-inspired methods, which could limit their model robustness and affect its convergence speed.

Lu et al. [40] studied a multi-subswarm PSO algorithm where several cooperative strategies were deployed to increase subswarm diversity in accordance with diverse search circumstances. Four subswarms were formed in the initial search stage. A stagnation inspection component was also utilized to examine the stagnation status of the subswarm leaders. A new exemplar solution was also constructed based on the global best solution and the stagnant subswarm leader to lead and re-activate the stagnant subswarm. Specifically, each dimension of the new leader signal was generated based on either the crossover operation using the swarm leader and the stagnant subswarm leader or a random initialization process. In addition, the former crossover operation for combining two leaders was also performed using randomly generated weighting factors. Moreover, a re-initialization method was also used to re-dispatch particles in other subswarms to remote regions when they showed high position proximities to those of the most optimal subswarm. Because of the increased search areas using the above search schemes, their PSO variant achieved a good performance for solving the CEC2017 test suite. The advantages of their study included stagnation checking strategies and construction of combined leaders for re-activating respective stagnant subswarms. However, in each dimension, their activation leader generation was based on either a random initialization process or a random combination of the stagnant subswarm leader and the global best leader using random weighting coefficients. The switching between these strategies was also based on the position closeness checking between subswarms. Therefore, when the former random initialization operation became dominating, the resulting new leader solution was likely to be randomly assigned, which could not guarantee to be an optimal signal. On the other hand, if the latter crossover process became dominating, because of the random allocation of weighting factors of the parent individuals, the new combined leader could still inherit many characteristics from the stagnant subwarm leader, therefore could not be effective enough to divert the subswarm out of stagnation. Moreover, since the dispatch re-initiation operation in their model was activated based on the position closeness checking between subswarms, customization effort was needed for different optimization contexts.

Li et al. [41] exploited a dynamic learning PSO model. Firstly, the swarm distribution was examined and used to determine different search stages. Next, in accordance with distinctive search stages, the cognitive and social search parameters were dynamically adjusted. The effect of the previous velocity was also fine-tuned in accordance with the swarm distribution. DE was applied to further improve the personal and global best solutions, to diversify the leader signals. Besides that, particles in the direct neighbourhood of these optimal indicators, i.e. personal and global best individuals, were used to replace them at different search stages for velocity update. To further diversify the search process, a compound signal integrating two elite solutions was also produced by using a dimension-based scheme. Evaluated using a set of 12 numerical optimization functions, their model showed a superior performance over those from five baseline PSO variants. The appealing aspects of the work included DE-based optimal individual enhancement and search coefficient and leader signal allocation corresponding to different search stages. One shortcoming was the pure adoption of the original PSO operation for velocity update without the integration of additional search mechanisms to diversify the search behaviours. As such, there was no alternative operation provided to inject momentum if the current PSO operation became stagnant before the search switched to the next stage. In comparison with reward-motivated schemes, a distance-based strategy was used to determine different search stages for search operation allocation, which required customization according to different optimization tasks. The Q-learning based search coefficient and action selection could also be considered to further increase robustness.

Chen et al. [42] diversified the original PSO operation using crossover formulae for elite signal generation. Such offspring compound signals were produced with the personal best solutions as the parent chromosomes. Stagnation monitoring was also performed. A fitter cross-breed leader was randomly selected to guide the search process when the current compound signal did not show improvement for several iterations. Their PSO algorithm obtained better accuracy rates in solving a number of mathematical landscapes as compared with those from several PSO variant methods. The integration of diverse personal best solutions for combined leader solution generation was advantageous to increase search robustness. However, the hybrid leader generation process employed random weighting coefficients for integrating two leader individuals, which could affect the balance between exploitation and diversification. Instead of using multiple distinctive combined leader signals to guide each particle, their work purely relied on single combined leader for position update for the entire swarm. Moreover, their model only employed single search operation to guide the search process. Alternative search strategies could be used in conjunction with diversified leader generation processes to better tackle stagnation. Besides the above studies, a variety of other PSO algorithms were also exploited for optimizing hyper-parameters in CNNs and CNN-RNNs with respect to skin lesion and pathological brain classification [43, 44,

45], optical disc segmentation [46], video action recognition [13, 47], and environmental, respiratory and heart sound identification [18].

There are also other automated machine learning (AutoML) methods for network architecture and hyper-parameter optimization. As an example, Lorenzo et al. [48] conducted optimal hyper-parameter selection for shallow CNN architectures using the original PSO algorithm. The model was used to optimize hyper-parameters, i.e. the numbers of filters and filter sizes of the convolutional and pooling layers, in simple CNNs with one convolutional block and 1 to 3 additional convolutional layers, another simple CNN with two convolutional blocks, as well as a LeNet-4 model. Their work investigated the effects of different settings of population and network depth to network performance for evaluating the MNIST and CIFAR-10 datasets. The experimental studies indicated that the network performance improved by increasing the network depth. In addition, despite the enhancement of optimization robustness with the increase of the population size for different image classification tasks, the effectiveness of using a small population size (e.g. 4 particles) for optimal parameter selection was also evidenced in their empirical studies. The work benefited from the investigations of optimizing hyper-parameters for different CNN architectures, but the employed network architectures were comparatively shallow dedicated to comparatively less challenging image classification tasks. Moreover, only the original PSO algorithm was used in their work for hyper-parameter search without the integration with any new search mechanisms or swarm leader enhancement strategies. If the original PSO algorithm was trapped in local optima, there were no alternative search operations to help overcome stagnation, which could limit their model performance.

Lorenzo et al. [49] exploited a parallel PSO for automated hyper-parameter optimization using shallow CNN architectures. Their parallel PSO model incorporated the original PSO operations but conducted the fitness evaluations of multiple particles in parallel. It optimized the filter sizes and numbers of filters of the convolutional and pooling layers as well as the number of nodes in the fully-connected dense layers in a shallow CNN and a LeNet-4 model. CNN models with optimized hyper-parameters were archived so that fitness scores could be extracted from those of the archived models directly when the newly optimized networks presented the same network configurations, in order to save cost. Two shallow CNN models with one convolutional block (i.e. the simple CNN) and two convolutional blocks (i.e. LeNet-4) were used as the base architectures. Evaluated using MNIST with a resolution of 28x28, the parallel PSO model obtained a better performance than the sequential/conventional PSO. However, their evaluation studies were only performed using comparatively shallow CNN models with 1-2 convolutional blocks on a comparatively simple image classification task. In addition, as the complexities of the optimization targets and test datasets increased, their parallel fitness evaluation strategy could be heavily constrained on the GPU resources available. The archive process of storing optimized parameters could also become resource consuming with reduced effects owing to the complexity of the high-dimensional search space and significantly diversified optimal network architectures. Their work also did not embed new search strategies to overcome limitations of the original PSO operation. Deeper neural networks including hybrid architectures (CNN-RNN) and those with residual connections in conjunction with large-scale datasets could be used to further evaluate the effectiveness of their proposed model.

Junior and Yen [50] developed a customized PSO operation for CNN architecture generation. An encoding mechanism was exploited to represent the respective deep neural architecture denoted by each particle. Search operations for velocity updating and position difference calculation were developed to allow the generation of new offspring networks based on the existing parent solutions. Despite the improved performance for evaluating on several small-scale image classification datasets, their customized search operations were not only constrained to specific CNN architectures, but also could not prevent the generation of invalid deep architectures. Their encoding and optimization processes also could not be easily extended to other types of networks such as RNNs, and CNN-RNNs and 3D CNNs. Lawrence et al. [51] performed residual network architecture generation using PSO. A new encoding scheme was developed for the swarm particles, with particle sub-dimensions representing the number of network clusters, the number of residual blocks within each cluster, the number of output channels, filter sizes and pooling types. Bespoke search operations were also designed to allow velocity updating between particles representing distinctive residual network architectures. Evaluated using MNIST and its variant datasets, their model outperformed other existing studies. But their encoding scheme was only dedicated to specific residual architectures with limited flexibilities of scaling the optimization process to completely distinctive or other more complex architectures.

Moreover, proposed by Baker et al. [52], reinforcement learning methods such as Q-learning were used for CNN architecture optimization in MetaQNN. The architecture generation process was guided by the Markov decision process and the Bellman equation, which maximized the expected reward return of the generated optimal policy (i.e. network architectures). However, owing to the generation of an effective Q-table for each pair of action-

state combinations, their model required a significantly expensive computation cost, consuming 10 GPUs even for a small-scale image classification dataset (e.g. CIFAR-10). Zoph and Le [53] performed RNN architecture search with a policy gradient reinforcement learning method, REINFORCE. The architecture search was guided by the policy optimization strategies via reward and punishment principles. While their model also achieved state-of-the-art performance with an error rate of 3.65% for solving image classification using CIFAR-10, it required a large number of GPU resources (800 GPUs with 672 GPU hours) for optimized architecture generation. Such an RL-based optimization process using Q-learning, REINFORCE, Proximal Policy Optimization (PPO) and Deep Deterministic Policy Gradient (DDPG) only generated one optimal solution, instead of a swarm of possible solutions as the cases for swarm intelligence algorithms. Since these RL methods were sensitive to the design of the reward schemes, instability in performance could occur [54-56].

Fielding and Zhang [43] performed evolving deep CNN architecture generation using a PSO variant with adaptive cosine search coefficients. A set of four convolutional blocks and a fully-connected block were optimized with each block containing 1-10 convolutional or fully-connected layers. A weight sharing mechanism was also employed to share network weights with generated similar architectures. In comparison with RL-based deep architecture optimization, their model achieved competitive performance with significantly reduced computational cost. But despite the proposal of cosine adaptive coefficients, their PSO algorithm largely relied on the original PSO operation, which could be further enhanced by adopting hybrid or multiple elite signals to better tackle local optima traps.

Tan et al. [45] developed a PSO variant with dimensional leader enhancement and random coefficients implemented using sine, cosine and circle formulae. It split the swarm particles into three subswarms and randomly generated a set of 10 offspring solutions using corresponding adaptive search parameters for each subswarm. The most optimal offspring individual was selected to replace the current particle in each subswarm. Partial dimensions of the swarm leader were updated in turn to improve the swarm leader. Their model outperformed classical and other PSO variants for deep architecture generation pertaining to skin lesion classification. The work took advantage of subswarm-based search processes with different search coefficients, but the position update operation in the each subswarm mainly depended on the original PSO operation, without any alternative new search strategies provided to help better deal with stagnation. Another PSO model was implemented by Tan et al. [57], which integrated multiple search actions led by randomly selected leader individuals, the mean solution of neighbouring fitter solutions, as well as the swarm leader for position update. Random walk and Genetic Algorithm (GA) operations were also used to improve top ranking particles. Their model employed multiple search actions with the attempt to better mitigate premature convergence, but the deployment of different search actions was performed based on a random selection instead of using RL-based strategies.

A Firefly Algorithm (FA) variant was also developed by Zhang et al. [58] for hyper-parameter optimization with respect to semantic segmentation. Neighbouring and randomly selected leader solutions were used to lead the top-ranking individuals in the swarm, while the remaining lower-ranking search agents were guided by three best swarm leaders integrated using diverse weighting coefficients to increase search flexibility. Their FA model was used to optimize key learning configurations of DeepLabV3+ for solving underwater image segmentation tasks. Their model increased search effectiveness by using multiple search actions led by distinctive elite leader signals. But the dispatch of different search actions was purely conducted based on a random factor. Fallahi et al. [59] utilized the Q-learning algorithm for search parameter generation for both PSO and DE, respectively. Their hybrid PSO and DE models were equipped with great flexibility owing to dynamic parameter generation. But in each of their hybrid models, the search process was largely dominated by either the original PSO operation guided by the swarm leader or the original DE operation, without additional search actions available. Therefore these single search actions could be prone to local optima traps.

Zhang et al. [60] exploited the integration of bare-bones PSO (BBPSO) with reinforcement learning (PSORL) for deep neural architecture generation for medical data classification. Besides the realization of search operations with local and global best signals, their PSO variant embedded the Q-learning algorithm to optimize the deployment of several root-finding algorithms for swarm leader enhancement. The effectiveness of PSORL-based deep networks was evidenced for evaluating diverse large-scale medical image datasets, coughing audio data for COVID detection as well as video action recognition. But in comparison with our studies in this research, their work deployed the RL strategies purely for the enhancement of several top-ranking particles, without customizing search actions of the majority of the swarm.

He et al. [61] studied a comprehensive survey of AutoML techniques for automated feature selection, deep architecture generation and hyper-parameter optimization without human intervention. Data oversampling

techniques such as SMOTE and data synthesis methods (such as GANs) were firstly studied to increase the sample sizes of the minority classes. A variety of augmentation techniques methods such as affine transformation, word embedding and noise injection and time shift for image, text, and audio data were discussed to help tackle overfitting. Feature optimization using feature ranking, Principal Component Analysis (PCA), and evolutionary algorithms were briefly addressed. Various variants of swarm intelligence algorithms have been studied for wrapper-based feature optimization. Architecture generation and hyper-parameter optimization for traditional machine learning methods (such as SVM) and deep neural networks (CNN and RNN) were investigated. Different types of architecture optimization methods were discussed including evolutionary algorithm (e.g. GA, Simulated Annealing (SA), PSO and FA), Bayesian optimization, random search, reinforcement learning (e.g. PPO and Q-learning algorithms), as well as hybrid methods (e.g. evolutionary algorithm combined with RL). Surrogate-based methods were also introduced to speed up the optimization process, where a surrogate model was used to replace the original objective function evaluation to guide architecture search and reduce computational cost. One-phase and two-phase strategies were adopted for optimizing deep networks. Specifically, the parameter search and final model evaluation were conducted in two processes in the two-stage method, while in one-stage method, these processes were conducted in parallel and the resulting optimized network did not require additional fine-tuning.

Another survey studies on AutoML techniques were conducted by Elshawi et al. [62]. A variety of deep architecture and hyper-parameter optimization methods were discussed, e.g. grid search, RL and swarm intelligence algorithms. Specifically, GA and SA were studied for various optimal hyper-parameter and network architecture identification processes, while the RL method, i.e. the Q-learning algorithm, was introduced for optimal network structure generation. Various challenges were summarized in their studies such as the limited scalability of many existing methods (e.g. Meng et al. [63]), no universal optimization strategies outperforming all other methods in any given optimization tasks, simple optimization targets (e.g. comparatively shallow network architectures), as well as expensive computational costs.

After the analysis of various existing studies, we identify the following research gaps, which motivate this research.

Despite the success of using RL algorithms such as Q-learning and PPO methods, as indicated in Baker et al. [52] and Zoph et al. [53], respectively, for architecture generation, these RL-based methods require substantial GPU time, which may not be affordable in most cases. The RL-based optimization processes are sensitive to the reward principle design, which may result in instability in cross-domain deployment. In addition, the RL algorithms usually recommend only one optimal solution, instead of multiple optimized solutions as compared with the case of evolutionary algorithms. Therefore, we propose a hybrid model combining RL with PSO in this research to diversify and stabilize the optimization process, while increasing search efficiency and generating multiple optimized solutions.

On the other hand, swarm intelligence algorithms have shown great efficiency in architecture and hyper-parameter search. Nevertheless, owing to substantial randomness embedded in diverse search operations in evolutionary algorithms introduced by random search coefficients (e.g. PSO, FA, Cuckoo Search (CS), DE, Dragonfly Algorithm (DA)), random search action deployment (Tan et al. [57]), mutation operations (e.g. GA), and random walk strategies (e.g. Levy and Gaussian distributions in FA, SA, Bat Algorithm (BA), Flower Pollination Algorithm (FPA)), more effective guiding strategies and mathematics-driven methods are required to increase search efficiency. As an example, although many new variants of PSO, FA, GA and other search methods have incorporated various search actions to overcome local optima traps, as indicated in Table 2, the majority of the aforementioned studies employ a random or threshold-based selection mechanism (controlled by distance thresholds and iteration numbers) to dispatch different search operations (e.g. Liu et al. [39], Lu et al. [40], Li et al. [41], Tan et al. [57], Zhang et al. [13], and Zhang et al. [58]), therefore comprising model efficiency.

To tackle the above limitations and increase search efficiency, we employ an RL algorithm, i.e. the Q-learning algorithm, in this research to dispatch distinctive search actions. This leads to the best long-term cumulative reward for each particle, instead of using random or threshold-based selection as in existing studies. Moreover, such RL-based optimal action selection is applied to each particle in each interaction to diversify swarm behaviours, instead of purely deploying it to several top-ranking individuals as in existing publications (e.g. Zhang et al. [60]).

To better tackle local optima traps, unlike many existing studies which rely on the original PSO operation guided purely by the swarm leader (Junior and Yen [50], Lawrence et al. [51], and Fielding and Zhang [43]), in

this research, diverse hybrid leaders are generated to lead distinctive local and global search actions, in order to further increase search diversity. Also, instead of using random factors for combining leader signals as in existing studies (e.g. Lu et al. [40] and Chen et al. [42]), adaptive weighting coefficients yielded using distinctive 3D formulae are utilized to better fine-tune the effects of the two leader signals, in order to gain a better balance between diversification and intensification.

On the other hand, random walk actions such as Levy, Gaussian and Cauchy distributions are often utilized in many classical and advanced search algorithms (e.g., Jordehi [64] and Zhang et al. [65]) for swarm leader enhancement. Instead of using such random jump operations, we exploit root-finding algorithms guided by the mathematical principles to provide more informative mechanisms for swarm leader enhancement.

Unlike existing studies where hyper-parameter search and architecture generation are normally performed for 2D CNNs for image classification in most cases, we leverage the proposed PSO variant to identify optimal hyper-parameters of hybrid networks (e.g. CNN-RNN) and 3D CNNs (I3D and MC3) for video classification.

In short, to tackle the limitations of RL (instability and sensitivity to reward strategy design and significant computational cost) and existing evolutionary algorithms (random operations for search action dispatch and swarm leader enhancement), in this research, a hybrid search algorithm integrating evolutionary algorithms and RL methods is formulated. RL is used to perform optimal local and global search action selection in our proposed model. Diverse hybrid leaders fine-tuned using adaptive weighting factors yielded by 3D contours in conjunction with swarm leader enhancement using mathematical informative root-finding algorithms are exploited to further increase search efficiency and robustness. Our empirical studies also further ascertain the effectiveness of the proposed hybrid PSO algorithm for hyper-parameter optimization in CNN-RNN and 3D CNNs, as well as evolving ensemble generation, for tackling challenging manipulated video deepfake identification.

Table 2 Comparison of different search strategies in existing studies

	Algorithm	Multiple leaders	Multiple search actions	Any new search operations	Switching between different search actions	Adaptive/new search coefficients	Integration with other search methods	Leader improvement
Zhang et al. [35]	PSO variant	Subswarm leaders and the global best solution used for velocity update	-	Subswarm formed using KM and a CH index used to identify the number of clusters	-	Adaptive nonlinear coefficients	Bayesian optimization to identify search boundaries of hyper-parameters	-
Zhang et al. [36]	PSO variant	-	-	PSO operation with an additional mutation operator	-	Adaptive inertia weight coefficient	-	-
Lorenzo et al. [48]	PSO	-	-	No (PSO operation)	-	-	-	-
Lorenzo et al. [49]	Parallel PSO	-	-	No (PSO operation, but with simultaneous fitness evaluation for multiple particles)	-	-	-	-
Junior and Yen [50]	PSO variant	-	-	New velocity and position difference calculation	-	-	-	-
Lawrence et al. [51]	PSO variant	-	-	New velocity and position difference calculation dedicated to residual networks	-	Linear adaptive search steps	-	-
Baker et al. [52]	RL (Q-learning)	-	-	-	-	-	-	-
Zoph et al. [53]	RL (PPO)	-	-	-	-	-	-	-

Jordehi [64]	PSO variant	-	-	No (PSO operation)	-	-	-	Random walk, DE and opposition based leader improvement
Tan et al. [45]	PSO variant	Subswarm leaders	-	No (PSO operation led by different search coefficients in subswarms)	-	Adaptive search parameters implemented using sine-cosine functions	-	Swarm leader enhancement using PSO operation in partial dimensions
Tan et al. [45]	PSO variant	Subswarm leaders	-	No (PSO operation but with several random offspring solutions generated in each subswarm)	-	Adaptive nonlinear sine-cosine coefficients	-	Swarm leader enhancement using PSO operation in partial dimensions
Tan et al. [57]	PSO variant	Swarm leader and neighbouring signals	Yes	PSO, FA and local search actions	Random switching	-	-	GA-based leader enhancement
Zhang et al. [58]	FA variant	Combing three best leaders using adaptive weighting factors	Yes	Search operations led by hybrid leaders	Random switching	Adaptive nonlinear sine-cosine coefficients	-	-
Fallahi et al. [59]	PSO or DE + RL (Q-learning)	-	-	No (PSO/DE actions)	-	Search coefficients selected using RL	Q-learning for search parameter generation	-
Zhang et al. [60]	BBPSO + RL (Q-learning)	Neighbouring and global best signals	Yes	BBPSO + local and global search actions with different averaged leader signals	Random switching	Coefficients produced using random distributions	Q-learning used for leader enhancement by selecting one of the root-finding algorithms	Root-finding algorithms and RL for leader enhancement
Liang et al. [37]	PSO variant	Two subswarm leaders and the swarm leader	Yes	A DE-based local search and subswarm leader based global search	Random switching	Adaptive inertia weight coefficient	-	-
Liu et al. [38]	PSO variant	All personal best experiences	-	A modified PSO velocity operation using all personal best experiences as leaders for position update of each particle	-	A coefficient generated based on the fitness difference between the current particle and its personal best solution	-	Randomly selected personal best individuals for information exchange
Liu et al. [39]	PSO variant	Randomly generated and immune orientation based leaders	Yes	A new nervous guidance scheme for velocity update	Random switching	Adaptive coefficients yielded by an endocrine regulation mechanism	-	-
Lu et al. [40]	PSO variant	An activation leader combining the swarm leader and stagnant subswarm leader using random weighting factors	Yes	A re-initialization method used to re-dispatch particles to unexplored regions	Threshold-based (distance-based) activation	-	-	-
Li et al. [41]	PSO variant	Neighbouring elite and global best	-	No (PSO operation with different leader	Threshold-based (distance-	Four sets of search coefficients	-	DE-based personal and global best

		signals		signals)	based) switching	allocated based on search stages		solution enhancement
Chen et al. [42]	PSO variant	Combined leader generation using personal best solutions	Yes	Movement update led by the combined leader signal	Stagnation checking using a stagnation counter	-	-	-
This research	PSO + RL (Q-learning)	Generating four hybrid leaders by combining the local and global elite signals using adaptive weighting coefficients yielded by 3D formulae	Yes	New local and global search operations led by hybrid leaders	Reinforcement learning based optimal search action selection	Adaptive nonlinear coefficients generated using 3D super formulae and a helix function.	RL (Q-learning) is used to select local and global search actions for each particle.	Root-finding algorithms (Muller's Method and fixed-point Iteration) employed for swarm leader enhancement

2.3 Ensemble Model Construction

A number of studies have explored ensemble model development. As an example, Zhang et al. [66] studied ensemble model development with imbalanced class distribution using genetic programming (GP). Multi-objective processes with the focus on false positive and negative rates and ensemble size were designed for selecting optimal subsets of the base classifiers while improving ensemble classification performance. After obtaining the Pareto front solutions from the evolutionary process, a weighted aggregation scheme was proposed to generate the final prediction using the results from each base classifier. A set of 40 UCI datasets was used to assess the effectiveness of the GP-based dynamic ensemble generation process. Their work exploited both GP-based base model selection and a weighted scheme for joint decision making. In view of the adoption of a multi-objective optimization process, their ensemble building process was computationally expensive in particular for multi-class classification problems. Their evaluation mainly focused on UCI datasets with traditional machine learning methods as the base classifiers, without involving large-scale video datasets with deep networks as the base learners. Since the base model diversity could have a significant influence toward the ensemble performance, their GP process could be also used to perform feature selection to generate more diversified base classifiers.

Fan et al. [67] developed a multi-tree GP for ensemble model construction. Their encoding process used one GP to represent one ensemble classifier. A multi-objective fitness evaluation was employed to increase ensemble accuracy while minimizing ensemble cost. The tree structures within the GP individuals were designed for image classification tasks. Specifically, the base classifiers within an individual GP consisted of feature descriptors such as LBP and SIFT, combined with machine learning algorithms such as SVM, Linear Regression (LR) and Random forest (RF). The NSGAII optimizer was used to guide optimal base classifier search with an ensemble by performing random crossover/mutation operations. After the search process, the GP solutions in the Pareto front were used to build the final ensemble classifier, where the outputs of the selected base classifiers were fused using a traditional majority voting method. Their model was evaluated using several image classification datasets. While the adoption of a GP embedding multiple trees (base classifiers) for ensemble model generation offered an advantage, the base model diversity method could be constrained by such tree structure representations. Moreover, their work mainly used hand-crafted features extracted by feature descriptors combined with traditional machine learning algorithms (e.g. SVM and RF), which could be replaced by deep CNNs, owing to the efficiency of machine learned features using deep networks for spatial pattern extraction. Bosowski et al. [68] studied dynamic ensemble development using GA for disease diagnosis from chest X-ray images. Deep CNN models (e.g. ResNet and VGG) pre-trained on ImageNet were employed as the base networks. Each search agent in the swarm represented the construction of one ensemble model. Besides the adoption of traditional ensemble strategies such as majority voting, supervised schemes were applied to fuse the selected base classifiers. Specifically, three additional classifiers (e.g. SVM, a feedforward neural network and a gradient boosting classifier) were trained as supervised meta-learners to combine the results from the selected base networks. The effectiveness of the GA-based ensemble construction was evaluated using 9 datasets. The advantage of their studies included the adoption of different types of base networks, increasing ensemble diversity, as well as the integration of three meta-learners. However, their model relied on the original GA algorithm for ensemble network development without integrating new search strategies to overcome premature convergence of the GA.

Nalepa et al. [69] exploited ensemble model construction with a supervised fusion scheme. Different base networks including different types of CNNs, such as 1D, 2.5D and 3D CNNs, were employed as the base learners for image classification problems, while 1D and 3D CNNs were adopted for image unmixing tasks. In addition, new 1D CNN base models were yielded by embedding Gaussian noise on network weights of the original base 1D CNN methods. The original base networks and any augmented networks (if applicable) were employed to construct the base classifier pool. A supervised ensemble scheme was subsequently used to combine the results of all the networks in the base classifier pool to generate the final ensemble prediction. Specifically, for each sample, the prediction probabilities of each base network for all the classes were concatenated and used as the inputs of the supervised ensemble learner. Three machine learning methods, i.e. RF, SVM and decision tree (DT), were applied as the supervised ensemble learners. The ensemble models showed an impressive performance in a number of image classification and unmixing tasks. While the formulated supervised ensemble learners and Gaussian noise-augmented base classifiers were useful, their ensemble strategy took outputs of all the base networks, regardless of their performance, into account for ensemble result generation via a supervised learner. In this regard, evolutionary algorithm-based ensemble construction methods could be exploited, in order to eliminate weak base classifiers and reduce cost. A weighted ensemble scheme could also be incorporated to enhance the impact of the original base classifiers in comparison with those from the augmented base learners.

Pratama et al. [70] constructed a dynamic ensemble fuzzy classifier for data stream classification. Concept drift detection, real-time feature selection, and ensemble pruning methods were incorporated in their model. A new base model was inserted if the drift of concepts was detected from the input stream. Two ensemble pruning schemes were developed by measuring the relevance and generalization impact of respective base learners. Moreover, the online feature selection method enabled the elimination or selection of a specific attribute by assigning binary crisp values. Evolving classifiers at the base and ensemble levels were devised. Evaluated using 15 datasets, their model achieved a superior performance. Their model benefited greatly from the integration of the aforementioned drift detection and base classifier pruning based on dynamic determination of the relevance and effectiveness of respective learners. But their online feature selection process relied on the importance measure of individual features without the consideration of feature interaction, where the significance of a feature could be affected by the presence of other features, as evidenced in evolutionary algorithm-based feature selection strategies [45, 57].

Ngo et al. [71] developed an evolutionary ensemble bagging scheme. A bootstrapping method with replacement was used to generate the initial bags (i.e. subsets of training data) for individual classifiers. Instead of using fixed samples as in the conventional bagging ensemble method, an evolutionary process including crossover and mutation operations was used to modify/improve training samples in the bags, based on the performance of the DT base classifiers. A ranking strategy was employed to select fitter individuals for offspring generation. To increase data diversity, in each iteration, a number of new bags were randomly generated using bootstrapping. Such a process could be used to replace weaker base learners through iterations. The work was useful for generating evolving samples using genetic operators to boost ensemble diversity. Nonetheless, their study focused on the evaluation using UCI or similar datasets. The scalability of their method could be further ascertained using comparatively more complex datasets.

Zhang et al. [72] developed an FA variant for dynamic ensemble model construction. The FA variant was equipped with local and global best and worst experiences to lead the attractiveness and fleeing operations, respectively. Their FA model performed optimal base classifier selection for dynamic ensemble model development, where each firefly was used to represent a candidate ensemble model. To further increase base model diversity, PSO was used to perform feature selection in order to generate corresponding new base learners. Their optimization process balanced between performance and ensemble complexity. The model outperformed 3 classical and 5 FA variants for solving a variety of high-dimensional classification datasets as well as complex benchmark functions. Their work exploited the strength of PSO-based feature selection for diversified base learner generation as well as the benefit a modified FA-based ensemble model construction for minimizing ensemble complexity while maximizing classification accuracy. However, the selected base classifiers within an optimized ensemble model were fused using a traditional majority voting scheme, instead of a weighted integration strategy to give more influence to those well-behaved base classifiers.

Besides the above, there are also a number of other studies using PSO and PSO variant methods for ensemble model construction, such as Cai et al. [73], Malhotra and Khanna [74], Hong et al. [75], Shafqat et al. [76] and Tan et al. [77]. A survey study for evolutionary algorithm-based ensemble construction was also performed by Cagnini et al. [78].

3. THE PROPOSED SWARM INTELLIGENCE ALGORITHM

We propose a variant of the PSO algorithm for network topology and hyper-parameter identification in CNN-RNN, I3D and MC3, as well as weighted and evolving ensemble construction for fake/real video classification. It incorporates the fixed-point iteration and Muller's method based leader enhancement, cross-breed leader generation using 3D adaptive parametric surfaces, a petal helix search mechanism and reinforcement learning-inspired sequential search operation deployment for solving a variety of optimization problems.

We first initialize a swarm with random particle positions. Configurations of each particle are used to establish the respective network with optimized settings for fitness evaluation. The particle with the most optimal solution is identified as g_{best}, which is further improved using numerical analysis algorithms, i.e., Muller's method and fixed-point iteration. Subsequently, crossover operators are exploited to integrate personal and global best solutions for elite cross-breed leader generation. Specifically, a set of four 3D geometric surfaces is used to generate adaptive weighting coefficients. They are used to fine-tune the effects of the personal and global best individuals for the cross-breed leader generation, in order to balance well between diversification and intensification. For each particle, such a customized combined leader is generated using a randomly selected set of crossover formulae among the four operators. A petal helix search intensification is also utilized to exploit optimal local regions. Moreover, a reinforcement learning algorithm, i.e. the Q-learning algorithm, is used to identify the optimal sequential deployment of these local and global search mechanisms to increase robustness. We elaborate each proposed scheme in detail, as follows.

3.1 Local Exploitation Using Muller's Method

Two numerical analysis methods, i.e., Muller's method and fixed-point iteration, are exploited for leader enhancement. The former adopts three guesses while the latter uses one to estimate a new root. Such numerical analysis methods take advantage of recursive root finding mechanisms for global minima estimation. Therefore, they are more capable of finding global optimality in comparison with other un-directed random search strategies for leader enhancement. Equations (3)-(4) illustrate the formula for the Muller's method [79].

$$x_j = x_{j-1} - \frac{2f(x_{j-1})}{b \pm \sqrt{b^2 - 4f(x_{j-1}) \times df[x_{j-1}, x_{j-2}, x_{j-3}]}} \tag{3}$$

$$b = df[x_{j-1}, x_{j-2}] + df[x_{j-1}, x_{j-3}] - df[x_{j-2}, x_{j-3}] \tag{4}$$

where x_{j-1}, x_{j-2}, and x_{j-3} indicate the three initial estimations of the root, and $f(x_{j-1})$, $f(x_{j-2})$, and $f(x_{j-3})$ denote their respective function values. In addition, x_j is the approximation of the root for the current iteration. The sign '$\pm$' in Equation (3) is selected based on the attempt to maximize the denominator to make it as large as possible in magnitude. Moreover, $df[x_{j-1}, x_{j-2}]$, $df[x_{j-1}, x_{j-3}]$, $df[x_{j-2}, x_{j-3}]$, and $df[x_{j-1}, x_{j-2}, x_{j-3}]$ represent the divided differences, which is a mathematical algorithm with a recursive division process. The divided differences algorithm is used to compute the coefficients of the interpolation polynomial of the given input data samples [79].

In this research, we employ the top three global best solutions, i.e., g_{best}, the second and the third bests, as the three initial approximations of the root, i.e. x_{j-1}, x_{j-2}, and x_{j-3}. $f(x_{j-1})$, $f(x_{j-2})$, and $f(x_{j-3})$ signify their respective fitness values (i.e., error rates) with respect to synthetic video classification. This process of Muller's method is used to estimate a new root. It terminates when a significantly small error rate is obtained. If the new estimated root is fitter than g_{best}, then it is used to update g_{best}, otherwise the current g_{best} is kept for subsequent processing. Such a steered guided leader enhancement can accelerate convergence with a better chance of finding global optima.

3.2 Local Exploitation Using Fixed-point Iteration

To diversify the search process, another numerical analysis method, i.e., fixed-point iteration, is used to improve the swarm leader. Equation (5) defines the detailed formula [79]. In comparison with Muller's method, it only requires one initial guess instead of three to calculate fixed points of a given function.

$$x_{k+1} = f(x_k) \tag{5}$$

where $k = 0, 1, 2, \ldots maximum_{iteration}$, and x_k denotes the initial guess. The process iterates until it reaches the maximum number of trials. In this research, we employ g_{best} as the initial estimation, while $f(x_k)$ denotes the fitness score of g_{best}, i.e., the error rate of video forgery classification. Because of the adoption of root

estimation strategies, similar to Muller's method, this fixed-point iteration method is more capable of attaining global optima as compared with chaotic movements, random mutations and Levy/Cauchy distributions.

These two numerical analysis methods, i.e. Muller's method and fixed-point iteration, are randomly selected to exploit leader enhancement.

3.3 The Cross-breed Leader-motivated Global Exploration

To diversify the search process, a new velocity generation formula is proposed, as shown in Equations (6)-(7). Instead of using the historical optimal solutions in separate social and cognitive components, Equation (6) adopts a cross-breed compound leader C_i to guide the search process. Equation (7) further defines the generation of this hybrid leader C_i with adaptive parameters α and β implemented using each of the four 3D geometric surfaces.

Specifically, four 3D geometric shapes are exploited for compound elite leader generation, which implement fine-grained, moderate and drastic non-linear varying increasing α and decreasing β trajectories to adjust the effects of the global and personal best signals for compound leader generation, as indicated in Equation (7). Such cross-breed signals serve as adaptive elite leaders to widen search territories and exploit optimal regions effectively.

To be precise, these four 3D geometric shapes implement distinctive border patterns for production of incremental and decremental adaptive coefficients, α and β, to empower diversified compound leader generation. In particular, as indicted in Equation (7), the decreasing β and increasing α trajectories are signified to emphasize the leadership of the personal best solutions at an initial level and the global elite signal towards the final search stage.

$$v_{id}^{t+1} = w \times v_{id}^{t} + \alpha \times rand \times (C_{id} - x_{id}^{t}) \quad (6)$$

$$C_{id} = \alpha g_{best\_d} + \beta p_{best\_id} \quad (7)$$

where α and β represent the increasing and decreasing adaptive crossover factors implemented using 3D geometric surfaces. In addition, C_{id} denotes the yielded cross-breed leader for particle i in a specific dimension. Owing to the variations of p_{best_i} for each particle, a distinctive bespoke C_i is generated for each individual. Besides that, α is also used as the search parameter in Equation (6) to fine-tune the effects of diversification and intensification. Precisely, the increasing coefficient α applies smaller forces while moving towards the cross-breed leader C_{id} to encourage global exploration in early iterations and adapts to larger strength of moving towards C_{id} to boost local exploitation in final search steps. Moreover, a random coefficient, $rand$, is used to multiply with α, in Equation (6) to diversify search steps. The yielded new velocity in Equation (6) is then used for position updating using Equation (1).

We subsequently elaborate the generation of the cross-breed leader, C_{id}, as well as the crossover factors, α and β, using four 3D geometric shapes in Subsections 3.3.1-3.3.4.

3.3.1 The Generation of the First Cross-breed Leader

To mitigate the likelihood of being trapped in local optima, four sets of adaptive functions are exploited for cross-breed leader generation. To diversify the search process, a bespoke leader signal is generated for each particle by taking varied proportions of the global and personal best signals in accordance with the proposed adaptive 3D geometrical surfaces. Equations (8)-(10) define the first set of formulae for the cross-breed leader generation.

$$x = 1.5 \times \left((|cos(\tfrac{5\varphi}{4})| + |sin(\tfrac{5\varphi}{4})|)^{-1} \right) \times ((|cos(\tfrac{5\sigma}{4})| + |sin(\tfrac{5\sigma}{4})|)^{-1}) \times cos(\varphi) \times cos(\sigma) \quad (8)$$

$$y = 1.5 \times \left((|cos(\tfrac{5\varphi}{4})| + |sin(\tfrac{5\varphi}{4})|)^{-1} \right) \times ((|cos(\tfrac{5\sigma}{4})| + |sin(\tfrac{5\sigma}{4})|)^{-1}) \times \sin(\varphi) \times cos(\sigma) \quad (9)$$

$$z = 1.5 \times ((|cos(\tfrac{5\sigma}{4})| + |sin(\tfrac{5\sigma}{4})|)^{-1}) \times \sin(\sigma) \quad (10)$$

where $\varphi = [-\pi: 0.05: \pi]$ and $\sigma = [-\pi/2: 0.05: \pi/2]$. The above formulae generate a complex 3D geometrical shape as illustrated in Figure 2, with smooth border patterns in the z-axis. The value range of the z-axis is [-1.3883, 1.3883]. First of all, we extract 7,938 numbers of unique absolute values from the z-axis of the yielded geometrical curve. These values are then ranked in increasing and decreasing orders, respectively. Each ranked

sequence is divided into T proportions, where T denotes the termination iteration number. As such, we employ a step of $7{,}938/T$, from both ranked increasing and decreasing sequences, to assign cross-breed factors, i.e., α and β, respectively, for compound leader generation.

As indicated in Equation (7), such adaptive increasing and decreasing coefficients, i.e. α and β, are able to assign smaller emphasis of g_{best} and larger impact of p_{best} in early iterations, while as the iteration increases, the search process adapts to a larger influence of g_{best} and a smaller effect of p_{best}. Owing to the variations of p_{best} for each particle in each iteration, a unique cross-breed leader is yielded for each individual. This tailored elite leader guides each search agent to explore a distinctive search region, independently.

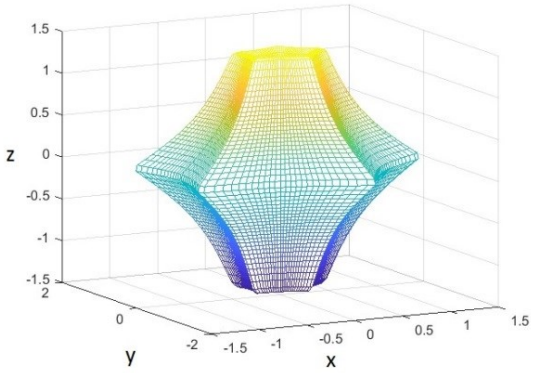

Figure 2 The respective 3D geometrical shape generated using Equations (8)-(10)

3.3.2 The Generation of the Second Cross-breed Leader

Besides the above geometrical formulae, hybrid leader generation is inspired by irregular adaptive artificial landscapes. The first irregular 3D surface is defined by Equations (11)-(13). Figure 3 shows the respective yielded 3D landscape. In comparison with the aforementioned surface generated using Equations (8)-(10) with regular smooth border outlines as shown in Figure 2, this new 3D layout contains irregular drastic border trajectory patterns to diversify cross-breed leader generation.

$$x = 2 \times \left((|cos(5\varphi)|^5 + |sin(5\varphi)|^{25})^{-\frac{1}{10}} \right) \times ((|cos(5\sigma)|^5 + |sin(5\sigma)|^{25})^{-\frac{1}{10}}) \times cos(\varphi) \times cos(\sigma) \tag{11}$$

$$y = 2 \times \left((|cos(5\varphi)|^5 + |sin(5\varphi)|^{25})^{-\frac{1}{10}} \right) \times ((|cos(5\sigma)|^5 + |sin(5\sigma)|^{25})^{-\frac{1}{10}}) \times sin(\varphi) \times cos(\sigma) \tag{12}$$

$$z = 2 \times ((|cos(5\sigma)|^5 + |sin(5\sigma)|^{25})^{-\frac{1}{10}}) \times sin(\sigma) \tag{13}$$

where $\varphi = [-\pi : 0.05 : \pi]$ and $\sigma = [-\pi/2 : 0.05 : \pi/2]$.

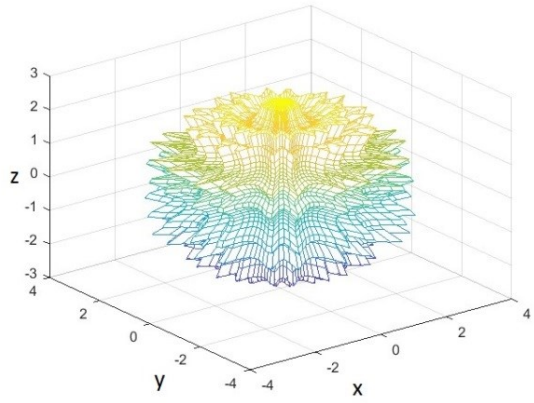

Figure 3 The respective 3D geometrical shape generated using Equations (11)-(13) with needle-shaped borders

This elliptical shape is constructed using 7,938 numbers of 3D points with a value range of [-3.3271, 3.3271] for the z-axis. The irregular trajectories of the z-dimension are used to produce the adaptive breeding factors. As shown in Figure 3, such trajectories from the z-axis have needle-shaped border styles, in comparison with smooth border trails yielded using Equations (8)-(10). As such, the resulting surface illustrates different border

patterns in the z-axis with more drastic variations between adjacent points. A total of 7,938 unique absolute values from the z-axis are also collected, which are subsequently ranked in the increasing and decreasing orders for breeding factor generation. Next, each adaptive border trajectory is divided into T portions. The two sets of adaptive coefficients are assigned as the respective breeding factors, α and β, for g_{best} and p_{best} respectively, in accordance with the iteration numbers with a step of $7{,}938/T$. As such, they enforce stronger influence of p_{best} along with subtle effects of g_{best} in the early iterations, and adapt to increasing emphasis of g_{best} with a minute impact of p_{best} as the search progresses. In comparison with adaptive factors generated using Equations (8)-(10), these elliptical curves with irregular increasing and decreasing borders are able to yield distinctive weighting factors to diversify compound leader generation.

3.3.3 The Generation of the Third Cross-breed Leader

To maintain robust and resilient search behaviours, another geometrical shape is employed for adaptive cross-breed factor generation. Equations (14)-(16) define the respective geometrical contour with Figure 4 depicting the respective visualised 3D shape.

$$x = 2 \times \left((|cos(7.5\varphi)|^{-10} + |sin(7.5\varphi)|^{65})^{-\frac{1}{88}} \right) \times ((|cos(7.5\sigma)|^{-10} + |sin(7.5\sigma)|^{65})^{-\frac{1}{88}}) \times cos(\varphi) \times cos(\sigma) \quad (14)$$

$$y = 2 \times \left((|cos(7.5\varphi)|^{-10} + |sin(7.5\varphi)|^{65})^{-\frac{1}{88}} \right) \times ((|cos(7.5\sigma)|^{-10} + |sin(7.5\sigma)|^{65})^{-\frac{1}{88}}) \times sin(\varphi) \times cos(\sigma) \quad (15)$$

$$z = 2 \times ((|cos(7.5\sigma)|^{-10} + |sin(7.5\sigma)|^{65})^{-\frac{1}{88}}) \times sin(\sigma) \quad (16)$$

where $\varphi = [-\pi : 0.05 : \pi]$ and $\sigma = [-\pi/2 : 0.05 : \pi/2]$.

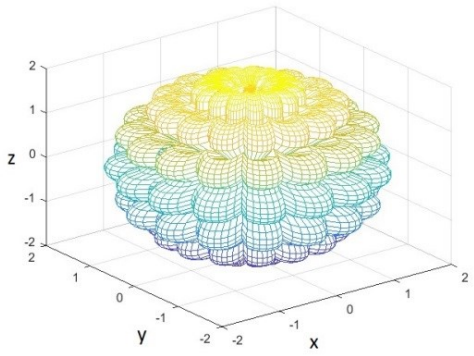

Figure 4 The respective 3D geometrical shape generated using Equations (14)-(16) with wave-like borders

Again, the unique absolute values of z-axis in the range of [-1.9873, 1.9873] are extracted and then ranked in both increasing and decreasing sequences. Then both ranking sequences are split into T proportions, which are utilized to assign the respective decremental and incremental values as the breeding factors for the p_{best} and g_{best} solutions, respectively, based on the iteration number. Since the resulting geometrical curve illustrates different wave-like border patterns in the z-axis with comparatively less drastic but moderate variations between adjacent points, it generates distinguishing weighting factors for cross-breed signal generation, as compared with those yielded by other aforementioned strategies. The resulting bespoke leader for each particle is thus able to explore wider search regions to better tackle local optimal traps.

3.3.4 The Generation of the Fourth Cross-breed Leader

The fourth set of formulae used for breeding leader generation is provided in Equations (17)-(19) where the z-axis has the value range of [-1.1206, 1.1206]. In comparison with those defined earlier, the visualized respective surface derived from Equations (17)-(19) shown in Figure 5 possesses distinctive tile-like border outlines with comparatively more mild and subtle variations between neighbouring points. A total of 7,938 unique absolute values are extracted from the z-axis and subsequently ranked. The resultant increasing and decreasing sequences are cropped into T portions, respectively, with a step of $7{,}938/T$ and employed to assign the breeding factors, α and β, with respect to g_{best} and p_{best} respectively, for compound leader generation.

$$x = \left((|cos(-5\varphi)|^{65} + |sin(-5\varphi)|^{10})^{-\frac{1}{88}} \right) \times ((|cos(-5\sigma)|^{65} + |sin(-5\sigma)|^{10})^{-\frac{1}{88}}) \times cos(\varphi) \times cos(\sigma) \quad (17)$$

$$y = \left((|cos(-5\varphi)|^{65} + |sin(-5\varphi)|^{10})^{-\frac{1}{88}} \right) \times ((|cos(-5\sigma)|^{65} + |sin(-5\sigma)|^{10})^{-\frac{1}{88}}) \times sin(\varphi) \times cos(\sigma) \quad (18)$$

$$z = ((|cos(-5\sigma)|^{65} + |sin(-5\sigma)|^{10})^{-\frac{1}{88}}) \times sin(\sigma) \quad (19)$$

where $\varphi = [-\pi: 0.05: \pi]$ and $\sigma = [-\pi/2: 0.05: \pi/2]$.

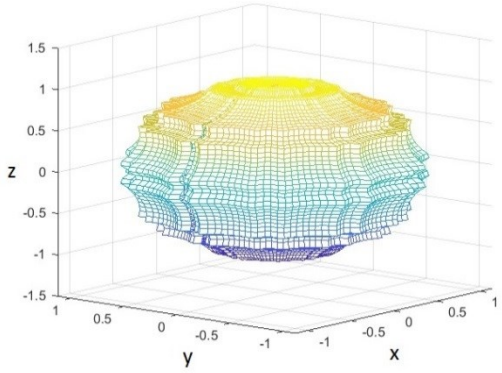

Figure 5 The respective 3D geometrical shape generated using Equations (17)-(19) with tile-like borders

The above four sets of adaptive crossover factor assigning schemes use subtle, moderate and drastic non-linear varying increasing and decreasing trajectories to generate diverse compound elite signals to overcome stagnation. Overall, these varying cross-breeding factors are able to obtain a better trade-off of diversification and intensification. They accentuate global diversification by allocating larger weights to p_{best} in early iterations and highlight local exploitation by strengthening the influence of g_{best} in subsequent iterations.

3.4 Local Exploitation Using Petal Helix Search

Besides the global exploration using Equations (6)-(7), a simulated petal helix local exploitation action is proposed as indicated in Equation (20)-(21). It embeds a new helix parameter δ to fine-tune search steps with the attempt to intensify the search around the cross-breed leader.

$$x_{id}^{t+1} = C_{id} + \delta \times (C_{id} - x_{id}^{t}) \quad (20)$$

$$\delta = 1.5\, cos(4\theta) \quad (21)$$

where δ denotes the proposed petal helix coefficient with $0 \leq \theta \leq 2\pi$. The value range of δ is [-1.5, 1.5].

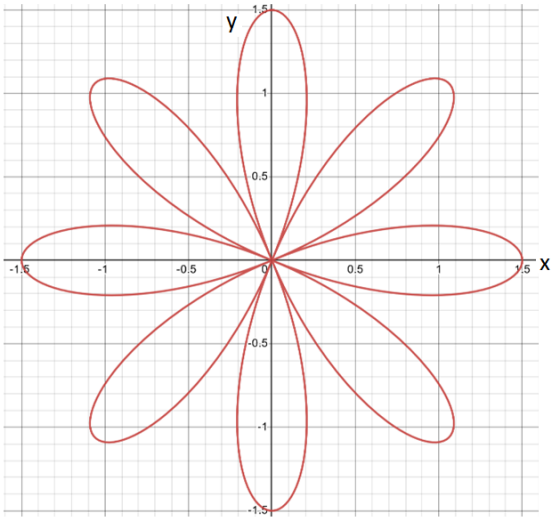

Figure 6 The 2D shape implemented using Equation (21)

The petal trajectories implemented by Equation (21) are shown in Figure 6. This yielded a flower shape with distinctive spiral petals is able to empower each particle to exploit optimal regions by traversing through irregular nonlinear paths. For each particle, we randomly select a θ value between 0 and 2π to produce the helix

coefficient δ, which is then used in Equation (20) to calculate the respective new position of each particle. Since both positive and negative values can be produced for δ using Equation (21), the particles are able to move towards or away from the cross-breed signal and thoroughly examine the optimal regions.

3.5 Reinforcement Learning-inspired Search Action Selection

The optimal deployment of the aforementioned global and local search schemes defined in Equations (6)-(7) and Equations (20)-(21) respectively is important in attaining global optimality. A reinforcement learning algorithm, i.e. the Q-learning algorithm [80], is thus used to identify the optimal sequential deployment of these local and global search operations for each particle. The working principle of reinforcement learning is to reward desired actions and punish undesired ones. A reinforcement learning agent takes an action from a set of actions to transit from one state to another. It perceives the environments via the reward signals based on trial-and-error. The Q-learning algorithm employs the Bellman equation as shown in Equation (22) to select a sequence of optimal actions that maximize the cumulative reward. Such a cumulative reward score with respect to each action-state pair is kept in a Q-table, which is used to guide the selection of an optimal action from a specific state.

$$Q(s_t, a_t) = (1 - \gamma) \times Q(s_t, a_t) + \gamma \times (r_t + \tau \times \max_a Q(s_{t+1}, a)) \quad (22)$$

where s_t and a_t denote a state and an action respectively. The Q-value, i.e. $Q(s_t, a_t)$, produced using Equation (22) is used to update the Q-table. An immediate reward r_t is calculated by performing the action a_t in the state s_t. A discount coefficient τ is used to fine-tune the effect of the future reward $\max_a Q(s_{t+1}, a)$, i.e. the reward to be obtained from the new state s_{t+1}. The learning rate γ is used to adjust the effect of the new reward to the Q-value update. The immediate reward r_t has a score of '1' if the new fitness is improved by implementing the action a_t in the state s_t in comparison with the previous fitness, otherwise '-1'.

The above Q-learning process is used to identify the optimal sequential deployment of the global and local search operations defined in Equations (6)-(7) and Equations (20)-(21) respectively. Each particle maintains a 2x2 Q-table in order to ensure the optimal deployment of the above local and global search actions. Therefore these local and global search operations are selected in a way to optimize search behaviours of each search agent. By assigning sequential customized search actions to each individual via solving Bellman optimality, the Q-learning-based strategy leads to a better balance between diversification and exploitation.

Overall, the proposed adaptive cross-breed operators oriented from 3D geometric surfaces, a helix-driven search mechanism, reinforcement learning-based search scheme deployment, and numerical analysis-inspired leader enhancement, work co-ordinately to overcome stagnation and increase search flexibility. The interactions of the aforementioned search strategies are depicted in Algorithm 1.

Algorithm 1: Data flow of the Proposed PSO Model	
1.	**Start**
2.	Initialize a swarm with n particles randomly;
3.	Calculate the fitness score of each particle;
4.	Select the swarm leader g_{best} based on the fitness scores;
5.	**While** (!Stagnation) {
6.	Perform g_{best} enhancement using any of the following operations;
7.	1. Improve g_{best} using the Muller's method (with three guesses) as defined in Equations (3)-(4);
8.	2. Improve g_{best} using the fixed-point iteration method (with one guess) as indicated in Equation (5);
9.	**For** (particle i in the swarm) do {
10.	Choose any of the following mechanisms for cross-breed leader generation;
11.	1. Generate cross-breed leader 1 using adaptive 3D coefficients yielded using Equations (8)-(10);
12.	2. Generate cross-breed leader 2 using adaptive 3D coefficients yielded using Equations (11)-(13);
13.	3. Generate cross-breed leader 3 using adaptive 3D coefficients yielded using Equations (14)-(16);
14.	4. Generate cross-breed leader 4 using adaptive 3D coefficients yielded using Equations (17)-(19);
15.	Use the Q-learning algorithm to select the following local and global search actions;
16.	1. Perform global search using the selected cross-breed leader and the respective adaptive coefficient α as defined in Equations (6)-(7) and (1);
17.	2. Perform local search using the selected cross-breed leader and the helix coefficient δ as defined in Equations (20)-(21);

18.	Update the p_{best} if the newly derived individual is fitter;
19.	} **End For**
20.	Update g_{best};
21.	} **Until** (Stagnation)
22.	Output g_{best};
23.	**End**

We subsequently introduce optimal network topology and hyper-parameter identification in CNN-RNN and 3D CNNs, respectively, as well as weighted and evolving ensemble generation, using the proposed PSO algorithm for video forgery classification.

4. GENERATION OF EVOLVING 3D CNN AND CNN-RNN MODELS

In this research, we conduct deepfake detection using CNN-RNNs and 3D CNNs with PSO-based optimal learning for configuration identification, along with weighted and evolving ensemble formulation. Specifically, for the hybrid CNN-RNN model, we employ Inceptionv3 as the CNN encoder and different types of RNNs as the decoder. In particular, trial-and-error is also conducted using other encoder networks such as ResNet50, ResNet101, VGG19 and VGG16. Inceptionv3 is selected owing to its significant superiority in spatial feature learning and a better trade-off between performance and cost. Specifically, to further enhance the spatial-temporal feature extraction capabilities, the new PSO variant discussed in Section 3 is used to optimize the type (i.e., BiLSTM, LSTM and GRU) and the number of hidden neurons of the RNN decoder for the identification of manipulated videos. In particular, these different RNN models employ different gating structures and unidirectional and bidirectional learning mechanisms. Precisely, LSTM and BiLSTM extract temporal patterns by using unidirectional and bidirectional sequences, respectively. In addition, an LSTM unit contains input, output, and forget gates, while GRU has a simpler structure with only reset and update gates embedding fewer parameters. Therefore, BiLSTM, LSTM and GRU possess significantly different temporal feature learning mechanisms as well as gating topologies to diversify memory management and increase flexibility. In addition, the internal structure of the yielded RNN layer also determines network capabilities in extracting sequential details. For instance, large and small numbers of hidden neurons can lead to the extraction of excessive or insufficient temporal patterns. Optimization of the network type and the number of hidden units of the RNN decoder is thus able to yield diverse learners with different learning strategies. The detailed Inceptionv3-RNN architecture used in this research is shown in Figure 7.

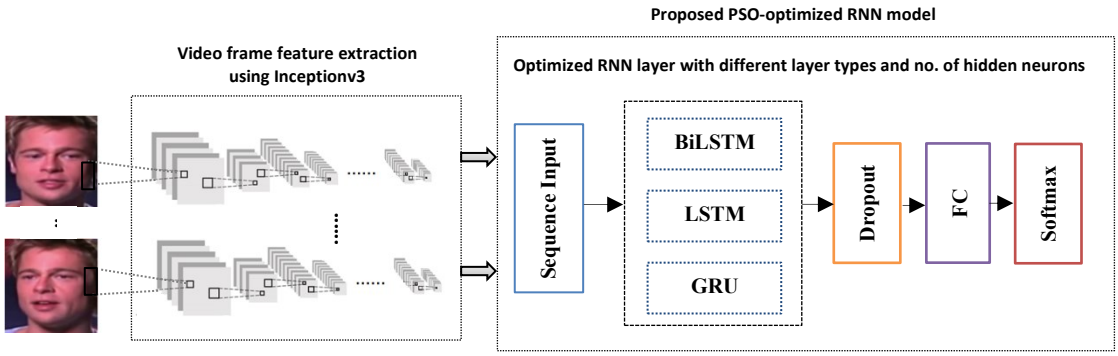

Figure 7 The network architecture of Inceptionv3-RNN with the proposed PSO-based hyper-parameter optimization (where the proposed algorithm is used to optimize the layer types, i.e. BiLSTM, LSTM and GRU, and number of hidden units of the RNN model)

Moreover, I3D [10-12] and MC3 [14] are employed as the 3D CNNs for manipulated video identification, owing to their impressive performance in comparison with those of 3D ResNeXt and ResNet models [14]. These two networks are provided by Python built-in libraries. In particular, the MC3 network is a variant of the 3D ResNet model where the latter contains 5 groups of 3D convolutions. Instead of using all 3D convolutions, MC3 embeds mixed convolutions, where 3D convolutions are substituted with 2D ones in the last 3 groups.

Both I3D and MC3 models are pre-trained using RGB frames of a large video action dataset, i.e. Kinetics, for the classification of 400 human actions. We are taking advantage of such pre-trained models with superior spatial-temporal feature learning capabilities by assigning the respective networks with their pre-trained weights on the human action dataset. Transfer learning is subsequently conducted to further fine-tune the 3D CNNs for tampered video classification. We optimize network learning configurations of both 3D CNNs, i.e., the initial

learning rate, learning rate decay/drop factor, and the regularization coefficient, to adapt them to different synthetic video classification tasks effectively. Different settings of these learning configurations in I3D and MC3 regulate significant distinctive learning behaviours and network capabilities in tackling under-fitting and over-fitting episodes.

Table 3 Key elements to be optimized

	Key parameters	Search ranges
Inceptionv3-RNN	Type of RNN layers	GRU, LSTM, BiLSTM
	Number of hidden units	[500, 1800]
I3D	Learning rate	[0.001, 0.01]
	Learning rate drop factor	[0.01, 0.1]
	Regularization coefficient	[0.0001, 0.003]
MC3	Learning rate	[0.001, 0.01]
	Learning rate drop factor	[0.01, 0.1]
	Regularization coefficient	[0.0001, 0.003]

In particular, the learning rate is the most important hyper-parameter of the optimizer. The initial learning rate and the learning rate drop factor work cooperatively to adjust network learning paces. The learning rate drop factor is a pre-defined constant value, which defines the proportion to decrease the current learning rate over a certain number of training epochs. For example, a large initial learning rate in combination with a small learning rate decay factor is more likely to alter the optimizer more drastically to overlook global optima. On the contrary, a small initial learning rate fine-tuned with a large learning rate decay schedule is inclined to perform an insufficient small adjustment to the optimizer to result in under-fitting. Furthermore, the regularization factor (i.e., the weight decay) adjusts the effects of the regularization term in the loss function with the attempt to fine-tune network capabilities in tackling over-fitting. A network with a very small weight decay coefficient can learn the training dataset exclusively tightly to result in over-fitting. Therefore, the aforementioned three hyper-parameters, i.e., the initial learning rate, learning rate decay schedule and regularization coefficient, are optimized using the proposed PSO model for both I3D and MC3. We summarize the fine-tuned parameters and their respective search ranges for different networks in Table 3.

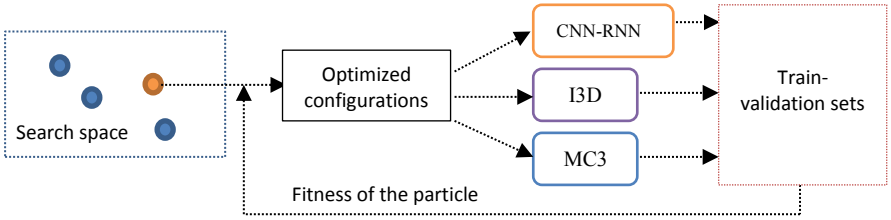

Figure 8 Interaction between a specific particle (in orange) and deep networks for fitness evaluation pertaining to hyper-parameter search (A set of optimized learning settings represented by a specific particle is used to set up a neural network whereby the validation accuracy rate is used as the fitness score of the current particle.)

Figure 8 depicts the interaction between a specific particle and deep networks for fitness evaluation with respect to hyper-parameter search. To be specific, for hyper-parameter search, we initialize a swarm with a dimension of two or three for the optimization of two or three hyper-parameters, e.g. in CNN-RNN, I3D and MC3 networks. A continuous search space is assigned for each dimension. For the fitness evaluation of each particle, each element of each search agent is converted into a hyper-parameter for the respective network. A set of optimized hyper-parameters represented by each particle is used to formulate the respective network. The resulting network is then trained using the training samples and tested using the validation instances. The accuracy rate of the validation dataset is utilized as the fitness score of each particle. The search strategies of each algorithm are used to update the position of each particle during the optimization. After reaching the maximum number of function evaluations, the most optimal hyper-parameter settings represented by the swarm leader are used to formulate the final network. This network, together with the identified optimal learning configuration, is then trained using the training samples and evaluated using the test instances. The detailed experimental studies are elaborated as follows.

5. ENSEMBLE MODEL CONSTRUCTION

In this research, we employ two new schemes for ensemble network construction, i.e. (1) a weighted scheme, i.e. ensemble scheme 1 (as discussed in Section 5.1), and (2) an evolving ensemble generation scheme devised by each optimization algorithm, i.e. ensemble scheme 2 (as depicted in Section 5.2). The former is able to effectively tackle class imbalanced classification problems, while the latter is able to eliminate weak or

redundant base classifiers and to minimize the ensemble sizes while maximizing performance. We employ the weighted scheme 1 to formulate ensemble models with the same types of base networks, while the evolving ensemble scheme 2 using optimization algorithms is used to devise ensemble classifiers integrating different types of base networks. In addition, for evolving ensemble generation scheme 2, each search algorithm is used to extract an optimal subset of base networks among all the base classifiers for ensemble network construction. After identifying the optimal subsets of base classifiers using each search algorithm, the above weighted ensemble scheme 1 (discussed in Section 5.1) is leveraged to integrate the results of these selected base networks and to form the ensemble prediction outcome. We introduce these two ensemble formulation schemes in detail, as follows.

5.1 Weighted Ensemble Construction

Firstly, we employ a weighted ensemble scheme (i.e. ensemble scheme 1) introduced by Zhang et al. [66], for ensemble model construction. Since our employed video deepfake datasets (e.g. Celeb-DFv2 and FaceForensics++) are extremely imbalanced for most test cases, as indicated in Zhang et al. [66], such a weighted ensemble strategy offers better capabilities in tackling samples with imbalanced class distributions. This weighted ensemble scheme is therefore employed in our studies to combine the predicted outputs from the same types of optimized base networks. Precisely, it is used to fuse the outputs of either a set of three optimized CNN-RNNs or a set of three optimized 3D CNNs. The detailed operations for this weighed ensemble scheme are explained, as follows.

During the training stage, we obtain the numbers of false positive (FP) and false negative (FN) instances of the training set from a classifier i, as denoted by M_{fp}^{i} and M_{fn}^{i}, respectively. The corresponding maximum numbers of the FP and FN samples are retrieved among the three optimized base classifiers, as represented by $M_{fp}^{\prime}$ and $M_{fn}^{\prime}$, respectively. Equation (23) is defined to generate the weight, w_i, of the base classifier i.

$$w_i = \left(1 - \frac{M_{fp}^{i}}{M_{fp}^{\prime}}\right) \times \left(1 - \frac{M_{fn'}^{i}}{M_{fn}^{\prime}}\right) \tag{23}$$

During the test stage, the weight w_i of base classifier i obtained from training is used to multiple with the respective prediction results for a test sample j. The summation of the resulting weighted prediction from each classifier is calculated for both real and fake classes, as indicated in Equations (24) and (25), respectively.

$$W_{real}^{j} = \sum_{i=1}^{B} w_i \times f(i_{real}) \tag{24}$$
$$W_{fake}^{j} = \sum_{i=1}^{B} w_i \times f(i_{fake}) \tag{25}$$

where $B = 3$ indicates the three optimized base networks and $f(i_{real})$ and $f(i_{fake})$ indicate the respective real and fake class prediction outputs from classifier i, respectively. If $W_{real}^{j} > W_{fake}^{j}$, the final prediction for sample j is real, otherwise synthetic. We employ this weighted ensemble scheme for homogeneous ensemble generation where three optimized base networks of the same types are used for ensemble generation. This weighted ensemble scheme is also used in evolving ensemble generation strategy 2 (presented in Section 5.2) to integrate the results of selected optimal subsets of networks to generate the ensemble prediction outcome.

5.2 Optimization Algorithm-based Evolving Ensemble Model Construction

Besides the weighted ensemble construction, the proposed PSO algorithm is employed for evolving ensemble generation by balancing between ensemble complexity and performance. We denote this optimization algorithm-based ensemble construction as ensemble development scheme 2, which is used to formulate ensemble models with different types of base networks (e.g. ensemble generation using optimized Inceptionv3-RNN, I3D and MC3 networks). Such an optimization algorithm-based ensemble model formulation enables the elimination of weak/redundant base classifiers to reduce cost, while maximizing performance.

After performing hyper-parameter optimization using each search method, we obtain optimized CNN-RNN, I3D and MC3 networks. We establish the base classifier pool by recruiting 30 best-performing models from different types of optimized networks. Therefore, a total of 30 base networks are used as the base classifiers. Each search method is subsequently used to identify the most optimal subset among these 30 base networks for evolving ensemble network construction. After identifying the optimal subset of base classifiers using each search method, the weighted ensemble scheme 1 discussed in Section 5.1 is used to integrate the outputs of these selected base networks for yielding the ensemble prediction outcome. We explain the detailed process for evolving ensemble generation, as follows.

Figure 9 depicts the evolutionary process for optimal ensemble construction during the training stage. Specifically, each search algorithm extracts an optimal subset of base classifiers among all the 30 base methods for ensemble network construction. The training and validation instances of each dataset are used for searching the optimal ensemble networks. Firstly, a swarm is initialized where each particle has a dimension of 30 representing 30 base networks. In other words, each element of the particle denotes a base classifier. A continuous search space in each dimension is used for base classifier selection. The search process is conducted using search operations of each optimization algorithm. For fitness evaluation of each particle, each dimension of a particle is transformed into a binary value by comparing it against a pre-defined threshold (e.g. 0.5), to determine the selection or elimination of a specific base classifier. After obtaining a set of selected base classifiers recommended by each particle, we employ the weighted ensemble formulation process defined in Equations (23)-(25) discussed in Section 5.1, to generate the ensemble prediction outcome with respect to the validation set by combining the results of the selected base networks.

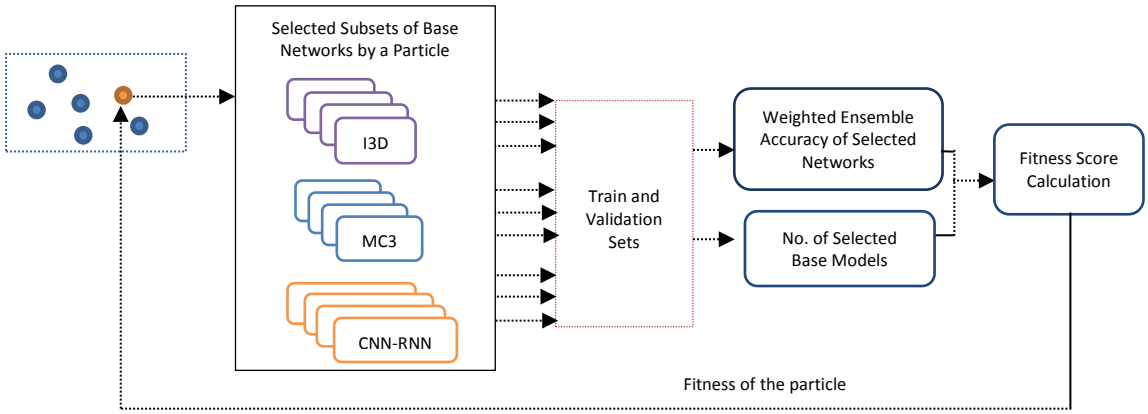

Figure 9 The evolutionary process for optimal ensemble construction in the training stage

Specifically, the weighting of each selected base model is calculated using Equation (23) based on its performance on the validation dataset. The ensemble prediction outcome fusing the results from all the selected base classifiers is obtained by using Equations (24)-(25). The resulting weighted ensemble accuracy rate of the above process on the validation dataset, as well as the number of the selected base classifiers, is used for fitness calculation as defined in Equation (26). This fitness evaluation formula is designed in such a way to reduce ensemble complexity by selecting an optimal subset of base classifiers, while increasing performance, as follows.

$$fitness_i = w_1 * accuracy_i + w_2 * (number\_base\_models_i)^{-1} \quad (26)$$

where $accuracy_i$ and $number\_base\_models_i$ represent the accuracy rate of the constructed ensemble on the validation dataset via the weighted scheme, and the number of selected base networks, respectively. In addition, we set the weighting factor w_1 for the ensemble accuracy rate as 0.9 and define the weighting coefficient w_2 for the number of selected base classifiers as 0.1. In other words, we employ a higher weighting ($w_1 = 0.9$) for the ensemble performance and a lower weighting ($w_2 = 0.1$) for the number of selected base networks, in order to ensure a higher priority of generating an ensemble model with a competitive performance than the generation of the smallest ensemble network. In this way, we take both criteria, i.e. the ensemble accuracy rate of the validation set and the number of selected base classifiers, into account to minimize computational cost while improving ensemble performance.

The final swarm leader obtained by the overall search process denotes the recommended most optimal subset of base networks for final ensemble model formulation. This identified most optimal ensemble model is then used for evaluation using unseen samples in the test set for each dataset. The final ensemble prediction outcome combining the outputs of all the selected base networks for the test set are also calculated using the weighted scheme shown in Equations (23)-(25). The above ensemble construction process is performed by each search method. The respective ensemble model results in terms of various evaluation metrics are used for performance comparison.

6. EXPERIMENTAL STUDIES

Several well-known synthetic video datasets have been used to test model efficiency, i.e., Celeb_DFv2 [1], FaceForensics++ [2] and Deepfakes (a subset of FaceForensics++) [2]. Celeb_DFv2 [1] comprises 590 original YouTube videos contributed by 59 celebrities from different age, gender, and ethnic groups. It also contains

5,639 corresponding synthetic videos generated using face swapping. We employ the official train-test split for this dataset in our studies. FaceForensics++ [2] consists of 1,000 real and 4,000 fake videos, where 1,000 synthetic videos are generated by each of the following generative methods, i.e., Deepfakes (i.e., deep learning based face identity manipulation), Face2Face (i.e., enacting facial expressions from one personal to another), FaceSwap (i.e., swapping faces using simple image processing methods) and NeuralTextures (i.e., deep learning based scene editing and static/dynamic content generation). In particular, Deepfakes [2] in FaceForensics++ has been regarded as the most challenging subset in comparison with counterparts yielded using other generative methods. Therefore, a performance comparison has also been conducted for this subset separately. For each of the above datasets, we employ a pre-processing procedure, i.e. Multi-Task Cascaded CNN (MTCNN) [81], to crop the facial regions automatically owing to the fact that facial regions are the targets of the attacks. A set of 50 frames is randomly extracted from each video for video authenticity classification.

For each dataset, optimal hyper-parameter and network topology selection has been performed for each network using the proposed PSO model. All the experiments employ the same maximum number of function evaluations, i.e., population (15) × iterations (20) = 300. Each search method is performed 10 times. Each network with identified optimized configurations is then trained using 25, 35, and 50 epochs, respectively.

Firstly, a weighted ensemble mechanism, i.e. the aforementioned ensemble scheme 1, is used to aggregate the results from three randomly selected optimized networks of the same types (i.e. the ensemble of CNN-RNN methods, the ensemble of I3D networks, or the ensemble of MC3 networks). In this way, we construct 10 ensemble models by using the resulting 30 optimized networks of a particular type. Evaluation metrics of the mean results of these 10 aggregated models are calculated for performance comparison.

In addition, to take advantage of different types of networks, e.g., CNN-RNN and 3D CNNs, besides constructing weighted ensemble models with the same types of base networks, cross-model ensemble formulation is implemented by embedding different types of base methods (e.g., the ensemble model consisting of Inceptionv3-RNN+I3D+MC3) for performance comparison. The optimization algorithm-based ensemble construction, i.e. the aforementioned ensemble scheme 2, is performed to formulate aggregation models with different types of optimized base networks. Specifically, we recruit the most performing 30 base networks from different types and an optimal subset is subsequently identified by an optimization algorithm from all the 30 base classifiers, for ensemble formulation.

A total of 14 baseline swarm intelligence and reinforcement learning methods are utilized in our experiments for performance comparison. These include, (i) classical search methods, i.e. PSO, GA, SA, FA [82], DA [83], CS [84], (ii) PSO variants, i.e. modified PSO with adaptive linear coefficients (MPSO) [44], PSO with elliptical coefficients (EPSO) [46], PSO with group-based autonomous search coefficients (AGPSO) [85], PSO integrated with Gravitational Search Algorithm (GSA) (PSOGSA) [86], PSO with random sine/cosine search coefficients and GA-based particle enhancement (RCPSO) [45], (iii) reinforcement learning algorithms, i.e. PPO [53, 56] and DDPG [55, 56], as well as (iv) a hybrid model integrating PSO with the reinforcement Q-learning algorithm (i.e. PSORL) [60]. Motivated by Lorenzo et al. [49], the original PSO model and all other search methods used in our experiments are equipped with parallel/simultaneous fitness evaluation for multiple individuals to reduce cost.

Specifically, PPO and DDPG are on-policy Actor-Critic (AC) reinforcement learning algorithms. Both algorithms learn an optimal policy that maximizes the total expected reward return. Each algorithm employs a pair of neural networks, i.e. a value-function critic and a policy-function actor, to approximate the cumulative long-term reward and determine optimal network hyper-parameters, respectively. In PPO, the Critic network employs the environmental observations (i.e. the fitness scores of individual particles in the swarm) as the input and produces a long-term reward return score as the output. Owing to the prediction of continuous optimal hyper-parameters, a continuous Gaussian actor is used in PPO to predict an optimal action (i.e. the network hyper-parameters) converted from a Gaussian distribution with environmental observations as the input. Precisely, the Actor network predicts the mean and standard deviations of the Gaussian distribution as the outputs based on the current observation. These predicted results are then transformed based on the Gaussian distribution to yield the respective valid optimized hyper-parameters for deep networks with respect to the fitness score and immediate reward calculation.

Moreover, DDPG also employs the Critic and Actor networks to learn an optimal policy by maximizing the total reward return. It leverages both the environmental observations (i.e. the fitness scores of individual particles in the swarm) and the action (i.e. the predicted hyper-parameters) as the inputs and produces the estimated cumulative reward as the scalar output. Instead of using Gaussian distribution-based sampling for optimal

parameter approximation as in PPO, the Actor network in DDPG predicts continuous optimal hyper-parameters as the output directly with environmental observations as the input. The DDPG and PPO agents are implemented using built-in functions in existing Python libraries in our experiments. Both Critic and Actor networks in DDPG and PPO are trained using a large number of episodes with 30 time steps for each episode. Once a threshold cumulative reward is reached over several successive episodes, the training process is concluded. The identified optimal hyper-parameters are used for performance comparison. For all other swarm-based optimization algorithms (e.g. classical search methods and PSO and FA variants), the search process is completed once the maximum number of function evaluations is fulfilled. Two evaluation metrics are used in our studies, i.e., the global accuracy rate and Area under the ROC (receiver operating characteristic curve) Curve (AUC) score, to assess the model performance.

The variable configurations of all the above baseline optimizers including swarm intelligence algorithms and reinforcement learning methods are extracted from their respective existing studies. Table 4 shows the parameter settings of the proposed PSO algorithm, which are mainly generated using mathematical formulae. Specifically, as discussed earlier, a new global search operation led by cross-breed leaders is defined in Equations (6)-(7), where adaptive weighting coefficients (i.e. α and β) are used for cross-breed leader formulation. Four sets of 3D formulae defined in Equations (8)-(10), (11)-(13), (14-16) and (17)-(19) are used to generate increasing and decreasing weighting coefficients, i.e. α and β, respectively. These increasing (α) and decreasing (β) weighting factors are utilized to adjust the effects of the global and personal best solutions for cross-breed leader generation to balance well between exploration and intensification. In addition, the respective increasing weighting coefficients α generated by the above four sets of 3D formulae is also used as the search parameter of the global search operation as defined in Equation (6). The swarm leader improvement is performed using two root-finding algorithms, i.e. Muller's method and fixed-point iteration algorithm, respectively.

Besides the aforementioned global search operation using Equations (6)-(7), a local search action is also developed as shown in Equation (20). It employs a new helix parameter δ defined in Equation (21) to fine-tune search steps to exploit search regions around the cross-breed leader. Finally, the reinforcement Q-learning algorithm is used to conduct the optimal selection between the global and local search actions provided in Equations (6)-(7) and (20)-(21) respectively, based on the Bellman principle defined in Equation (22).

Table 4 Parameter configurations of the proposed PSO model

Parameters of the propose model	Functions/processes used for parameter generation
Adaptive weighting coefficients (i.e. α and β) for cross-breed leader generation as defined in Equations (6)-(7)	Four sets of 3D formulae defined in Equations (8)-(10), (11)-(13), (14-16) and (17)-(19) are used to generate increasing and decreasing weighting coefficients, i.e. α and β, respectively, for cross-breed leader formulation.
The increasing weighting coefficient α is also used as the search parameter for the global search operation as defined in Equation (6).	The corresponding increasing weighting coefficient α generated by the above four sets of 3D formulae is also utilized as the search parameter in Equation (6).
A new helix parameter δ used as the search step in local search mechanism illustrated in Equations (20)-(21)	The local search parameter δ is produced using Equation (21).
Swarm leader improvement using two numerical analysis algorithms	Muller's method and the fixed-point iteration algorithm are used to improve the swarm leader as shown in Equations (3)-(4) and (5), respectively.
The selection of local and global search operations defined in Equations (6)-(7) and (20)-(21) respectively, using reinforcement learning	The dispatch of local and global search operations is determined using the reinforcement learning algorithm (the Q-learning algorithm) based on the long-term cumulative reward calculated using the Bellman optimality, as defined in Equation (22).

6.1 Evaluation Using the Celeb_DFv2 Dataset

We first employ the Celeb_DFv2 dataset to evaluate the optimized ensemble model comprising Inceptionv3-RNN, I3D and MC3 networks for synthetic video classification. As discussed earlier, we employ the official train-test split in our experiments. In the training set, we notice that the size of synthetic videos is significantly larger than that of the original ones. We duplicate the real samples in the training set 7.4 times in order to achieve a balanced distribution of both fake and real class instances. Note the videos in the unseen test set are not duplicated. The augmented training set is further divided into 80-20 for training and validation. The weighted ensemble formulation using ensemble scheme 1 with the same types of base networks, as well as dynamic ensemble development using ensemble scheme 2 with different types of base learners selected by each search method are introduced in detail, as follows. We utilize '+', '-', and '=' to indicate whether our resulting

ensemble models are better, worse, or the same as those devised by other search methods based on the Wilcoxon rank sum test.

6.1.1 Evaluation Using the CNN-RNN model

We first employ the Inceptionv3-RNN model for undertaking deepfake detection. Different types of RNNs with different number of hidden units are produced using the proposed PSO model. Multiple trials are performed for each optimizer. The network with identified optimal settings is trained using 25, 35 and 50 epochs, respectively. As mentioned earlier, we employ the first weighted ensemble scheme 1 discussed in Section 5.1 to aggregate three yielded Inceptionv3-RNN models within an ensemble. The mean results of 10 such weighted ensembles are used for performance comparison. A weighted ensemble model comprising Inceptionv3-BiLSTM, Inceptionv3-LSTM and Inceptionv3-GRU, with a constant number (1800) of hidden neurons, is also utilized as a default baseline method for performance comparison. Table 5 shows the detailed results of the weighted ensemble Inceptionv3-RNN models devised by each search method.

Table 5 Mean results of the weighted ensemble Inceptionv3-RNN models using Celeb-DFv2

Methods	Ensemble topologies	Mean accuracy rates	Mean AUC	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregated Inceptionv3-RNN model	0.7992	0.7105	n/a
MPSO-based Ensemble	MPSO + aggregated Inceptionv3-RNN model	0.7722	0.6685	+
EPSO-based Ensemble	EPSO + aggregated Inceptionv3-RNN model	0.7780	0.6770	+
PSO-based Ensemble	PSO + aggregated Inceptionv3-RNN model	0.7355	0.6272	+
CS-based Ensemble	CS + aggregated Inceptionv3-RNN model	0.7587	0.6502	+
GA-based Ensemble	GA + aggregated Inceptionv3-RNN model	0.7568	0.6528	+
SA-based Ensemble	SA + aggregated Inceptionv3-RNN model	0.7490	0.6362	+
FA-based Ensemble	FA + aggregated Inceptionv3-RNN model	0.7896	0.6938	+
DA-based Ensemble	DA + aggregated Inceptionv3-RNN model	0.7568	0.6461	+
RCPSO-based Ensemble	RCPSO + aggregated Inceptionv3-RNN model	0.7394	0.6395	+
AGPSO-based Ensemble	AGPSO + aggregated Inceptionv3-RNN model	0.7625	0.6545	+
PSOGSA-based Ensemble	PSOGSA + aggregated Inceptionv3-RNN model	0.7568	0.6514	+
PPO-based Ensemble	PPO + aggregated Inceptionv3-RNN model	0.7683	0.6629	+
DDPG-based Ensemble	DDPG + aggregated Inceptionv3-RNN model	0.7625	0.6558	+
PSORL-based Ensemble	PSORL + aggregated Inceptionv3-RNN model	0.7606	0.6557	+
Default Ensemble Model	Aggregation of Inceptionv3-BiLSTM, Inceptionv3-LSTM and Inceptionv3-GRU with default settings	0.7336	0.6204	+

Table 5 depicts ensemble performance of each optimizer. Our devised weighted ensemble Inceptionv3-RNN models obtain a better performance than those of the counterparts optimized by all other classical search methods and PSO variants. Moreover, among the baseline methods, FA, PPO, MPSO and EPSO-based ensemble networks are comparatively more effective with better accuracy and AUC scores, while DA, RCPSO, SA and PSO-based ensemble models have the lowest accuracy or AUC results. Owing to the embedding of diverse base networks with different optimized configurations, the ensemble Inceptionv3-RNNs generated by all search methods outperform those with default settings in most test cases. The statistical test is also performed to compare the accuracy result distributions of our algorithm and those from other search methods. The '+' symbol in Table 5 indicates the statistical significance of our ensemble against those devised by other search methods.

Table 6 Optimized network topologies of Inceptionv3-RNN generated by each optimizer using Celeb-DFv2

	Layer type	**No. of hidden units**
Prop. PSO	BiLSTM	1342.67
MPSO	BiLSTM/GRU	1372.33
EPSO	BiLSTM/GRU	1232.00
PSO	LSTM/GRU	1752.50
CS	LSTM	1172.55
GA	BiLSTM	1155.00
SA	LSTM/GRU	1578.15
FA	BiLSTM/LSTM	1266.28
DA	LSTM/GRU	1487.54
RCPSO	BiLSTM/LSTM	1020.45
AGPSO	LSTM	1443.89
PSOGSA	LSTM	1176.97
PPO	LSTM	1237.55
DDPG	LSTM/GRU	1393.71
PSORL	BiLSTM/GRU	1205.03
Default	BiLSTM/LSTM/GRU	1800

Table 6 shows the identified network configurations for the Inceptionv3-RNN models. The proposed PSO model and GA construct BiLSTM networks in majority of the test cases. MPSO, EPSO and PSORL yield RNN models with BiLSTM and GRU layers, while FA and RCPSO generate recurrent networks with BiLSTM and LSTM layers. CS, AGPSO, PSOGSA, and PPO establish networks with LSTM layers in most cases, with SA, DA, DDPG and PSO formulating models mainly with LSTM and GRU layers. Besides the above, the smallest numbers of hidden units are embedded into the RCPSO, GA, CS and PSOGSA-yielded BiLSTM/LSTM networks, while the largest numbers of hidden nodes are included in the DA, AGPSO, SA and PSO-optimized LSTM/GRU models. Therefore, the former networks are more likely to overlook important temporal features, while the latter methods are more inclined to capture redundant excessive details for manipulated video classification. Moderate numbers of hidden units are employed by the proposed PSO, DDPG, MPSO, FA, PPO and EPSO-optimized BiLSTM/LSTM/GRU networks, which indicate better discriminative dynamic sequential feature extraction and generalization capabilities. The default network employs the largest number of hidden units which may result in extracting excessive details.

6.1.2 Evaluation Using the I3D Network

The 3D CNNs are also used for video forgery classification. We optimize the I3D network by identifying the optimal learning options (i.e., the initial learning rate, learning rate decay factor and regularization coefficient) using the proposed PSO method. Each optimized network is subsequently trained using 25, 35, and 50 epochs, respectively. Next, the ensemble scheme 1, i.e. the weighted ensemble formation, is used to aggregate the optimized I3D base networks. A default weighted baseline ensemble model with fixed learning configurations is also implemented, where we assign learning rate, learning rate decay factor and regularization coefficient with 0.001, 0.1 and 0.0001, respectively.

Table 7 Mean results of the weighted ensemble I3D networks using Celeb-DFv2

Methods	Ensemble topologies	Mean accuracy rates	Mean AUC	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregated I3D model	0.9093	0.8747	n/a
MPSO-based Ensemble	MPSO + aggregated I3D model	0.8571	0.7975	+
EPSO-based Ensemble	EPSO + aggregated I3D model	0.8591	0.8257	+
PSO-based Ensemble	PSO + aggregated I3D model	0.8397	0.7695	+
CS-based Ensemble	CS + aggregated I3D model	0.8378	0.7667	+
GA-based Ensemble	GA + aggregated I3D model	0.8552	0.7893	+
SA-based Ensemble	SA + aggregated I3D model	0.8327	0.8422	+
FA-based Ensemble	FA + aggregated I3D model	0.8822	0.8514	+
DA-based Ensemble	DA + aggregated I3D model	0.8436	0.8019	+
RCPSO-based Ensemble	RCPSO + aggregated I3D model	0.8745	0.8696	+
AGPSO-based Ensemble	AGPSO + aggregated I3D model	0.8340	0.7624	+
PSOGSA-based Ensemble	PSOGSA + aggregated I3D model	0.8378	0.7654	+
PPO-based Ensemble	PPO + aggregated I3D model	0.8359	0.7679	+
DDPG-based Ensemble	DDPG + aggregated I3D model	0.8533	0.7945	+
PSORL-based Ensemble	PSORL + aggregated I3D model	0.8436	0.7872	+
Default Ensemble Model	Aggregation of I3D with default settings	0.8475	0.7808	+

We present the results of the optimized I3D ensemble models in Table 7. For all search methods, the resulting weighted ensemble I3D networks show better feature learning capabilities with more competitive performance than those of devised ensemble Inceptionv3-RNN methods. The proposed PSO-based ensemble I3D networks outperform those yielded by all other search methods with statistical significance, owing to the search processes inspired by diverse cross-breed leaders, petal search trajectories, reinforcement learning-based search action selection, as well as numerical recursive leader enhancement. In addition, FA, RCPSO, SA, DA, and EPSO-devised ensemble models achieve a better performance than those formulated by MPSO, PSO, GA, PSOGSA, CS, PSORL and AGPSO, as well as those built using reinforcement learning algorithms, i.e. DDPG and PPO. DDPG-based ensemble networks also outperform those yielded by PPO, PSO, CS, PSOGSA and AGPSO. Optimized ensemble I3D models generated by each search method show better capabilities in distinguishing fake from real videos than those from the default networks in most test cases.

Table 8 Optimized learning configurations of the I3D networks generated by each optimizer using Celeb-DFv2

	Learning rate	**Learning rate drop factor**	**Weight decay**
Prop. PSO	0.007576	0.06452	0.000585
MPSO	0.003942	0.06123	0.000594
EPSO	0.006771	0.05882	0.000399

PSO	0.009465	0.05128	0.000175
CS	0.008310	0.01290	0.000341
GA	0.001800	0.07000	0.000900
SA	0.003270	0.06755	0.000883
FA	0.005800	0.02132	0.000116
DA	0.003320	0.05421	0.000229
RCPSO	0.005550	0.01102	0.000550
AGPSO	0.004860	0.03853	0.000754
PSOGSA	0.009260	0.06603	0.000382
PPO	0.008210	0.04608	0.000633
DDPG	0.005120	0.04836	0.000505
PSORL	0.001555	0.06316	0.000573
I3D (default)	0.001000	0.10000	0.000100

Table 8 shows the key learning configurations identified by each algorithm. The proposed model obtains a comparatively moderate mean initial learning rate with a larger mean learning rate drop factor as well as a larger average weight decay. As such, the resulting I3D networks are equipped with effective learning steps for pattern extraction and have better capabilities in avoiding over-fitting. In contrast, PSO, PSOGSA, PPO and CS methods identify the largest initial learning rates with the small or moderate learning rate decay schedules in most cases, which are more likely to cause oscillations in weight updates. Similar to the proposed optimizer, the efficiency of the ensemble networks generated using FA, RCPSO, SA, DA, and EPSO is dependent on the moderate initial learning rates, which lead to refined but effective updates of the learning mechanisms. GA and PSORL produce the smallest initial learning rates with comparatively larger decreasing drop schedule rates, which may result in model under-fitting. Similarly, the network with the default learning settings is more likely to be under-fitted, because of the adoption of a small initial learning rate and a large decreasing decay rate, in combination with a very small regularization factor. In addition, the performance of the default ensemble model is limited by the fixed model configurations.

6.1.3. Evaluation Using the MC3 Network

Another 3D CNN, i.e. MC3, is also used for performance comparison. We optimize the same learning options, i.e. learning rate, learning rate decay factor and regularization coefficient, of the MC3 model. A similar experimental setting is utilized with a set of 10 trials. The weighted ensemble scheme 1, discussed in Section 5.1, is also utilized for ensemble construction with optimized MC3 networks as the base classifiers. The mean results of the resulting 10 weighted ensemble models with each comprising three optimized MC3 networks are used for performance comparison.

Table 9 Mean results of the weighted ensemble MC3 networks using Celeb-DFv2

Methods	Ensemble topologies	Mean accuracy rates	Mean AUC	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregated MC3 model	0.8890	0.8546	n/a
MPSO-based Ensemble	MPSO + aggregated MC3 model	0.8475	0.7982	+
EPSO-based Ensemble	EPSO + aggregated MC3 model	0.8282	0.7754	+
PSO-based Ensemble	PSO + aggregated MC3 model	0.8532	0.8012	+
CS-based Ensemble	CS + aggregated MC3 model	0.8147	0.7611	+
GA-based Ensemble	GA + aggregated MC3 model	0.8687	0.8384	+
SA-based Ensemble	SA + aggregated MC3 model	0.8263	0.7472	+
FA-based Ensemble	FA + aggregated MC3 model	0.8591	0.7949	+
DA-based Ensemble	DA + aggregated MC3 model	0.8398	0.7923	+
RCPSO-based Ensemble	RCPSO + aggregated MC3 model	0.8571	0.7921	+
AGPSO-based Ensemble	AGPSO + aggregated MC3 model	0.8436	0.8033	+
PSOGSA-based Ensemble	PSOGSA + aggregated MC3 model	0.8417	0.7710	+
PPO-based Ensemble	PPO + aggregated MC3 model	0.8745	0.8308	+
DDPG-based Ensemble	DDPG + aggregated MC3 model	0.8668	0.8169	+
PSORL-based Ensemble	PSORL + aggregated MC3 model	0.8726	0.8333	+
Default Ensemble Model	Aggregation of MC3 with default settings	0.8301	0.7702	+

Table 9 presents the performance of the optimized weighted ensemble MC3 networks for all the search methods. These resulting models show a better performance than those of optimized Inceptionv3-RNN architectures, but an inferior performance to those of devised I3D networks, for most search methods. The proposed PSO-based weighted ensemble MC3 networks depict a competitive performance, outperforming the counterparts generated by all other search methods with statistical significance. In addition, GA, PPO, DDPG, PSORL and AGPSO-devised networks outperform those optimized by all other baseline methods, while SA, CS and PSOGSA-based ensemble models show least efficiency. Because of the diverse optimized base model configurations recommended by all the search methods, their resulting weighted ensemble networks outperform those with default parameter configurations in most test cases.

Table 10 Optimized learning configurations of the MC3 networks generated by each optimizer using Celeb-DFv2

	Learning rate	Learning rate drop factor	Weight decay
Prop. PSO	0.001556	0.05882	0.000183
MPSO	0.002669	0.04762	0.000727
EPSO	0.003698	0.04545	0.000311
PSO	0.002856	0.04762	0.000783
CS	0.007731	0.05025	0.000679
GA	0.001397	0.04781	0.000596
SA	0.008977	0.05064	0.000824
FA	0.003323	0.03656	0.000225
DA	0.003485	0.05570	0.000814
RCPSO	0.002622	0.02067	0.000662
AGPSO	0.006592	0.01465	0.000230
PSOGSA	0.009214	0.03669	0.000735
PPO	0.001060	0.05638	0.000490
DDPG	0.004852	0.01269	0.000520
PSORL	0.001191	0.03197	0.000332
MC3 (default)	0.001	0.1	0.0001

As shown in Table 10, the initial learning rate setting in the MC3 model has more effects on the model performance in comparison with those of other parameters. The empirical results indicate that the best results are correlated with small or moderate initial learning rates, while large learning rates tend to result in poor accuracy rates. Specifically, the proposed algorithm, GA, PPO and PSORL select the smallest initial learning rates with the large/moderate learning rate decay factors as compared with those obtained by other search methods. The detailed results indicate that such learning configurations are able to improve the network performance steadily over a number of learning epochs. The largest mean initial learning rates are extracted by SA, CS and PSOGSA, and their resulting networks are more likely to overlook the global optima and suffer from oscillations in gradient descent as the training progresses. DDPG and AGPSO with moderate initial learning rates also show competitive performance with effective learning steps to update the learning mechanisms. The MC3 ensemble with default parameter settings also achieves a reasonable performance. However, because of the adoption of the fixed base model configurations, their resulting weighted ensemble networks have limited diversity with comparatively small performance improvements.

6.1.4. Ensemble Model Construction Using Optimization Algorithms

We subsequently use the optimization algorithm-based scheme for ensemble formulation, i.e. ensemble scheme 2, provided in Section 5.2. In other words, we employ each search method to identify an optimal subset of Inceptionv3-RNN, I3D and MC3 networks with optimized settings for ensemble model construction.

As mentioned earlier, for hyper-parameter search, we perform 10 runs for each method. For each identified configuration, we train the respective network using three different epochs, i.e. 25, 35, and 50 epochs. For the base classifier pool construction for each search method, we select the top 10 Inceptionv3-RNN, top 10 I3D and top 10 MC3 networks on the validation set with the corresponding optimal settings obtained from the previous process. The selected 30 base classifiers with respective optimized settings are used for ensemble model construction by each search method. We explain the optimal ensemble construction process in detail, as follows.

Specifically, each search method is used to extract an optimal subset of base classifiers among all the 30 base methods for ensemble network construction. The training and validation datasets of Celeb-DFv2 are utilized for searching the optimal subsets of base networks. As mentioned earlier, each search agent in each search method has a dimension of 30 with each element representing a base classifier. The optimization process is performed using search operations within each search method. For the fitness evaluation of each particle, as discussed earlier, we first compare the value of each dimension against a threshold value (e.g. 0.5) to determine the selection or omission of the respective base classifier. After obtaining the selected subset of networks, the weighted ensemble strategy shown in Equations (23)-(25) is used to fuse the results of all the selected base classifiers. The resulting validation accuracy rate along with the number of selected base networks is employed for fitness calculation, as shown in Equation (26). After the completion of the required number of function evaluations, the swarm leader representing the most optimal subset of the selected base classifiers is attained. This subset of the selected most optimal base networks is used to form the final optimized ensemble, which is used to evaluate unseen samples in the test set. Their weighted ensemble results calculated using Equations (23)-(25) are used for performance comparison.

The optimal base classifier selection is performed using the following configurations, i.e. a dimension of 30 and a maximum number of function evaluations of 1,000 (population=20 and iterations=50). The mean result of a set of 30 runs is produced for each search method. A default ensemble model is also formulated by integrating 30 base networks with 10 I3D, 10 MC3 and 10 CNN-RNN models using default learning configurations.

Table 11 depicts the mean ensemble performance comparison over 30 runs. As indicated in Table 11, for all search methods, the aggregation of three different types of optimized base learners (i.e., Inceptionv3-RNN, I3D and MC3) with different learning mechanisms using the evolving ensemble construction method further boosts the performance and achieves the best results for synthetic video classification, in comparison with those from other devised homogeneous ensembles with the same types of base networks. In particular, our optimized ensemble model aggregating three different types of networks outperforms the counterparts yielded by all other search methods, as evidenced by the statistical test results. Most search methods select 16-22 base classifiers among the 30 respective distinctive base networks. The proposed PSO algorithm identifies moderate numbers (e.g. 16.76) of base models for ensemble network construction, as compared with those devised by other search methods, resulting in improved efficiency while maintaining sufficient diversity. CS and AGPSO develop the largest ensemble models with 21.8 and 21.1 selected base classifiers on average, respectively. These ensemble models with the largest numbers of base networks are highly likely to cause redundancy and contradiction in decision making, therefore obtaining lower accuracy rates and AUC scores. In contrast, MPSO and PSOGSA form the smallest ensemble networks with 15.2 and 16.4 selected base classifiers on average, respectively, which may limit network diversity. Similar to the proposed model, RCPSO, FA and GA identify moderate numbers of base networks, i.e. 17.5, 17 and 17.6, respectively, which balance well between robustness and cost. Most ensemble models built by search methods with optimized numbers of base networks outperform the default ensemble network with all 30 base classifiers. This is owing to high flexibility and robustness in the optimized ensemble models yielded by most search algorithms.

Table 11 Mean results for optimal ensemble networks integrating Inceptionv3-RNN, I3D and MC3 devised by each search method for Celeb-DFv2

Methods	Ensemble topologies	Mean accuracy rates	Mean AUC	Ensemble size	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregation of Inceptionv3-RNN, I3D and MC3	0.9498	0.9270	16.76	n/a
MPSO-based Ensemble	MPSO + aggregation of Inceptionv3-RNN, I3D and MC3	0.8919	0.8454	15.20	+
EPSO-based Ensemble	EPSO + aggregation of Inceptionv3-RNN, I3D and MC3	0.8958	0.8523	17.20	+
PSO-based Ensemble	PSO + aggregation of Inceptionv3-RNN, I3D and MC3	0.8900	0.8466	20.60	+
CS-based Ensemble	CS + aggregation of Inceptionv3-RNN, I3D and MC3	0.8745	0.8228	21.80	+
GA-based Ensemble	GA + aggregation of Inceptionv3-RNN, I3D and MC3	0.8972	0.8785	17.60	+
SA-based Ensemble	SA + aggregation of Inceptionv3-RNN, I3D and MC3	0.8745	0.8509	19.40	+
FA-based Ensemble	FA + aggregation of Inceptionv3-RNN, I3D and MC3	0.9151	0.8858	17.00	+
DA-based Ensemble	DA + aggregation of Inceptionv3-RNN, I3D and MC3	0.8938	0.8669	20.20	+
RCPSO-based Ensemble	RCPSO + aggregation of Inceptionv3-RNN, I3D and MC3	0.9163	0.9079	17.50	+
AGPSO-based Ensemble	AGPSO + aggregation of Inceptionv3-RNN, I3D and MC3	0.8770	0.8425	21.10	+
PSOGSA-based Ensemble	PSOGSA + aggregation of Inceptionv3-RNN, I3D and MC3	0.8764	0.8390	16.40	+
PPO-based Ensemble	PPO + aggregation of Inceptionv3-RNN, I3D and MC3	0.8919	0.8614	18.38	+
DDPG-based Ensemble	DDPG + aggregation of Inceptionv3-RNN, I3D and MC3	0.8996	0.8673	18.56	+
PSORL-based Ensemble	PSORL + aggregation of Inceptionv3-RNN, I3D and MC3	0.8996	0.8713	17.29	+
Default Ensemble Model	Aggregation of Inceptionv3-RNN, I3D and MC3 with default settings	0.888	0.8438	30.00	+

6.2 Evaluation Using the FaceForensics++ Dataset

We have evaluated the proposed PSO-optimized I3D and MC3 networks using FaceForensics++, owing to their impressive performance in comparison with optimized inceptionv3-RNN models, as evidenced in the previous experiments. Besides 1,000 original YouTube videos, the FaceForensics++ dataset contains 4,000 manipulated videos with 1,000 forged videos generated using FaceSwap, Deepfakes, NeuralTextures, and Face2Face, respectively. We combine all the synthetic videos yielded using different techniques into one fake video set of 4,000. As such, a dataset of 1,000 original and 4,000 synthetic videos is studied in this experiment. As mentioned earlier, the I3D and MC3 networks are employed owing to their superiority over the Inceptionv3-RNN network. As recommended in existing studies [15], both 3D CNNs employ a 60-20-20 train-validation-test split of each class for model evaluation. The original authentic videos in the training and validation sets are duplicated a number of times to ensure balanced numbers of authentic and synthetic videos for training and validation. Such a duplication process is not applied to the test set. We present the detailed ensemble results for each search method, as follows. The weighted ensemble classifiers with the same types of the base networks, as well as ensemble networks with optimal subsets of different types of base learners devised by each search

method, are formulated. The detailed ensemble performance comparison for both ensemble schemes is provided below.

6.2.1 Evaluation Using the I3D Network

We optimize I3D using each search method for video forgery classification using FaceForensics++. The same experimental setting used for Celeb-DFv2 is also used in this experiment. We repeat hyper-parameter search 10 times for each search method. Each devised network is trained using several different training epochs. The weighted ensemble scheme 1 presented in Section 5.1 is used for ensemble model formation using optimized I3D networks. Specifically the weighted strategy defined in Equation (23)-(25) is used to combine the outputs of three optimized I3D networks in one ensemble classifier. The mean result of these weighted ensemble models is used for performance comparison. Table 12 summarizes the weighted ensemble result comparison between different search methods.

Table 12 Mean results of the weighted ensemble I3D networks using FaceForensics++

Methods	Ensemble topologies	Mean Accuracy rates	Mean AUC	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregated I3D model	0.9255	0.9281	n/a
MPSO-based Ensemble	MPSO + aggregated I3D model	0.9040	0.9025	+
EPSO-based Ensemble	EPSO + aggregated I3D model	0.9030	0.8830	+
PSO-based Ensemble	PSO + aggregated I3D model	0.8920	0.8519	+
CS-based Ensemble	CS + aggregated I3D model	0.8710	0.9194	+
GA-based Ensemble	GA + aggregated I3D model	0.8975	0.9003	+
SA-based Ensemble	SA + aggregated I3D model	0.8920	0.8870	+
FA-based Ensemble	FA + aggregated I3D model	0.8723	0.8916	+
DA-based Ensemble	DA + aggregated I3D model	0.8978	0.8855	+
RCPSO-based Ensemble	RCPSO + aggregated I3D model	0.9179	0.8532	+
AGPSO-based Ensemble	AGPSO + aggregated I3D model	0.8920	0.8369	+
PSOGSA-based Ensemble	PSOGSA + aggregated I3D model	0.8880	0.8363	+
PPO-based Ensemble	PPO + aggregated I3D model	0.8860	0.8425	+
DDPG-based Ensemble	DDPG + aggregated I3D model	0.8850	0.8975	+
PSORL-based Ensemble	PSORL + aggregated I3D model	0.9002	0.9001	+
Default Ensemble Model	Aggregation of I3D with default settings	0.8790	0.8513	+

As depicted in Table 12, our weighted ensemble I3D networks achieve a reliable performance for identifying both original and manipulated samples with statistical significance in performance. From the AUC score perspective, MPSO, CS, GA, and PSORL-based ensemble models are comparatively more effective than those generated by the remaining baseline search methods, as well as those constructed by the RL methods, i.e. PPO and DDPG. In addition, DDPG-based weighted ensemble networks outperform those formulated by EPSO, PSO, SA, FA, DA, RCPSO, AGPSO, PSOGSA, and PPO, respectively. AGPSO and PSOGSA-devised ensemble networks achieved the least efficiency. In addition, owing to the adoption of the same base model configurations with the default configurations, the weighted default ensemble model is less competitive than those with optimal learning settings yielded by most of the search methods.

Table 13 Optimized learning configurations of the I3D networks generated by each optimizer using FaceForensics++

	Learning rate	**Learning rate drop factor**	**Weight decay**
Prop. PSO	0.001223	0.06667	0.000202
MPSO	0.001336	0.08333	0.000141
EPSO	0.001584	0.05000	0.000223
PSO	0.003061	0.05556	0.000806
CS	0.001710	0.06074	0.000526
GA	0.001090	0.05700	0.000866
SA	0.005198	0.01433	0.000170
FA	0.004456	0.02223	0.000887
DA	0.002491	0.04967	0.000537
RCPSO	0.003349	0.01664	0.000816
AGPSO	0.008467	0.02276	0.000776
PSOGSA	0.009693	0.02850	0.000694
PPO	0.006274	0.06833	0.000129
DDPG	0.005353	0.03777	0.000743
PSORL	0.001329	0.01813	0.000320
I3D (default)	0.001	0.1	0.0001

The hyper-parameters identified by each algorithm are listed in Table 13. The empirical results indicate the preferences of the smaller learning rate, moderate/large learning rate drop factor and moderate weight decay parameters, which are correlated with an improved network performance pertaining to fake/real video classification. Such settings are favoured by the proposed model, MPSO, EPSO, CS, GA and PSORL with competitive/reasonable classification performance for both original and manipulated video classes. Comparatively, larger learning rates are yielded by AGPSO, PSOGSA, and PPO, therefore the performance of their resulting networks is significantly affected by large oscillations in weight updates, leading to suboptimal outcomes. The default I3D network adopts a small learning rate, which allows granular learning steps to improve network performance, but it shows limited capabilities in tackling over-fitting by using a small weight decay factor. Owing to the adoption of same base network configurations, its resulting weighted ensemble model depicts limited flexibility and complementary properties to boost network performance.

6.2.2 Evaluation Using the MC3 Network

We also evaluate the proposed PSO-optimized ensemble MC3 network using FaceForensics++. Again we repeat hyper-parameter search 10 times for each search method. Each optimized network is trained using several larger numbers of training epochs. The aggregation of devised MC3 networks is performed by using the weighted ensemble strategy shown in Equations (23)-(25). Table 14 shows the detailed ensemble performance with respect to each algorithm.

Table 14 Mean results of the weighted ensemble MC3 networks using FaceForensics++

Methods	Ensemble topologies	Mean accuracy rates	Mean AUC	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregated MC3 model	0.9200	0.9156	n/a
MPSO-based Ensemble	MPSO + aggregated MC3 model	0.8910	0.8475	+
EPSO-based Ensemble	EPSO + aggregated MC3 model	0.8893	0.8752	+
PSO-based Ensemble	PSO + aggregated MC3 model	0.8745	0.8794	+
CS-based Ensemble	CS + aggregated MC3 model	0.8916	0.9045	+
GA-based Ensemble	GA + aggregated MC3 model	0.8747	0.8863	+
SA-based Ensemble	SA + aggregated MC3 model	0.8760	0.8306	+
FA-based Ensemble	FA + aggregated MC3 model	0.8710	0.8988	+
DA-based Ensemble	DA + aggregated MC3 model	0.8810	0.8628	+
RCPSO-based Ensemble	RCPSO + aggregated MC3 model	0.8841	0.8363	+
AGPSO-based Ensemble	AGPSO + aggregated MC3 model	0.8650	0.8350	+
PSOGSA-based Ensemble	PSOGSA + aggregated MC3 model	0.8610	0.8288	+
PPO-based Ensemble	PPO + aggregated MC3 model	0.8870	0.8394	+
DDPG-based Ensemble	DDPG + aggregated MC3 model	0.8720	0.8338	+
PSORL-based Ensemble	PSORL + aggregated MC3 model	0.8850	0.8400	+
Default Ensemble Model	Aggregation of MC3 with default settings	0.8670	0.8250	+

Table 14 depicts the mean weighted ensemble network performance regarding each search method. The default and devised MC3 networks perform generally worse than those of the respective I3D models. Our yielded MC3 weighted ensemble models outperform the counterparts generated by all other search methods with a statistical margin in performance. The root-finding algorithm driven swarm leader enhancement, Q-learning based search action selection and hybrid leader-motivated local and global search mechanisms offer better capabilities in tackling local optima traps. CS, FA, GA, PSO and EPSO-devised weighted ensemble networks achieve better AUC scores in comparison with those of the networks optimized by other baseline search methods. PPO-based ensemble networks achieve more competitive accuracy and AUC results than those of RCPSO, AGPSO, DDPG, SA, and PSOGSA-based models. The weighted ensemble networks constructed by all search methods outperform the counterpart with default settings in terms of the AUC scores.

Table 15 Optimized learning configurations of the MC3 networks generated by each optimizer using FaceForensics++

	Learning rate	**Learning rate drop factor**	**Weight decay**
Prop. PSO	0.001753	0.05000	0.000327
MPSO	0.002563	0.05556	0.000316
EPSO	0.002097	0.04545	0.000205
PSO	0.002026	0.04545	0.000194
CS	0.001488	0.03163	0.000631
GA	0.001013	0.03264	0.000762
SA	0.005916	0.06968	0.000786
FA	0.001220	0.05347	0.000899
DA	0.002269	0.03278	0.000117
RCPSO	0.003730	0.01416	0.000397

AGPSO	0.004340	0.05735	0.000708
PSOGSA	0.006131	0.01225	0.000213
PPO	0.002940	0.05566	0.000434
DDPG	0.003964	0.01129	0.000397
PSORL	0.002973	0.02541	0.000124
MC3 (default)	0.001	0.1	0.0001

Table 15 illustrates key learning options yielded by each method. The most robust networks are yielded by the proposed PSO model with identified moderate mean learning rate, moderate mean learning decay factor and moderate mean weight decay configurations. The resulting networks conduct a refined but effective adjustment to the network weights while deploying reasonable penalties to the loss function via the regularization term to reduce over-fitting. FA, GA and CS identify smallest learning rates, which may prevent their optimized networks from achieving the best performances within the required number of iterations. RCPSO, DDPG, AGPSO, SA, and PSOGSA extract comparatively larger learning rates. Thus, their resulting networks may experience oscillatory behaviours over epochs with comparatively more significant learning steps, and are prone to a fast convergence toward suboptimal solutions. All the devised ensemble networks of each search method embed a variety of base network learning settings in comparison with those of the default ensemble model, therefore illustrating better performances.

6.2.3. Ensemble Model Construction Using Optimization Algorithms

After obtaining optimized I3D and MC3 networks, ensemble model construction using optimization algorithms, i.e. the ensemble scheme 2 discussed in Section 5.2, is used to build fusion models with different types of base networks. Specifically, each search algorithm is subsequently used to construct optimized ensemble models. As mentioned earlier, since we employ a set of 10 runs for each search method, and each optimized I3D/MC3 network with the identified configurations are trained using three different numbers (i.e. 25, 35 and 50) of epochs, we obtain 30 optimized I3D and 30 optimized MC3 networks with respect to each search algorithm. The top 15 optimized I3D and top 15 optimized MC3 networks on the validation dataset are used to construct the ensemble base classifier pool.

Each search algorithm is then used to extract an optimal subset of base classifiers among all the 30 base methods for ensemble network construction. Again, each particle has a dimension of 30 with each element representing a base classifier. The optimization process is performed using search operations within each search method, with a maximum number of function evaluations of 1,000. Such a search process is repeated 30 times for each method. For fitness evaluation of each particle, we first compare the value of each dimension against a threshold value (e.g. 0.5) to determine the selection or omission of the respective base classifier. After obtaining the selected subset of networks, the weighted ensemble strategy shown in Equations (23)-(25) is used to aggregate the results of all the selected base classifiers. The resulting validation accuracy rate along with the number of selected base networks is used for fitness calculation as shown in Equation (26). After the completion of the search process, the swarm leader representing the most optimal subset of the selected base classifiers is obtained. This subset of the selected most optimal base networks is used to form the final optimized ensemble, which is used to evaluate unseen samples in the test set. The weighted ensemble results calculated using Equations (23)-(25) are used for performance comparison.

Table 16 presents the mean ensemble results of the selected devised I3D and MC3 networks by each search method over 30 runs. Because of the optimal selection of two types of base networks in combination with unique optimized learning configurations of each base classifier, the resulting hybrid ensemble models show significant robustness and obtain a better performance. For all search methods, these optimized hybrid ensemble networks outperform those constructed purely by either optimized I3D or MC3 networks. Moreover, the statistical test results indicate the significance in performance of our hybrid ensembles over the counterparts yielded by other search methods.

Table 16 Mean results for optimal ensemble networks integrating I3D and MC3 devised by each search method for FaceForensics++

Methods	Ensemble topologies	Mean accuracy rates	Mean AUC	Ensemble size	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregation of I3D and MC3	0.9620	0.9425	20.0	n/a
MPSO-based Ensemble	MPSO + aggregation of I3D and MC3	0.9200	0.9156	22.0	+
EPSO-based Ensemble	EPSO + aggregation of I3D and MC3	0.9390	0.8963	20.8	+
PSO-based Ensemble	PSO + aggregation of I3D and MC3	0.9130	0.8856	23.1	+
CS-based Ensemble	CS + aggregation of I3D and MC3	0.9050	0.9275	23.0	+
GA-based Ensemble	GA + aggregation of I3D and MC3	0.9120	0.9225	21.4	+
SA-based Ensemble	SA + aggregation of I3D and MC3	0.9076	0.8908	22.2	+
FA-based Ensemble	FA + aggregation of I3D and MC3	0.8840	0.9209	23.2	+

DA-based Ensemble	DA + aggregation of I3D and MC3	0.9140	0.8919	25.4	+
RCPSO-based Ensemble	RCPSO + aggregation of I3D and MC3	0.9270	0.8700	19.4	+
AGPSO-based Ensemble	AGPSO + aggregation of I3D and MC3	0.9050	0.8563	23.4	+
PSOGSA-based Ensemble	PSOGSA + aggregation of I3D and MC3	0.8930	0.8506	19.8	+
PPO-based Ensemble	PPO + aggregation of I3D and MC3	0.8990	0.8656	18.8	+
DDPG-based Ensemble	DDPG + aggregation of I3D and MC3	0.9120	0.9075	18.4	+
PSORL-based Ensemble	PSORL + aggregation of I3D and MC3	0.9230	0.9088	22.4	+
Default Ensemble Model	Aggregation of I3D and MC3 with default settings	0.8930	0.8600	30.0	+

As indicated in Table 16, the proposed algorithm selects a mean size of 20 base classifiers for ensemble model construction with over 30 runs. Our resulting ensemble models depict a better performance than those established by other search methods, statistically. DA, FA, CS, PSO, and AGPSO produce the largest ensemble models with mean ensemble sizes of 25.4, 23.2, 23, 23.1, and 23.4, respectively. Such large ensemble models cause high computational costs for all four search methods, as well as contradictory decisions for AGPSO. On the other hand, RCPSO, PSOGSA, PPO, and DDPG establish the smallest ensembles with mean ensemble sizes of 19.4, 19.8, 18.8 and 18.4, respectively. Their devised fusion networks show enhanced efficiency, but in exchange of limited complementary characteristics, affecting the aggregation performance. Similar to the proposed model, GA, MPSO, and PSORL extract moderate numbers of base classifiers with mean ensemble sizes of 21.4, 22 and 22.4, respectively, and achieve a reasonable balance between ensemble complexity and weighted ensemble performance. Ensemble networks optimized by most search algorithms possess sufficient diversity and depict a better performance than those from the default ensemble method integrating all 30 base classifiers with default learning settings.

6.3 Evaluation Using the Deepfakes Dataset

The deepfake dataset generated using the deepfake synthetic method is the most challenging subset in FaceForensics++. We employ 1,000 synthetic videos from this deepfake subset together with 1,000 original videos in this experiment. A 60-20-20 split is applied for training, validation and test, respectively. The proposed PSO-optimized I3D and MC3 networks are used in this experiment owing to their competitive performance when compared with those of devised Inceptionv3-RNNs. Specifically, each search method is employed to fine-tune the hyper-parameters of the two 3D CNN models. A set of 30 optimized I3D or MC3 networks is generated, respectively. Besides building weighted ensemble models with the same types of base networks using ensemble scheme 1, dynamic ensembles comprising optimal subsets of I3D and MC3 networks identified by each search method using ensemble scheme 2, are also generated for synthetic video classification. The evaluation details of each optimized 3D CNN, and within- and cross-network ensemble models are presented, as follows.

6.3.1 Evaluation Using the I3D Network

We conduct hyper-parameter search for 10 trials. Each devised 3D CNN is trained using the maximum numbers of 25, 35 and 50 epochs, respectively. The weighted ensemble scheme 1 as defined in Section 5.1 is used to construct ensemble networks with optimized I3Ds as the base classifiers. As depicted in Table 17, the average results of a set of 10 weighted ensembles are used for performance comparison. The statistical test is also performed to compare accuracy result distributions of our optimized ensemble I3D networks against those formulated by the baseline methods.

Table 17 Mean results of the weighted ensemble I3D networks using Deepfakes

Methods	Ensemble topologies	Mean Accuracy rates	Mean AUC	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregated I3D model	0.9825	0.9825	n/a
MPSO-based Ensemble	MPSO + aggregated I3D model	0.9550	0.9550	+
EPSO-based Ensemble	EPSO + aggregated I3D model	0.9725	0.9725	+
PSO-based Ensemble	PSO + aggregated I3D model	0.9600	0.9600	+
CS-based Ensemble	CS + aggregated I3D model	0.9675	0.9675	+
GA-based Ensemble	GA + aggregated I3D model	0.9633	0.9633	+
SA-based Ensemble	SA + aggregated I3D model	0.9650	0.9650	+
FA-based Ensemble	FA + aggregated I3D model	0.9438	0.9438	+
DA-based Ensemble	DA + aggregated I3D model	0.9375	0.9375	+
RCPSO-based Ensemble	RCPSO + aggregated I3D model	0.9325	0.9325	+
AGPSO-based Ensemble	AGPSO + aggregated I3D model	0.9300	0.9300	+
PSOGSA-based Ensemble	PSOGSA + aggregated I3D model	0.9588	0.9588	+
PPO-based Ensemble	PPO + aggregated I3D model	0.9575	0.9575	+
DDPG-based Ensemble	DDPG + aggregated I3D model	0.9317	0.9317	+
PSORL-based Ensemble	PSORL + aggregated I3D model	0.9713	0.9713	+
Default Ensemble Model	Aggregation of I3D with default settings	0.9250	0.9250	+

From Table 17, our weighted ensemble I3D networks outperform counterparts fine-tuned by all baseline search methods statistically as indicated by the statistical test results. EPSO, PSORL, CS, SA and GA-based ensemble models show competitive performance than those of the ensemble models formulated by other baseline search algorithms. In particular, the weighted ensemble models developed by the hybrid algorithm, i.e. PSORL, achieve comparatively more robust performances than those of the ensemble networks yielded by PPO and DDPG algorithms. Among the evolutionary algorithms, AGPSO, RCPSO, DA and FA-based ensemble models show the least efficiency. All weighted ensemble networks established by all search algorithms outperform the corresponding ensemble model with default settings.

Table 18 Optimized learning configurations of the I3D networks generated by each optimizer using Deepfakes

	Learning rate	Learning rate drop factor	Weight decay
Prop. PSO	0.003061	0.05556	0.000806
MPSO	0.006636	0.06667	0.000141
EPSO	0.002045	0.10000	0.000671
PSO	0.005989	0.04762	0.000655
CS	0.004755	0.06612	0.000595
GA	0.001538	0.04158	0.000667
SA	0.004934	0.02934	0.000442
FA	0.007126	0.02350	0.000896
DA	0.007199	0.02874	0.000545
RCPSO	0.007749	0.02632	0.000762
AGPSO	0.009483	0.07321	0.000310
PSOGSA	0.006179	0.06573	0.000596
PPO	0.006406	0.02925	0.000290
DDPG	0.009447	0.02997	0.000108
PSORL	0.004360	0.04081	0.000126
I3D (default)	0.001	0.1	0.0001

The hyper-parameters identified by each optimizer are shown in Table 18. The correlation of moderate learning parameters settings with reliable network performance can be observed in Table 18. Such settings are often retrieved by the proposed PSO model. Similar to the proposed model, EPSO, CS, SA and PSORL identify moderate mean learning rate settings, which show great efficiency in refining network learning weights while tackling stagnations. On the contrary, MPSO, FA, DA, RCPSO, AGPSO, PSOGSA, PPO and DDPG yield comparatively larger learning rates, therefore their resulting networks are more likely to overlook the global optima owing to the instability of gradient descent-based learning updates as well as the employment of comparatively larger learning steps. GA obtains comparatively much smaller learning rates, and its resulting ensemble networks demonstrate less competent performance than some of other devised networks within the pre-defined number of training epochs. A similar observation is also applied to the default ensemble model with comparatively smaller learning rates. Its performance is further affected by its limited base model flexibility.

6.3.2 Evaluation Using the MC3 Network

The weighted ensemble MC3 networks based on ensemble scheme 1 are also subsequently established, where each ensemble model consists of three MC3 networks with distinctive learning settings. Table 19 depicts the model performance based on the average result of 10 such weighted ensemble models. It can be observed that optimized MC3 networks achieve similar results as compared with those of devised I3D models for most search methods.

Table 19 Mean results of the weighted ensemble MC3 networks using Deepfakes

Methods	Ensemble topologies	Mean Accuracy rates	Mean AUC	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregated MC3 model	0.9850	0.9850	n/a
MPSO-based Ensemble	MPSO + aggregated MC3 model	0.9600	0.9600	+
EPSO-based Ensemble	EPSO + aggregated MC3 model	0.9450	0.9450	+
PSO-based Ensemble	PSO + aggregated MC3 model	0.9575	0.9575	+
CS-based Ensemble	CS + aggregated MC3 model	0.9573	0.9573	+
GA-based Ensemble	GA + aggregated MC3 model	0.9375	0.9375	+
SA-based Ensemble	SA + aggregated MC3 model	0.9675	0.9675	+
FA-based Ensemble	FA + aggregated MC3 model	0.9593	0.9593	+
DA-based Ensemble	DA + aggregated MC3 model	0.9609	0.9609	+
RCPSO-based Ensemble	RCPSO + aggregated MC3 model	0.9392	0.9392	+
AGPSO-based Ensemble	AGPSO + aggregated MC3 model	0.9383	0.9383	+
PSOGSA-based Ensemble	PSOGSA + aggregated MC3 model	0.9690	0.9690	+
PPO-based Ensemble	PPO + aggregated MC3 model	0.9581	0.9581	+

DDPG-based Ensemble	DDPG + aggregated MC3 model	0.9400	0.9400	+
PSORL-based Ensemble	PSORL + aggregated MC3 model	0.9425	0.9425	+
Default Ensemble Model	Aggregation of MC3 with default settings	0.9350	0.9350	+

As shown in Table 19, the proposed PSO-based weighted ensemble models show a statistically better performance than those of the ensembles formulated by all baseline algorithms. The weighted fusion networks with learning settings extracted by PSOGSA, SA, DA and MPSO achieve better results than those of ensembles optimized by other baseline search algorithms. GA, AGPSO, and RCPSO-based ensemble networks obtain the lowest performance. The optimized base networks generated by all search methods depict adaptive learning behaviours which outperform those with default learning configurations in most test scenarios.

Table 20 Optimized learning configurations of the MC3 networks generated by each optimizer using Deepfakes

	Learning rate	Learning rate drop factor	Weight decay
Prop. PSO	0.003912	0.04348	0.000119
MPSO	0.002670	0.05000	0.000171
EPSO	0.006409	0.05556	0.000551
PSO	0.004942	0.07143	0.001061
CS	0.007193	0.01963	0.000809
GA	0.001141	0.03319	0.000445
SA	0.002656	0.01665	0.000450
FA	0.005844	0.03577	0.000418
DA	0.005784	0.03141	0.000606
RCPSO	0.008718	0.06086	0.000690
AGPSO	0.008909	0.03087	0.000660
PSOGSA	0.002193	0.06527	0.000793
PPO	0.004895	0.04861	0.000808
DDPG	0.008782	0.06208	0.000218
PSORL	0.001260	0.06872	0.000632
MC3 (default)	0.001	0.1	0.0001

Table 20 reveals the identified learning configurations for each search method. The performance of devised MC3 networks is largely dominated by the optimized learning rates. As an example, AGPSO, RCPSO, DDPG, and EPSO generate larger learning rates, while GA, PSORL, and the default MC3 model use smaller learning rates. The former leads to large alterations to the networks in each learning step, while the latter results in an insufficient learning pace susceptible to under-fitting. Moderate learning rate configurations are produced by the proposed model, PSOGSA, SA, MPSO, DA and FA, which adopt reasonable learning paces for gradient update.

6.3.3. Ensemble Model Construction Using Optimization Algorithms

To take advantage of optimized I3D and MC3 networks, evolving ensemble models with optimal subsets of different types of base networks are constructed using ensemble scheme 2 as explained in Section 5.2. Specifically, each search algorithm is used for ensemble model construction using optimized I3D and MC3 networks as the base classifiers. For each 3D CNN, we generate a set of 10 optimized settings using each search algorithm as discussed earlier. For each optimized setting, we train the respective network with three (i.e. 25, 35 and 50) epochs, respectively. We select the top 15 optimized I3D and top 15 devised MC3 networks for the validation sets to construct the base classifier pool.

Each search algorithm is then used to extract an optimal subset of base classifiers among all the 30 base methods for ensemble network construction. Again, each particle has a dimension of 30 with each element representing a base classifier. A dimension of 30 is assigned to each particle with each dimension representing the selection or omission of a specific base classifier. The optimal subset base network selection is guided by the search mechanisms of each search method. The optimization process is performed with a maximum number of function evaluations of 1,000, and is repeated for 30 trials. For the fitness evaluation, we first compare the value of each dimension in each particle against a threshold value to determine the selection or elimination of the respective base classifier. After obtaining the selected subset of networks, the weighted ensemble strategy shown in Equations (23)-(25) is deployed to fuse the results of all the selected base classifiers. The resulting validation accuracy rate along with the number of selected base networks is employed for fitness calculation as shown in Equation (26). After the completion of the evolutionary process, the swarm leader representing the most optimal subset of the selected base classifiers is derived. This subset of the selected most optimal base networks is leveraged to form the final optimized ensemble, which is used to evaluate unseen samples in the test set. The weighted ensemble results calculated using Equations (23)-(25) are utilized for performance comparison.

Table 21 depicts the mean results of the resulting optimized ensemble networks over a set of 30 runs. By assembling an optimized subset of I3D and MC3 networks with distinctive learning configurations, ensemble models formulated by each search method improve the performance as compared with those with the same types of base networks. The statistical results positively showcase the significance in performance of the optimized ensemble models identified by the proposed optimizer over those generated by other search algorithms.

Table 21 Mean results for optimal ensemble networks integrating I3D and MC3 devised by each search method for Deepfakes

Methods	Ensemble topologies	Mean Accuracy rate	Mean AUC	Ensemble size	Rank sum test results
Proposed PSO-based Ensemble	New PSO + aggregation of I3D and MC3	0.9988	0.9988	16.60	n/a
MPSO-based Ensemble	MPSO + aggregation of I3D and MC3	0.9723	0.9723	18.60	+
EPSO-based Ensemble	EPSO + aggregation of I3D and MC3	0.9771	0.9771	17.20	+
PSO-based Ensemble	PSO + aggregation of I3D and MC3	0.9675	0.9675	17.09	+
CS-based Ensemble	CS + aggregation of I3D and MC3	0.9725	0.9725	19.60	+
GA-based Ensemble	GA + aggregation of I3D and MC3	0.9654	0.9654	17.20	+
SA-based Ensemble	SA + aggregation of I3D and MC3	0.9737	0.9737	21.50	+
FA-based Ensemble	FA + aggregation of I3D and MC3	0.9616	0.9616	15.60	+
DA-based Ensemble	DA + aggregation of I3D and MC3	0.9623	0.9623	19.80	+
RCPSO-based Ensemble	RCPSO + aggregation of I3D and MC3	0.9438	0.9438	21.10	+
AGPSO-based Ensemble	AGPSO + aggregation of I3D and MC3	0.9413	0.9413	15.00	+
PSOGSA-based Ensemble	PSOGSA + aggregation of I3D and MC3	0.9720	0.9720	18.60	+
PPO-based Ensemble	PPO + aggregation of I3D and MC3	0.9606	0.9606	14.33	+
DDPG-based Ensemble	DDPG + aggregation of I3D and MC3	0.9432	0.9432	16.20	+
PSORL-based Ensemble	PSORL + aggregation of I3D and MC3	0.9733	0.9733	16.90	+
Default Ensemble Model	Aggregation of I3D and MC3 with default settings	0.9408	0.9408	30.00	+

As indicated in Table 21, the proposed algorithm extracts a moderate mean number, i.e. 16.6, of base classifiers over 30 runs with high classification accuracy rates and reasonable computational costs. In comparison with our devised ensemble models, CS and SA-based ensemble networks show competitive performance, but their devised ensemble networks consist of comparatively much larger numbers of base classifiers, i.e. 19.6 and 21.5, respectively, with much costly computational complexity. In addition, PPO, AGPSO, FA, and DDPG identify smallest ensemble networks with mean ensemble sizes of 14.33, 15, 15.6, and 16.2, respectively. Such ensemble networks with very small numbers of selected base classifiers demonstrate limited complementary capabilities with suboptimal performances. On the contrary, RCPSO and DA also establish much larger ensemble networks with mean ensemble sizes of 21.1 and 19.8, respectively. Besides increased complexity, their ensemble networks are highly likely to suffer from redundancy, with lower accuracy rates. Ensemble networks optimized by all search algorithms possess significant robustness and demonstrate better performance than those of the default ensemble method integrating all 30 base classifiers.

6.4 Computational Cost Comparison

A computational cost analysis of the proposed model is conducted, as follows. The computational cost of the key operations in the proposed algorithm is represented in Equation (27).

$$Complexity_{prop} = O\big(maxi_{iteration} \times (time_{gbest_{enhancement}} + n' \times (time_{leadergeneration} + time_{RL_{actionselection}} + Fitness_{complexity}))\big) \quad (27)$$

where $maxi_{iteration}$ indicates the pre-defined maximum number of iterations for all search methods, while n' represents the adjusted smaller swarm size for the proposed model. Note that n' is a proportion of the original swarm size n to ensure that our model conducts the same maximum number of function evaluations as those of other search methods for establishing a fair comparison. In each iteration, we take the following costs into account for analysing the complexity, i.e. the cost for swarm leader enhancement ($time_{gbest_{enhancement}}$), as well as the cost incurred for each particle in the swarm. To be specific, the costs of each particle in each iteration include the time allocated for (1) cross-breed leader generation ($time_{leadergeneration}$), (2) Q-learning based local/global search action selection ($time_{RL_{actionselection}}$), and (3) fitness evaluation ($Fitness_{complexity}$) of its respective offspring individual.

For the swarm leader improvement, we randomly select one root-finding algorithm, i.e. Muller's method or fixed-point iteration with the cost of $O(d)$, where d denotes the dimension of the swarm leader. An additional function evaluation with the cost of $Fitness_{complexity}$ is incurred to evaluate the newly generated offspring leader individual using one of these mathematical operations. We therefore have the following updated complexity formula as shown in Equation (28) with respect to the cost for the swarm leader enhancement.

$$time_{gbest_{enhancement}} = O(d) + Fitness_{complexity} \quad (28)$$

Equation (29) provides a breakdown cost of each fitness function evaluation using optimized deep networks.

$$\begin{aligned} Fitness_{complexity} &= \left(k_{epoch} \times train_{size} + test_{size}\right) \times \left(time_{forward}\right) \\ &= O\left(\left(k_{epoch} \times train_{size} + test_{size}\right) \times \left(N_{frame} \times \sum_{l=1}^{L} a_l \times {s_l}^2 \times g_l \times {v_l}^2\right)\right) \end{aligned} \quad (29)$$

where l is the index of a specific convolutional layer out of L convolutional layers, while a_l, v_l, s_l and g_l denote the number of input channels, output feature map size, filter size and filter number, respectively.

For each deep network, frame-level spatial features from a sequence of video frames are extracted. Owing to the slightly different operations for such feature learning processes for different networks, we employ the typical convolutional operation of $\sum_{l=1}^{L} a_l \times {s_l}^2 \times g_l \times {v_l}^2$ for spatial feature learning. Since we need to perform such feedforward feature learning processes for all the image frames (e.g. 50 frames) extracted from each video, the overall complexity of spatial feature learning becomes $N_{frame} \times \sum_{l=1}^{L} a_l \times {s_l}^2 \times g_l \times {v_l}^2$, where N_{frame} denotes the number of frames extracted from each video. The above spatial feature learning process is performed for both training and test datasets with sample sizes of $train_{size}$ and $test_{size}$, respectively. In particular, in the training stage, this feedforward process is repeated for k_{epoch} number of training epochs. Equation (30) is generated by incorporating the cost details pertaining to the fitness evaluation illustrated in Equation (29).

$$\begin{aligned} time_{gbest_{enhancement}} &= O(d) + Fitness_{complexity} \\ &= O(d) + O\left(\left(k_{epoch} \times train_{size} + test_{size}\right) \times \left(N_{frame} \times \sum_{l=1}^{L} a_l \times {s_l}^2 \times g_l \times {v_l}^2\right)\right) \\ &\approx O\left(\left(k_{epoch} \times train_{size} + test_{size}\right) \times \left(N_{frame} \times \sum_{l=1}^{L} a_l \times {s_l}^2 \times g_l \times {v_l}^2\right)\right) \\ &= Fitness_{complexity} \end{aligned} \quad (30)$$

Since the fitness evaluation using deep networks is comparatively more costly than those from the root-finding method for swarm leader enhancement, we further simplify Equation (30) by omitting the cost of $O(d)$.

Moreover, Equation (31) represents the updated new complexity formula by replacing the swarm leader enhancement cost in Equation (27) with Equation (30).

$$Complexity_{prop} \approx O\big(maxi_{iteration} \times (Fitness_{complexity} + n' \times (time_{leadergeneration} + time_{RL_{actionselection}} + Fitness_{complexity}))\big) \quad (31)$$

As mentioned earlier, for each particle, there are three costs, i.e. the costs for (1) cross-breed leader generation $time_{leadergeneration}$, (2) Q-learning based search action selection $time_{RL_{actionselection}}$, and (3) one fitness function evaluation for its offspring solution $Fitness_{complexity}$. We analyse these cost details pertaining to each particle, as follows.

Firstly, for each particle, a cross-breed leader is generated for its position update, where weighting coefficients of the global and personal best solutions are produced using a set of 3D formulae. This process has a computational complexity of $O(m)$, where m represents the number of 3D points generated using the corresponding 3D formulae. Because of the ranking of these generated values in the z-axis using a Python built-in function, it requires an additional cost of $O(m)$. Therefore, the total cost of the adaptive weighting coefficient generation with respect to cross-breed leader production is $2 \times O(m)$. We simplify the cost representation by omitting the constant factor, as shown in Equation (32).

$$time_{leadergeneration} = O(m) \quad (32)$$

Secondly, each particle employs the Q-learning method to select local and global search operations. This results in one additional function evaluation for measuring the fitness of the new particle position by performing a selected action (i.e. either a selected local or a global search operation). This new fitness result is employed to compare with the previous fitness score of the current particle for immediate reward generation. Therefore, we define $time_{RL_{actionselection}}$ as follows.

$$time_{RL_{actionselection}} = Fitness_{complexity} + O(d) \approx Fitness_{complexity} \quad (33)$$

where $O(d)$ denotes the cost for implementing the selected search action. As discussed earlier, since this position updating operation cost for each particle is comparatively smaller as compared with the time spent for fitness evaluation, it is omitted.

We replace the cost details obtained in Equations (32) and (33) with the corresponding components in Equation (27). Equation (34) shows the updated complexity formula.

$$\begin{aligned} Complexity_{prop} &= O\big(maxi_{iteration} \times (Fitness_{complexity} + n' \times (O(m) + Fitness_{complexity} + Fitness_{complexity}))\big) \\ &\approx O\big(maxi_{iteration} \times (Fitness_{complexity} + n' \times (Fitness_{complexity} + Fitness_{complexity}))\big) \\ &= O\big(maxi_{iteration} \times (2n' + 1) \times Fitness_{complexity}\big) \end{aligned} \quad (34)$$

As indicated earlier, since the fitness evaluation involving deep networks is a far more computationally complex process than the cost of the adaptive weighting coefficient generation, we further simplify Equation (34) by discarding the cost of $O(m)$.

In our experimental studies, all search methods are established to execute the same pre-defined number of function evaluations, in order to ensure their results are comparable. Because of employing additional function evaluations as discussed above during the search process, we adjust the population size of the proposed model to $n' = (n-1)/2$, where n is the original population size for all other search methods without the requirement of additional function evaluations. Such a process for population size adjustment is also applied to other PSO/FA variants if additional function evaluations are incurred. We further update the complexity calculation in Equation (34) to the following formula.

$$Complexity_{prop} = O\big(maxi_{iteration} \times n \times Fitness_{complexity}\big) \quad (35)$$

As depicted in Equation (35), the computational complexity of the proposed model largely relies on the pre-defined maximum number of function evaluations, i.e. $maxi_{iteration} \times n$.

After performing the complexity analysis, we also perform a practical computational cost comparison between the proposed model and all baseline search methods for hyper-parameter search using the three video deepfake datasets. Table 22 depicts the computational-wise comparison for Celeb-DFv2, FaceForensics++, and Deepfakes datasets, respectively.

Table 22 Computational cost comparison with respect to hyper-parameter search using video deepfake datasets (in seconds)

Methods	Celeb-DFv2	FaceForensics++	Deepfakes
Prop. PSO	60.5368	111.1530	51.4215
MPSO	60.1449	108.7653	49.9884
EPSO	63.0685	114.4960	54.5360
PSO	59.1618	108.7589	49.0206
CS	59.2949	108.8343	48.4897
GA	59.9933	109.4659	50.3709
SA	59.6803	108.8736	49.5974
FA	59.4326	108.4193	49.0232
DA	59.1697	107.7223	49.8914
RCPSO	60.6861	112.5235	51.8656
AGPSO	60.2443	109.6587	50.2418
PSOGSA	60.3211	110.1778	50.9331
PPO	81.2031	125.1420	70.0105
DDPG	78.2640	122.2030	67.0715
PSORL	68.2307	119.9914	56.7037

As discussed earlier, a pre-defined maximum number of function evaluations is utilized as the termination criterion for all the search methods with respect to a specific dataset. Since such a fitness evaluation procedure embedding deep networks is the most computationally expensive process in comparison with other search related operations, the time spent for hyper-parameter search for each optimization algorithm is comparatively identical. In order to indicate the operational variations of different search algorithms, we generate the mean computational cost in seconds with respect to one fitness evaluation in conjunction with the time spent in traversing through the key search operations of each algorithm over a set of 30 runs. The cost variations between different algorithms indicate the operational differences in execution of these methods. The cost details in Table 22 are obtained using a NVIDIA RTX 3090 GPU.

As showcased in Table 22, our algorithm has obtained a better balance between performance and computational cost with moderate mean computational costs over 30 runs across all three video datasets. In addition, owing to the employment of neural network-based cumulative reward score generation and hyper-parameter prediction, DDPG and PPO illustrate larger computational costs as compared with those of the proposed model and other swarm-based methods. PSORL also shows comparatively higher computational costs because of the construction of a larger Q-table for the dispatching of several root-finding algorithms for top-ranking particle enhancement. EPSO is also computationally more costly owing to the integration of multiple search actions fine-tuned by nonlinear geometric function-based search parameters and DE-based swarm leader enhancement. RCPSO also requires slightly larger computational costs than those of the proposed model, because of multiple distinctive subswarm-based search actions with sine/cosine-based search parameters.

The remaining PSO variants (PSOGSA, AGPSO and MPSO) and classical search algorithms show lower costs than those of the proposed optimizer. For instance, PSOGSA illustrates longer processing time than those of AGPSO and MPSO, due to the integration of GSA with PSO. The smallest costs are obtained by CS, FA, PSO, DA, and SA, due to the simplicity of these classical search algorithms. In short, the proposed model is equipped with not only effective search strategies but also reasonable computational efficiency as compared with those of all the swarm-based and reinforcement learning algorithms, as evidenced in cost comparison results.

To visualize the effects of feature learning capabilities of the optimized 3D CNNs, the gradient-weighted class activation mapping (Grad-CAM) algorithm [87] is utilized to generate heatmaps. Such heatmaps use different colour schemes to illustrate which image pixels/regions contribute the most to the target class prediction. To be specific, we firstly calculate gradient descent back-propagated to the final convolutional layers with respect to the target fake/real classes. These gradient descent results are subsequently summed and averaged to produce the dominating weights for the corresponding feature maps. We then multiply these weights with the respective feature maps to generate the Grad-CAM heatmaps. These heatmaps are used to indicate which convolutional features are comparatively more significant to the fake or real class prediction. Figure 10 shows examples of real and manipulated video frames taken from the Celeb-DFv2 dataset, along with the Grad-CAM heatmaps produced using the I3D and MC3 networks for the manipulated samples with optimal hyper-parameters yielded by the proposed model.

As discussed earlier, since face swap has been performed on this Celeb-DFv2 dataset, a facial cropping algorithm, i.e. MTCNN [81], is employed in the pre-processing stage to extract facial regions automatically. This also allows us to better present the significance of different facial attributes to deepfake detection. The colours embedded in the Grad-CAM heatmaps range from bright red to dark blue, indicating the most significant (bright red) to the most irrelevant (dark blue) facial characteristics to video authenticity classification.

As indicated in existing studies [1, 2], manipulated facial image frames can include a variety of properties, such as blurred mouth, nose and eye regions, unsmoothed textures of mouth/nose and facial borders, asymmetric eye pupils and facial hair, misaligned jawlines, and unnatural facial lighting and expressions. As indicated in Figure 10, our optimized I3D and MC3 models are able to extract such important synthetic facial features highlighted by the red colour in the generated heatmaps pertaining to the fake images to inform deepfake detection. For instance, the blurred eye and mouth regions, misaligned teeth, unsmoothed facial borders, and unnatural high contrast lighting around eye pupils have been regarded as the most dominating features by both of our optimized 3D CNNs with respect to manipulated video frame classification.

Since a set of 50 image frames is randomly extracted from an input video, a sequence of such highly discriminative feature maps is used for the identification of a synthetic and real video input by each optimized network. These heatmaps extracted by our optimized networks further ascertain the network effectiveness in spatial-temporal feature learning, leading to high classification accuracy for various video forgery detection, as indicated in our experimental studies.

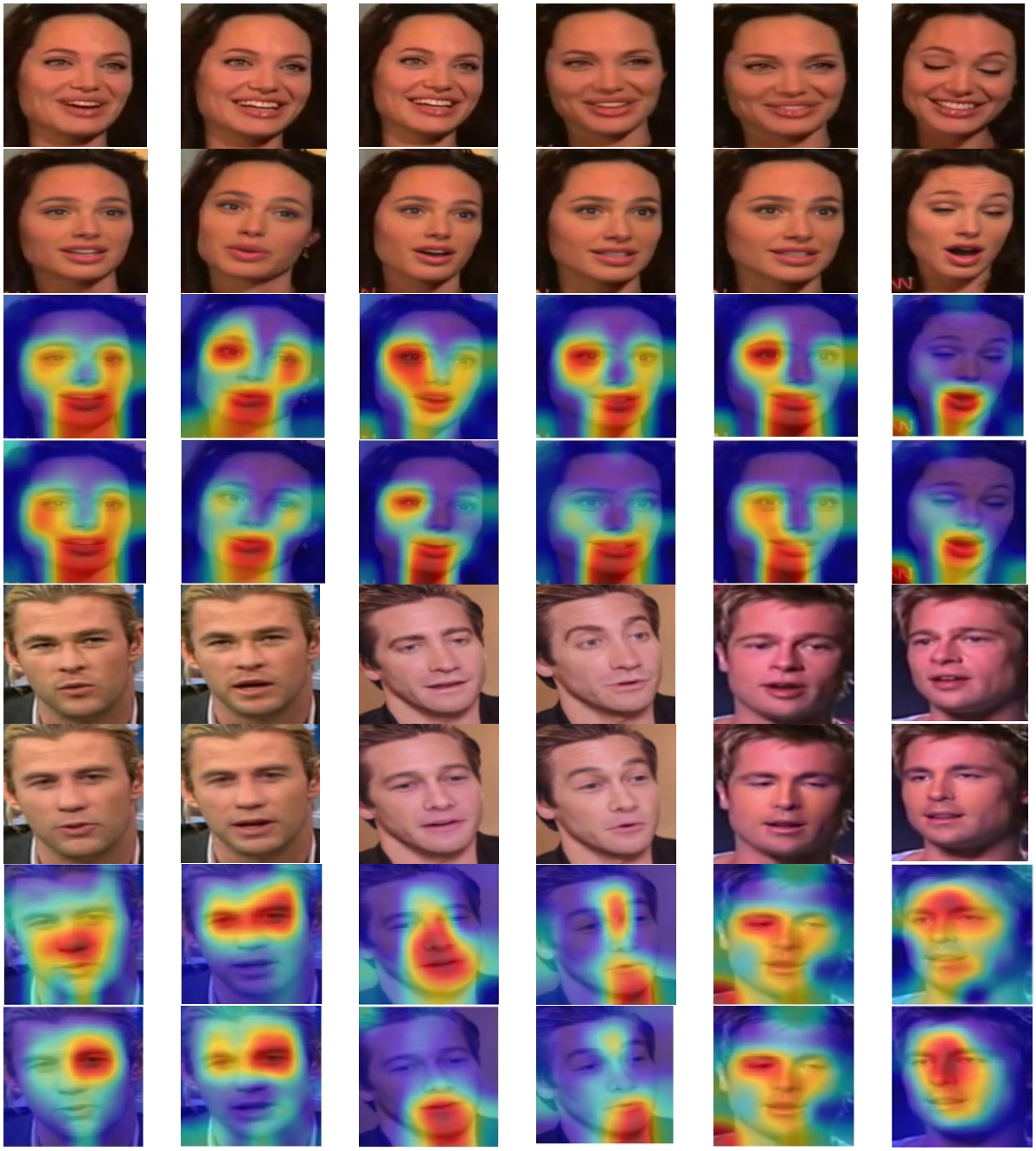

Figure 10 Example video frames taken from Celeb-DFv2 along with extracted Grad-CAM heatmaps for the manipulate frames (For each set of images, the first and second rows illustrate the original and synthetic images, with the third and fourth rows showing the heapmaps generated by our optimized I3D and MC3, respectively.)

A comparison with existing state-of-the-art related studies is presented in Tables 23-25 for Celeb-DFv2, FaceForensics++ and Deepfakes, respectively. Some related studies employed the same official train-test split for Celeb-DFv2, or the same experimental settings for Deepfakes and FaceForensics++ as those in this research, while other studies were trained using combined or other significantly larger datasets. As indicated in Tables 23-25, our model depicts a competitive performance for all the three test datasets owing to the adoption of diverse networks, e.g. CNN-RNN, I3D and MC3, with distinctive spatial-temporal feature extraction and learning mechanisms. These unique learning behaviours are strengthened by applying the proposed PSO-based network structure and hyper-parameter fine-tuning method. The evolving ensemble generation with the identification of optimal subsets of different types of optimized networks is able to even further boost the performance of individual classifiers. In comparison with our approach, most existing studies utilize homogenous single 2D or 3D networks without taking advantage of diverse distinctive learning mechanisms of different networks and

evolutionary algorithm-based hyper-parameter optimization and dynamic ensemble construction mechanisms, therefore constraining their model performance.

Table 23 Related work comparison for Celeb-DFv2

Related works	Strategies	Accuracy	AUC
Zhang et al. [15]	Temporal Dropout 3D CNN	0.8108	0.8883
Ciftci et al. [7]	FakeCatcher	0.9150	-
Pu et al. [21]	CNN-LSTM	0.874±0.23	85.5±0.35
Pu et al. [21]	CNN-GRU	0.923±0.17	89.9±0.37
Demir and Ciftci [88]	Gaze tracking	0.8835	-
Afchar et al. [89]	Meso4	0.720±0.99	83.0±1.65
Afchar et al. [89]	MesoInception4	0.853±1.53	89.7±2.11
Nguyen et al. [90]	Capsule network	0.91±0.35	88.5±0.26
Rossler et al. [2]	XN-max	0.8989	-
Wang et al. [5]	LiSiam	-	0.7821
Wang et al. [91]	FakeSpotter	-	0.668
Liu et al. [92]	SPSL	-	0.7688
Zhao et al. [93]	Multi-attention	-	0.6744
Yang et al. [94]	MTD-Net	-	0.7012
Wang et al. [23]	MC-LCR	-	0.7161
Zhou et al. [95]	A dual neural network	-	0.7341
Li et al. [1]	Xception-c23	-	0.653
Li et al. [1]	Xception-c40	-	0.655
Li and Lyu [96]	D-FWA	-	0.646
Ours	The proposed PSO-based evolving ensemble model combining Inceptionv3-RNN, I3D and MC3	0.9498	0.9270

Table 24 Related work comparison for FaceForensics++

Related works	Strategies	Accuracy	AUC
Zhang et al. [15]	Temporal Dropout 3D CNN (60:20:20)	0.7909	0.7222
Ciftci et al. [7]	FakeCatcher (60:40)	0.9465	-
Fridrich and Kodovsky [97]	Steganalysis features + SVM	0.7097	-
Cozzolino et al. [98]	LD-CNN (72:14:14)	0.7845	-
Bayar and Stamm [99]	Constrained Conv (72:14:14)	0.8297	-
Rahmouni et al. [100]	CustomPooling CNN (72:14:14)	0.7908	-
Afchar et al. [89]	MesoNet (72:14:14)	0.8310	-
Gunawan et al. [101]	Xception-ELA (72:14:14)	0.9386	-
Demir and Ciftci [88]	Gaze tracking	0.9248	-
Zhang et al. [102]	Face-Alignment (FA)-LSTM	0.825	-
Zhang et al. [102]	Dense Face-Alignment (DFA)-LSTM	0.924	-
Kim et al. [103]	CNN-Eye	0.791	-
Li et al. [104]	LRCN	0.836	-
Güera and Delp [105]	ConvLSTM	0.8784	-
Sohrawardi et al. [106]	FaceNetLSTM	0.8957	-
Nguyen et al. [25]	ClassNSeg	0.7976	-
Sabir et al. [107]	DenseNetAligned	0.9053	-
Li et al. [108]	Face X-ray	-	0.874
Li and Lyu [96]	D-FWA	-	0.575
Ours	The proposed PSO-based evolving ensemble model combining I3D and MC3 (60:20:20)	0.9620	0.9425

Table 25 Related work comparison for Deepfakes

Related works	Strategies	Accuracy	AUC
Li and Lyu [96]	D-FWA	0.512	0.514
Nguyen et al. [90]	Capsule network	0.846±0.12	0.847±0.13
Ciftci et al. [7]	FakeCatcher	0.9375	-
Afchar et al. [89]	Meso4	0.704±1.02	0.776±0.85
Afchar et al. [89]	MesoInception4	0.823±1.32	0.839±0.93
Pu et al. [21]	ResNet50-GRU	0.948±0.25	0.984±0.23
Pu et al. [21]	CNN-LSTM	0.892±0.23	0.887±0.31
Pu et al. [21]	CNN-GRU	0.912±0.65	0.899±0.27
Rossler et al. [2]	Xception	0.835±0.75	0.899±0.53
Shang et al. [22]	Pixel-Region Relation Network	0.8470	-
Xu and Yayilgan [109]	Xception	0.9877	-

Xu and Yayilgan [109]	Resnet152+LSTM	0.8853	-
Li et al. [110]	Sharp multi-instance learning	0.9071	-
Li et al. [110]	LSTM	0.8500	-
Carreira and Zisserman [10]	3D CNN	0.93	-
Güera and Delp [105]	ConvLSTM	0.9429	-
Sohrawardi et al. [106]	FaceNetLSTM	0.9571	-
Nguyen et al. [25]	ClassNSeg	0.6429	-
Sabir et al. [107]	DenseNetAligned	0.9643	-
Zi et al. [111]	ADDNet-2D	0.9011	-
Ours	The proposed PSO-based evolving ensemble model combining I3D and MC3	0.9988	0.9988

6.5 Evaluation Using Numerical Optimization Problems

Besides the evaluation using deepfake video classification problems, we exploit unimodal (Sum of Different Powers, Dixon-Price, Sum Squares, Rotated Hyper-Ellipsoid, Sphere, Zakharov and Rosenbrock) and multimodal (Ackley, Griewank, Rastrigin, and Powell) numerical functions, to further assess model efficiency in optimization. These benchmark functions possess complex artificial geometric contours with single or many local optima. They have been widely used to evaluate the diversification and intensification search capabilities of swarm intelligence algorithms in dynamic search space. Besides the previous 12 selected classical and advanced search methods, a total of 12 additional baseline methods are employed for performance comparison, namely a classical research method (i.e. Moth-Flame Optimization (MFO) [112]), PSO variants (i.e. PSO with cosine search coefficients (CPSO) [43], Genetic PSO (GPSO) [113], PSO with preserved neighbourhood learning (DNLPSO) [114], Genetic and Mutated PSO (GMPSO) [65], and PSO with opposition and DE-based leader mutation (ELPSO) [64]), and FA variants (i.e. Repulsive FA (RFA) [115], FA with Logistic map-oriented search coefficients (LFA) [116], FA with Gauss map-based search parameters (GFA) [117], FA with an adaptive randomization factor (VSSFA) [118], an FA variant (FAV) [119], and Neighbourhood Attraction FA (NaFA) [120]). These algorithms are selected owing to their competitive performance in tackling diverse mathematical landscapes in optimization.

In this study, a total of 25,000 function evaluations, a dimension of 30 and 30 trials are employed to test model convergence and accuracy rates in attaining global minima. We provide the average, best (minimum), worst (maximum), standard deviation and statistical test results of each search algorithm for all the numerical optimization functions in Table 26. As evidenced by results in Table 26, the proposed model is able to obtain the minimum mean results for solving nearly all the benchmark functions as compared with those of 24 baseline methods. In particular, it achieves the global optima (i.e. '0') for 5 unimodal (Sum of Different Powers, Zakharov, Sum Squares, Sphere and Rotated Hyper-Ellipsoid) and 3 multimodal (Griewank, Rastrigin, and Powell) mathematical landscapes. PSORL obtains the global minimum solution of '0' for two benchmark functions, i.e. Griewank and Sum of Different Powers. In addition, PSOGSA, GMPSO, MPSO, CPSO, NaFA and RFA show comparatively better results than those from other baseline methods.

A further Wilcoxon rank sum test is conducted, where '+', '-' and '=' are used to indicate if our results are statistically better, worse, or the same as compared with those from the baseline methods. As shown in Table 26, the proposed model shows significance in performance difference against 24 baseline methods in most evaluation scenarios. The rare exceptional cases are, for Rosenbrock, CPSO, MPSO, PSOGSA, NaFA, FA and PSORL obtain similar results to those of the proposed model, while for Dixon-Price, PSORL obtains statistically better results than those from our algorithm with PSOGSA achieving similar result distributions to those from the proposed model.

Table 26 Evaluation and statistical results of unimodal and multimodal artificial landscapes over 30 runs

		Prop. PSO	**EPSO**	**GPSO**	**AGPSO**	**DNLPSO**	**ELPSO**	**GMPSO**	**CPSO**	**MPSO**	**PSOGSA**
Ackley	mean	**4.44E-16**	1.70E+01	1.76E+01	2.69E+00	2.73E+00	1.50E+01	3.12E+00	1.21E+01	1.68E+01	1.25E+01
	min	**4.44E-16**	1.44E+01	1.60E+01	1.16E+00	9.01E-02	1.34E+01	1.50E+00	1.07E+01	1.44E+01	3.93E+00
	max	**4.44E-16**	1.88E+01	1.83E+01	9.04E+00	5.47E+00	1.60E+01	5.68E+00	1.34E+01	1.81E+01	1.82E+01
	std	**0.00E+00**	9.92E-01	5.17E-01	1.38E+00	1.55E+00	5.94E-01	1.27E+00	6.49E-01	8.47E-01	3.94E+00
	RS	n/a	+	+	+	+	+	+	+	+	+
Dixon	mean	**6.67E-01**	1.32E+05	5.01E+05	1.57E+01	2.55E+02	1.39E+05	6.95E-01	6.15E+00	1.13E+01	6.71E-01
	min	6.67E-01	1.27E+04	2.45E+05	6.67E-01	6.67E-01	4.21E+04	6.67E-01	6.67E-01	6.67E-01	**6.67E-01**
	max	**6.67E-01**	4.66E+05	8.96E+05	3.21E+02	6.63E+03	2.17E+05	1.27E+00	8.32E+01	3.21E+02	8.02E-01
	std	**2.38E-05**	1.16E+05	1.79E+05	5.97E+01	1.21E+03	4.64E+04	1.15E-01	2.08E+01	5.85E+01	2.46E-02
	RS	n/a	+	+	+	+	+	+	+	+	=
Griewank	mean	**0.00E+00**	1.19E+02	2.88E+02	3.92E-02	7.32E-01	1.44E+02	1.38E-02	2.63E-02	5.66E-02	1.89E+01
	min	**0.00E+00**	1.20E+01	2.05E+02	1.22E-04	8.69E-05	8.87E+01	7.66E-09	6.03E-04	5.56E-04	5.04E-02
	max	**0.00E+00**	3.15E+02	3.80E+02	1.71E-01	2.88E+00	1.99E+02	6.58E-02	1.14E-01	5.36E-01	9.11E+01
	std	**0.00E+00**	7.42E+01	4.22E+01	3.94E-02	8.52E-01	2.31E+01	1.48E-02	2.89E-02	9.98E-02	3.65E+01
	RS	n/a	+	+	+	+	+	+	+	+	+

Rastrigin	mean	**0.00E+00**	2.45E+02	3.51E+02	5.82E+01	9.02E+01	2.72E+02	4.27E+01	6.63E+01	5.09E+01	1.35E+02
	min	**0.00E+00**	1.45E+02	2.71E+02	3.38E+01	3.67E+01	2.12E+02	1.87E+01	3.78E+01	3.18E+01	7.96E+01
	max	**0.00E+00**	3.11E+02	3.87E+02	1.10E+02	2.23E+02	3.12E+02	7.83E+01	1.10E+02	9.75E+01	1.86E+02
	std	**0.00E+00**	4.36E+01	2.52E+01	1.78E+01	4.22E+01	2.53E+01	1.55E+01	1.75E+01	1.54E+01	2.84E+01
	RS	n/a	+	+	+	+	+	+	+	+	+
Rothyp	mean	**0.00E+00**	7.77E+04	1.99E+05	2.82E+02	1.84E+03	8.86E+04	7.28E-06	5.63E+02	9.27E-04	7.90E+03
	min	**0.00E+00**	1.40E+04	1.29E+05	6.83E-05	1.44E-03	4.70E+04	4.58E-08	3.45E-04	7.46E-05	2.42E-01
	max	**0.00E+00**	1.59E+05	2.71E+05	4.23E+03	3.94E+04	1.23E+05	1.41E-04	1.27E+04	2.95E-03	6.76E+04
	std	**0.00E+00**	3.59E+04	3.40E+04	1.07E+03	7.22E+03	1.78E+04	2.55E-05	2.41E+03	7.82E-04	1.86E+04
	RS	n/a	+	+	+	+	+	+	+	+	+
Rosenbrock	mean	**2.83E+01**	1.39E+05	3.71E+05	1.09E+02	1.15E+03	1.10E+05	3.61E+01	2.45E+02	9.91E+01	4.01E+01
	min	2.78E+01	2.08E+04	1.90E+05	3.39E+00	1.78E+01	3.99E+04	1.39E+01	**2.76E+00**	3.94E+00	1.96E+01
	max	**2.87E+01**	4.47E+05	6.46E+05	1.07E+03	1.44E+04	2.11E+05	9.14E+01	2.58E+03	1.01E+03	1.26E+02
	std	**2.24E-01**	9.58E+04	1.11E+05	1.86E+02	3.55E+03	4.16E+04	2.47E+01	6.53E+02	1.92E+02	3.05E+01
	RS	n/a	+	+	+	+	+	+	=	=	=
Sphere	mean	**0.00E+00**	4.21E+01	7.97E+01	1.32E-06	2.16E-01	3.70E+01	3.01E-09	1.74E-03	1.08E-03	7.94E-19
	min	**0.00E+00**	1.11E+01	4.34E+01	4.78E-09	1.17E-09	2.72E+01	6.27E-11	2.59E-04	1.39E-04	4.56E-19
	max	**0.00E+00**	9.15E+01	1.06E+02	1.24E-05	3.53E+00	5.07E+01	2.37E-08	6.35E-03	1.13E-02	1.47E-18
	std	**0.00E+00**	1.94E+01	1.44E+01	2.47E-06	6.62E-01	5.84E+00	5.60E-09	1.26E-03	2.00E-03	2.31E-19
	RS	n/a	+	+	+	+	+	+	+	+	+
Sumpow	mean	**0.00E+00**	2.88E-03	1.49E-01	2.63E-17	1.76E-06	1.65E-02	1.43E-27	5.86E-07	3.90E-07	2.02E-09
	min	**0.00E+00**	3.91E-06	4.22E-03	1.03E-23	2.28E-37	5.58E-05	6.75E-34	3.83E-09	4.55E-10	1.34E-10
	max	**0.00E+00**	2.45E-02	7.66E-01	4.09E-16	4.24E-05	4.03E-02	1.90E-26	5.85E-06	7.46E-06	8.85E-09
	std	**0.00E+00**	5.67E-03	1.77E-01	8.53E-17	7.81E-06	1.13E-02	4.10E-27	1.29E-06	1.36E-06	2.17E-09
	RS	n/a	+	+	+	+	+	+	+	+	+
Zakharov	mean	**0.00E+00**	3.75E+02	4.75E+02	7.56E+01	1.09E+02	3.46E+02	3.18E+01	2.98E+02	4.13E+02	2.21E+02
	min	**0.00E+00**	2.57E+02	4.06E+02	3.69E+01	5.12E+01	3.03E+02	1.14E+01	2.41E+02	3.13E+02	1.24E+02
	max	**0.00E+00**	5.62E+02	5.29E+02	1.14E+02	2.11E+02	3.93E+02	5.04E+01	3.34E+02	6.19E+02	3.11E+02
	std	**0.00E+00**	8.15E+01	2.84E+01	1.76E+01	5.10E+01	2.49E+01	1.04E+01	2.11E+01	6.30E+01	5.54E+01
	RS	n/a	+	+	+	+	+	+	+	+	+
Sumsqu	mean	**0.00E+00**	5.41E+02	1.16E+03	2.84E-05	8.44E+00	5.79E+02	6.61E-08	1.75E+00	1.02E-03	2.30E-17
	min	**0.00E+00**	1.29E+02	7.29E+02	1.86E-07	7.16E-10	3.64E+02	3.29E-10	4.53E-04	9.84E-05	5.96E-18
	max	**0.00E+00**	1.38E+03	1.55E+03	2.63E-04	1.73E+02	7.18E+02	4.45E-07	2.62E+01	3.07E-03	4.78E-17
	std	**0.00E+00**	2.72E+02	2.22E+02	5.51E-05	3.16E+01	9.41E+01	1.13E-07	6.65E+00	7.16E-04	9.72E-18
	RS	n/a	+	+	+	+	+	+	+	+	+
Powell	mean	**0.00E+00**	1.63E+03	4.17E+03	1.91E+01	9.64E+00	1.78E+03	6.43E-02	3.59E+02	4.61E+03	1.00E-01
	min	**0.00E+00**	1.46E+02	1.27E+03	2.48E-03	6.23E-04	5.47E+02	7.05E-04	1.77E+02	5.25E+02	4.47E-03
	max	**0.00E+00**	5.47E+03	6.69E+03	1.04E+02	1.19E+02	2.51E+03	9.02E-01	5.66E+02	1.77E+04	4.41E-01
	std	**0.00E+00**	1.47E+03	1.31E+03	3.58E+01	2.25E+01	4.71E+02	1.68E-01	8.80E+01	3.99E+03	1.04E-01
	RS	n/a	+	+	+	+	+	+	+	+	+

		Prop. PSO	**RFA**	**LFA**	**GFA**	**VSSFA**	**FAV**	**NaFA**	**PSO**	**FA**	**DA**	**CS**	**MFO**
Ackley	mean	**4.44E-16**	7.31E-03	1.57E+01	1.46E+01	1.04E+01	2.02E+01	8.34E-03	1.50E+01	4.43E-02	7.30E+00	3.76E+00	1.14E+01
	min	**4.44E-16**	6.09E-03	1.46E+01	1.33E+01	9.38E+00	2.02E+01	6.23E-03	1.39E+01	2.68E-02	1.72E+00	2.67E+00	4.31E-01
	max	**4.44E-16**	7.97E-03	1.65E+01	1.53E+01	1.10E+01	2.02E+01	1.07E-02	1.60E+01	8.44E-02	1.15E+01	4.89E+00	1.67E+01
	std	**0.00E+00**	4.92E-04	4.46E-01	4.77E-01	4.37E-01	4.12E-15	1.25E-03	5.40E-01	1.32E-02	2.31E+00	5.93E-01	4.66E+00
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+
Dixon	mean	**6.67E-01**	7.44E-01	1.48E+05	1.30E+05	1.08E+04	1.62E+06	1.48E+00	1.19E+00	3.96E+00	1.15E+03	9.67E+00	3.90E+04
	min	**6.67E-01**	6.90E-01	6.35E+04	8.78E+04	7.20E+03	1.62E+06	6.68E-01	6.76E-01	7.26E-01	1.96E+01	5.16E+00	3.03E+00
	max	**6.67E-01**	8.28E-01	2.26E+05	1.76E+05	1.80E+04	1.62E+06	1.18E+01	4.59E+00	2.09E+01	7.85E+03	1.61E+01	5.42E+05
	std	2.38E-05	3.93E-02	4.09E+04	2.32E+04	2.89E+03	**1.62E-10**	2.14E+00	9.84E-01	5.22E+00	1.78E+03	2.98E+00	1.15E+05
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+
Griewank	mean	**0.00E+00**	1.92E-03	1.66E+02	1.55E+02	4.46E+01	6.08E+02	3.74E-03	3.08E-01	5.27E-03	1.00E+01	1.14E+00	1.28E+01
	min	**0.00E+00**	1.23E-03	1.13E+02	1.10E+02	3.06E+01	6.08E+02	2.23E-03	1.14E-02	2.95E-03	2.07E+00	1.08E+00	2.73E-01
	max	**0.00E+00**	2.73E-03	2.05E+02	1.86E+02	5.17E+01	6.08E+02	6.00E-03	1.08E+00	8.06E-03	2.47E+01	1.22E+00	1.81E+02
	std	**0.00E+00**	3.37E-04	2.46E+01	1.84E+01	4.98E+00	2.36E-13	1.16E-03	3.64E-01	1.26E-03	6.38E+00	3.55E-02	3.91E+01
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+
Rastrigin	mean	**0.00E+00**	9.14E-04	2.75E+02	2.67E+02	2.09E+02	4.29E+02	3.01E+01	5.27E+01	2.48E+01	1.24E+02	1.09E+02	1.45E+02
	min	**0.00E+00**	6.08E-04	2.34E+02	2.47E+02	1.70E+02	4.29E+02	1.49E+01	3.29E+01	1.56E+01	1.00E+00	7.63E+01	7.87E+01
	max	**0.00E+00**	1.19E-03	2.95E+02	2.84E+02	2.27E+02	4.29E+02	8.06E+01	8.76E+01	3.51E+01	2.51E+02	1.40E+02	2.16E+02
	std	**0.00E+00**	1.40E-04	1.41E+01	1.07E+01	1.36E+01	1.13E-13	1.22E+01	1.33E+01	5.66E+00	5.83E+01	1.52E+01	3.70E+01
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+
Rothyp	mean	**0.00E+00**	1.03E-02	1.10E+05	1.02E+05	2.94E+04	4.38E+05	3.72E+01	5.58E-01	8.88E+00	4.96E+03	7.88E+01	1.27E+04
	min	**0.00E+00**	7.27E-03	7.32E+04	7.40E+04	1.97E+04	4.38E+05	1.29E+00	1.13E-02	2.61E-01	9.00E+02	4.01E+01	3.42E+00
	max	**0.00E+00**	1.25E-02	1.38E+05	1.20E+05	3.59E+04	4.38E+05	1.75E+02	4.84E+00	3.43E+01	1.76E+04	1.36E+02	6.76E+04
	std	**0.00E+00**	1.44E-03	1.69E+04	1.07E+04	3.44E+03	2.01E-10	4.42E+01	1.28E+00	8.96E+00	3.78E+03	2.21E+01	1.68E+04
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+
Rosenbrock	mean	**2.83E+01**	2.86E+01	1.09E+05	7.97E+04	8.01E+03	2.84E+06	4.13E+01	8.84E+01	4.18E+01	2.14E+03	1.66E+02	6.69E+04
	min	2.78E+01	2.85E+01	6.52E+04	5.30E+04	4.22E+03	2.84E+06	2.48E+01	**1.55E+00**	2.69E+01	2.07E+02	9.36E+01	3.26E+01
	max	**2.87E+01**	2.87E+01	1.76E+05	1.03E+05	1.12E+04	2.84E+06	1.06E+02	1.06E+03	1.25E+02	1.29E+04	2.25E+02	2.23E+05
	std	2.24E-01	4.92E-02	2.27E+04	1.41E+04	1.71E+03	**4.74E-10**	2.53E+01	1.87E+02	2.95E+01	2.62E+03	4.06E+01	6.86E+04
	RS	n/a	+	+	+	+	+	=	+	=	+	+	+
Sphere	mean	**0.00E+00**	4.80E-06	5.03E+01	4.44E+01	1.28E+01	1.77E+02	6.00E-06	3.52E-02	1.45E-03	2.61E+00	3.85E-02	5.25E+00
	min	**0.00E+00**	3.38E-06	3.52E+01	3.38E+01	8.86E+00	1.77E+02	3.30E-06	1.26E-02	3.30E-04	2.75E-01	1.76E-02	3.99E-04
	max	**0.00E+00**	6.19E-06	5.73E+01	5.08E+01	1.54E+01	1.77E+02	1.04E-05	8.54E-02	4.27E-03	1.25E+01	6.85E-02	2.62E+01
	std	**0.00E+00**	6.14E-07	4.72E+00	3.94E+00	1.41E+00	2.64E-14	1.69E-06	1.89E-02	1.05E-03	2.54E+00	1.32E-02	1.07E+01
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+
Sumpow	mean	**0.00E+00**	1.19E-13	8.95E-03	7.55E-03	2.07E-04	5.82E-01	8.39E-08	1.93E-05	3.58E-07	2.44E-05	4.81E-11	2.33E-10
	min	**0.00E+00**	2.08E-15	2.18E-03	1.17E-03	8.50E-05	5.82E-01	6.52E-09	4.22E-07	4.80E-08	2.07E-28	1.64E-12	6.55E-15
	max	**0.00E+00**	7.19E-13	2.29E-02	1.83E-02	3.60E-04	5.82E-01	2.31E-07	1.01E-04	1.68E-06	3.07E-04	2.77E-10	3.89E-09
	std	**0.00E+00**	1.46E-13	4.53E-03	4.06E-03	8.39E-05	1.47E-16	5.88E-08	2.16E-05	3.79E-07	6.95E-05	5.56E-11	7.18E-10
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+
Zakharov	mean	**0.00E+00**	2.04E-03	3.77E+02	3.35E+02	2.39E+02	8.89E+02	3.52E+01	3.57E+02	2.57E+01	1.82E+02	1.47E+02	2.05E+02

	min	**0.00E+00**	1.47E-03	3.09E+02	2.90E+02	2.11E+02	8.89E+02	1.99E+01	3.15E+02	1.25E+01	8.90E+01	1.08E+02	1.21E+02
	max	**0.00E+00**	2.42E-03	4.04E+02	3.71E+02	2.64E+02	8.89E+02	5.07E+01	3.97E+02	4.23E+01	3.60E+02	1.87E+02	3.00E+02
	std	**0.00E+00**	2.41E-04	1.95E+01	2.01E+01	1.18E+01	3.23E-13	7.37E+00	2.32E+01	6.31E+00	5.78E+01	1.97E+01	4.82E+01
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+
Sumsqu	mean	**0.00E+00**	6.38E-05	6.90E+02	6.17E+02	1.80E+02	2.74E+03	2.44E-01	9.37E-02	3.87E-01	2.78E+01	5.22E-01	1.20E+02
	min	**0.00E+00**	5.07E-05	5.24E+02	4.62E+02	1.40E+02	2.74E+03	2.03E-03	9.53E-03	5.40E-02	6.62E-01	2.73E-01	1.12E-02
	max	**0.00E+00**	7.79E-05	8.04E+02	7.37E+02	2.21E+02	2.74E+03	2.07E+00	1.29E+00	1.18E+00	1.69E+02	9.05E-01	7.60E+02
	std	**0.00E+00**	6.21E-06	6.84E+01	6.85E+01	2.02E+01	5.66E-13	4.01E-01	2.32E-01	2.93E-01	3.36E+01	1.38E-01	1.86E+02
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+
Powell	mean	**0.00E+00**	2.77E-05	1.73E+03	1.42E+03	3.01E+02	8.55E+03	1.90E+00	1.12E+03	3.76E+00	6.65E+01	4.41E-01	6.16E+02
	min	**0.00E+00**	1.76E-05	1.04E+03	8.37E+02	1.87E+02	8.55E+03	2.26E-01	6.49E+02	4.26E-01	4.90E+00	1.13E-01	1.53E-01
	max	**0.00E+00**	4.15E-05	2.35E+03	1.87E+03	4.09E+02	8.55E+03	4.53E+00	1.68E+03	9.06E+00	3.52E+02	1.23E+00	3.19E+03
	std	**0.00E+00**	6.75E-06	4.11E+02	2.77E+02	5.88E+01	1.01E-12	1.17E+00	2.55E+02	2.49E+00	7.76E+01	2.58E-01	8.42E+02
	RS	n/a	+	+	+	+	+	+	+	+	+	+	+

		Prop. PSO	**PSORL**	**RCPSO**	**GA**	**SA**
Ackley	mean	**4.44E-16**	6.33E+00	1.61E+01	2.16E+01	2.05E+01
	min	**4.44E-16**	2.22E+00	1.28E+01	2.14E+01	1.96E+01
	max	**4.44E-16**	1.29E+01	1.95E+01	2.18E+01	2.11E+01
	std	**0.00E+00**	2.67E+00	1.82E+00	9.59E-02	3.77E-01
	RS	n/a	+	+	+	+
Dixon	mean	6.67E-01	**3.43E-08**	7.43E+04	9.99E+06	1.11E+06
	min	6.67E-01	**1.42E-09**	2.46E+03	7.79E+06	7.48E+05
	max	6.67E-01	**3.68E-07**	3.80E+05	1.17E+07	1.53E+06
	std	2.38E-05	**7.10E-08**	1.03E+05	8.80E+05	1.98E+05
	RS	n/a	-	+	+	+
Griewank	mean	**0.00E+00**	**0.00E+00**	9.43E+01	8.69E+01	4.84E+02
	min	**0.00E+00**	**0.00E+00**	3.18E+01	7.50E+01	3.65E+02
	max	**0.00E+00**	**0.00E+00**	2.63E+02	9.20E+01	5.74E+02
	std	**0.00E+00**	**0.00E+00**	4.83E+01	4.03E+00	4.49E+01
	RS	n/a	=	+	+	+
Rastrigin	mean	**0.00E+00**	6.71E+01	2.35E+02	3.38E+01	3.92E+02
	min	**0.00E+00**	3.38E+01	1.40E+02	1.19E+01	3.52E+02
	max	**0.00E+00**	1.13E+02	2.89E+02	5.76E+01	4.25E+02
	std	**0.00E+00**	1.87E+01	3.00E+01	1.25E+01	1.74E+01
	RS	n/a	+	+	+	+
Rothyp	mean	**0.00E+00**	1.20E-167	5.96E+04	1.28E+06	3.15E+05
	min	**0.00E+00**	**0.00E+00**	1.43E+04	1.16E+06	2.48E+05
	max	**0.00E+00**	3.30E-166	1.71E+05	1.41E+06	3.68E+05
	std	**0.00E+00**	**0.00E+00**	3.29E+04	7.12E+04	2.87E+04
	RS	n/a	+	+	+	+
Rosenbrock	mean	**2.83E+01**	4.73E+01	8.66E+04	1.13E+07	8.33E+05
	min	2.78E+01	**5.65E+00**	4.41E+03	9.50E+06	5.21E+05
	max	**2.87E+01**	9.48E+01	4.84E+05	1.26E+07	1.19E+06
	std	**2.24E-01**	3.27E+01	1.00E+05	8.06E+05	1.92E+05
	RS	n/a	=	+	+	+
Sphere	mean	**0.00E+00**	2.14E-177	2.17E+01	5.08E+02	1.42E+02
	min	**0.00E+00**	**0.00E+00**	1.05E+01	4.56E+02	1.09E+02
	max	**0.00E+00**	6.32E-176	5.40E+01	5.63E+02	1.63E+02
	std	**0.00E+00**	**0.00E+00**	1.09E+01	3.17E+01	1.27E+01
	RS	n/a	+	+	+	+
Sumpow	mean	**0.00E+00**	**0.00E+00**	1.41E-02	2.98E+01	1.58E+00
	min	**0.00E+00**	**0.00E+00**	1.54E-06	2.93E+01	3.14E-01
	max	**0.00E+00**	**0.00E+00**	3.05E-01	3.00E+01	3.14E+00
	std	**0.00E+00**	**0.00E+00**	5.59E-02	2.20E-01	6.56E-01
	RS	n/a	=	+	+	+
Zakharov	mean	**0.00E+00**	1.15E+02	4.08E+02	6.30E+02	6.26E+02
	min	**0.00E+00**	6.67E+01	2.46E+02	5.36E+02	5.34E+02
	max	**0.00E+00**	1.75E+02	5.41E+02	7.27E+02	7.01E+02
	std	**0.00E+00**	2.97E+01	7.69E+01	3.93E+01	4.37E+01
	RS	n/a	+	+	+	+
Sumsqu	mean	**0.00E+00**	2.16E-174	3.30E+02	7.95E+03	2.02E+03
	min	**0.00E+00**	**0.00E+00**	5.71E+01	6.79E+03	1.49E+03
	max	**0.00E+00**	6.47E-173	7.54E+02	9.53E+03	2.34E+03
	std	**0.00E+00**	**0.00E+00**	1.83E+02	5.97E+02	1.89E+02
	RS	n/a	+	+	+	+
Powell	mean	**0.00E+00**	5.10E-04	8.76E+02	8.36E+03	7.81E+03
	min	**0.00E+00**	1.75E-04	1.76E+02	5.56E+03	4.27E+03
	max	**0.00E+00**	1.55E-03	2.96E+03	1.14E+04	1.28E+04
	std	**0.00E+00**	2.51E-04	7.86E+02	1.58E+03	1.85E+03
	RS	n/a	+	+	+	+

6.5.1. Convergence Analysis and Comparison

We conduct a theoretical convergence analysis of the proposed PSO variant, as follows. As discussed earlier, the proposed model incorporates reinforcement learning-based optimal search action selection, cross-breed elite signal generation based on adaptive 3D geometric contours, mathematical root-finding algorithm based swarm leader enhancement, and a spiral simulated search mechanism, to bridge the current research gaps and overcome limitations (local optimum traps) of the original PSO model. Specifically, in comparison with random walk operations such as Levy distributions for swarm leader improvement as in existing studies (Jordehi [64] and Zhang et al. [65]), we exploit root-finding algorithms, i.e. Muller's method and the fixed-point iteration

algorithm, guided by the mathematical principles, to provide more informative mechanisms for swarm leader enhancement. Such a strategy is able to accelerate convergence within a small number of iterations in comparison with those from random jump mechanisms. In order to better balance between search diversification and intensification, instead of using random or threshold-based search action selection, the reinforcement Q-learning algorithm is used to dispatch a sequence of local and global search actions, leading to the most optimal long-term cumulative reward, therefore diversifying search behaviours while accelerating convergence.

To overcome local optima traps, instead of using a single swarm leader as in the original PSO model and other existing works, a variety of hybrid leaders integrating the global and personal best solutions are used to lead the search process and to divert the search out of local optima traps when the search operation led by the single global best solution becomes stagnant. Moreover, adaptive weighting coefficients using distinctive 3D formulae are utilized to better adjust the impact of the two local and global leader signals, achieving a better balance between diversification and intensification.

In short, our proposed search strategies, i.e. the reinforcement learning search action selection, hybrid leaders fine-tuned using adaptive weighting factors in conjunction with swarm leader improvement using mathematical informative root-finding methods, operate cooperatively to increase search diversity, avoid local optimum traps and fasten model convergence.

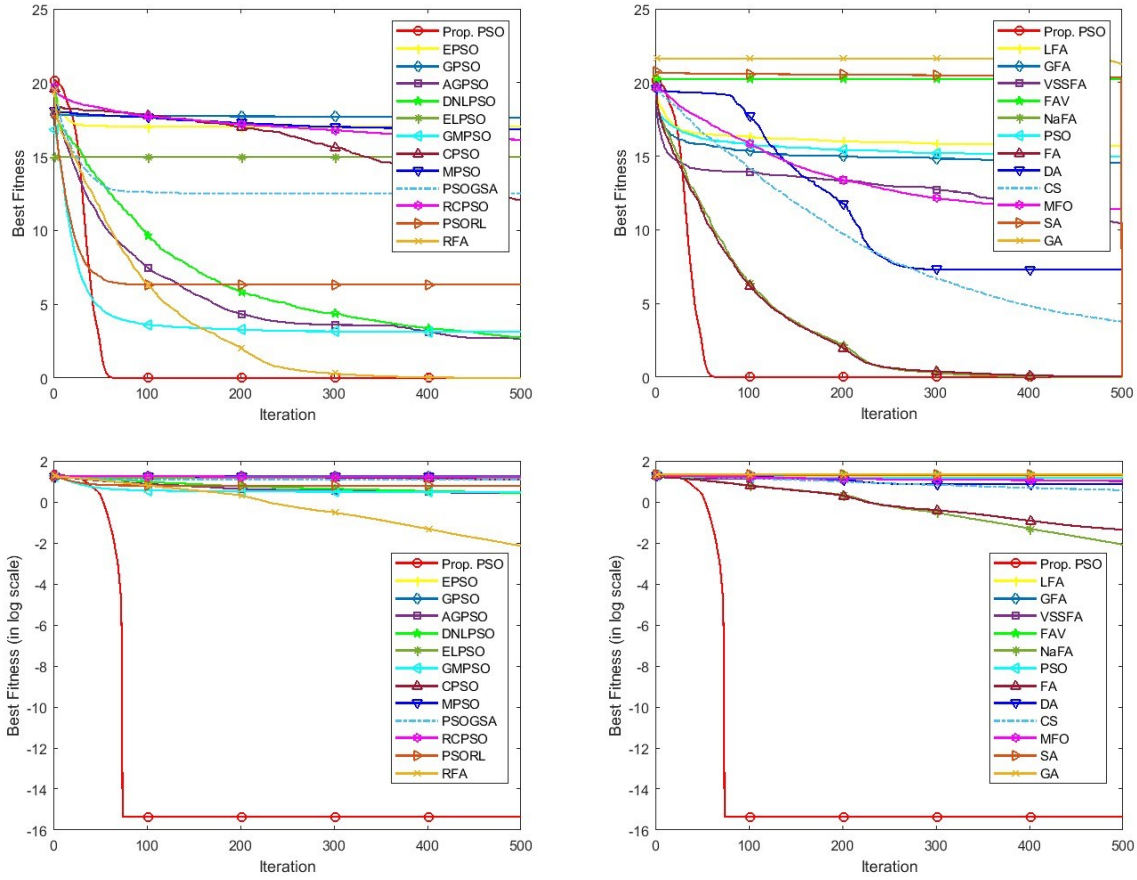

Figure 11 Mean convergence curve comparison between the proposed PSO and all baseline search methods for Ackley over 30 runs (Top: in the original forms, bottom: in the logarithm scales)

To ascertain the above theoretical convergence analysis, Figure 11 depicts a convergence comparison between our algorithm and other search methods for Ackley over 30 runs. The convergence curve of each method is generated by averaging the global best solutions in each iteration over 30 runs, as indicated in the top row in Figure 11. To further indicate the convergence speed of the proposed model, a logarithm function with a base of 10 is used to convert the convergence curves into the logarithm scale, as shown in the bottom row in Figure 11. As indicated in Figure 11, the proposed model, guided by multiple hybrid leaders, root finding algorithm-based leader enhancement and Q-learning based optimal search action selection, demonstrates faster convergence rates than those from all other search methods for the Ackley function. RFA, NaFA, FA, CS and GMPSO also illustrate comparatively faster convergence rates than those from other search methods.

We also conduct a convergence speed comparison between our algorithm and baseline search methods for the Rastrigin function over a set of 30 trials, as shown in Figure 12. To clearly showcase the model convergence speed, the original convergence curves and the converted convergence graphs in the log scale are provided in the top and bottom rows in Figure 12, respectively. As indicated in Figure 12, our model depicts the fastest convergence speed as compared with those from all the baseline methods, and obtains the minimum solution of '0' on average at iteration 107 over a set of 30 runs. Owing to the fact that $\log_{10}(0) = -\infty$, which cannot be represented by a numerical value, our convergence graph in the log scale (i.e. the bottom row in Figure 12) shows the mean global best values until iteration 106. In addition, RFA, GMPSO, AGPSO, NaFA and FA also achieve better results than those from other baselines over 30 runs with a comparatively faster convergence speed.

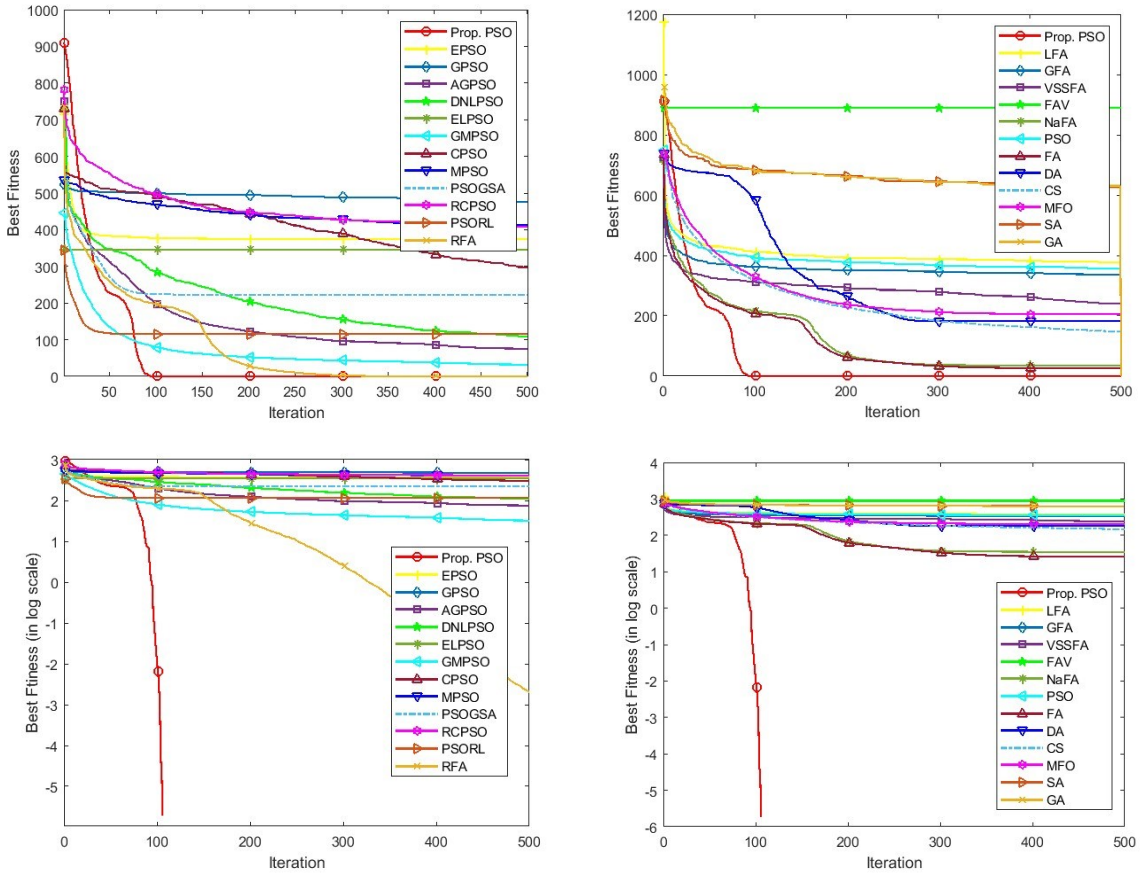

Figure 12 Mean convergence curve comparison between the proposed PSO and all baseline search methods for Rastrigin over 30 runs (Top: in the original forms, bottom: in the logarithm scales)

Figure 13 depicts the convergence comparison between our algorithm and other search methods for the Powell function over 30 runs. Again, the proposed model outperforms all baseline search methods with the fastest convergence rates. The proposed model attains the global minimum of '0' at iteration 72 and $\log_{10}(0) = -\infty$, therefore the log-scale convergence graph in the bottom row in Figure 13 is provided until iteration 71. PSORL, RFA, GMPSO, PSOGSA, CS, NaFA and FA also demonstrate competitive performance than those from other baseline search methods with a comparatively faster convergence speed. Similar faster convergence rates of the proposed model are also observed as compared with those from all other search methods for nearly all numerical functions.

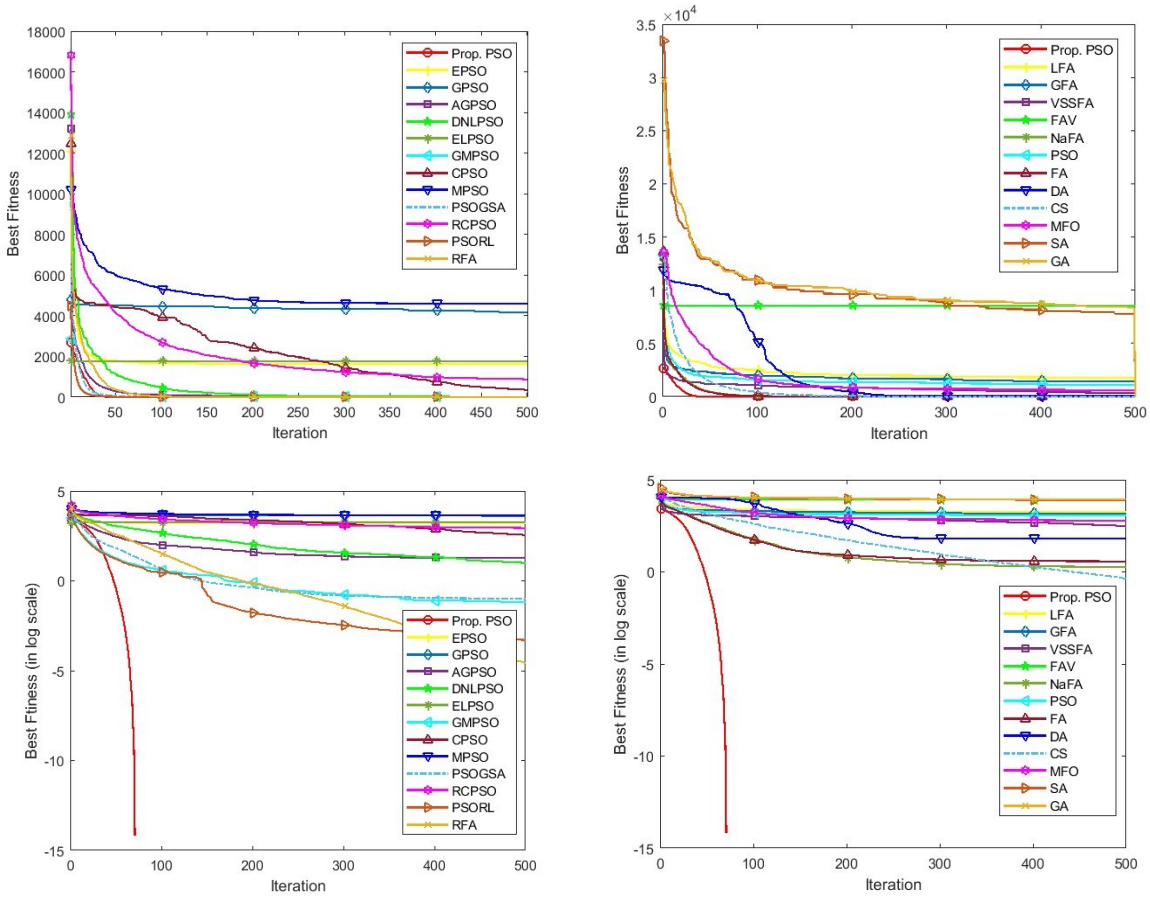

Figure 13 Mean convergence curve comparison between the proposed PSO and all baseline search methods for Powell over 30 runs (Top: in the original forms, bottom: in the logarithm scales)

We justify model performance variations from theoretical perspectives. MPSO and CPSO use adaptive linear and cosine coefficients to formulate search exploration and exploitation. However, because these methods adopt a single swarm leader as in the original PSO, they are likely to become stagnant. Instead of using linear or cosine search coefficients, EPSO employs elliptical functions for search parameter generation as well as DE for leader enhancement to diversify the search process. Nonetheless, its search operation is guided by either the global best solution or an average leader signal determined by a random probability, without producing any fused leader signals. In addition, genetic operators and probability distributions such as Levy flight are embedded in GMPSO for swarm leader enhancement. GPSO and ELPSO employ crossover, and opposition and DE-based operators for leader and population diversity enhancement, while AGPSO utilizes adaptive search coefficients to fine-tune intensification and diversification. DNLPSO adopts dynamic historical neighbouring elite signals for search exploration. PSOGSA embeds PSO and GSA for undertaking exploration and exploitation, respectively. Notice that these PSO variants either rely on the original PSO operation or a neighbouring optimal signal-led search process for position updating, without taking advantage of hybrid leader indicators.

RFA, NaFA, and VSSFA employ repulsive strategies, reserved neighbourhood attraction, and variable step sizes for search diversification, while LFA and GFA exploit Logistic and Gauss maps-motivated search coefficients, to overcome stagnation. FAV leverages the Tent map for population initialization and a swarm leader-based search process combined with a FA-based neighbourhood attraction mechanism. The primary search operations of these FA variants mainly adopt either individual neighbouring or global best signals in search for optimality without exploiting adaptive composite leaders. Besides PSO and FA employing global and neighbouring optimal solutions for position updating, MFO employs a spiral search operation to exploit the optimal regions of each flame. CS uses signals selected using random permutation to guide the swarm. These classical search methods use either global/neighbouring elite indicators or random search agents as the guiding signals without taking advantage of the fusion of these signals for search exploration, therefore yielding less optimal performance with slow convergence rates.

In comparison with the above methods, instead of using random mutations, the proposed PSO algorithm employs fixed-point iteration and Muller's method as navigated steered leader enhancement strategies. It also takes advantage of the fusion of diverse local and global elite signals constructed by adaptive 3D geometrical operators to tackle the constraints of local optima traps suffered by the original PSO algorithm. A petal helix search mechanism is utilized to exploit optimal regions around the cross-breed leaders. The Q-learning method is then adopted to identify an optimal distribution of these local and global search operations to assist a better balance of exploration and intensification. The aforementioned 3D geometric landscape-inspired cross-breed leader generation, reinforcement learning-based sequential search scheme deployment, petal spiral local intensification and root finding algorithm-based leader enhancement, account for the strength and superiority of the proposed PSO algorithm against other search methods in solving diverse numerical optimization formulae with a variety of challenging landscapes.

7. CONCLUSIONS

In this research, we have devised weighted and evolving ensemble models integrating CNN-RNN, I3D and MC3 networks with newly proposed PSO-based network topology and hyper-parameter optimization for video authenticity classification. The ensemble robustness is enhanced by the proposed PSO-optimized network hyper-parameters as well as the optimal selection of subsets of base classifiers. Specifically, the new PSO variant employs several schemes, including numerical analysis-based leader enhancement, the Q-learning based optimal search operation selection, petal helix search intensification and cross-breed elite signal generation using adaptive 3D landscapes, to overcome the limitations of the original PSO model.

In both weighted and evolving ensemble schemes, the superior performance of our devised ensemble models integrating diverse optimized base networks is evidenced in our experimental studies and through the statistical test results, as compared with those from counterparts yielded by existing search algorithms. Because of the optimal selection of complementary subsets of base classifiers using the proposed algorithm, our resulting evolving ensemble models achieve the most competitive performance, while maintaining efficient computational costs, in comparison with those from the fusion models yielded by other search methods. The proposed algorithm also outperforms 24 baseline search methods with statistical significance in solving diverse numerical optimization problems with challenging landscapes.

In future work, other 3D CNNs such as 3D ResNeXt and 3D ResNet will be examined to complement spatial-temporal feature learning and to further boost performance. Besides video deepfake detection, the proposed PSO algorithm will be employed to formulate other deep networks, e.g. BiLSTM and CRNN [28], to tackle audio deepfake detection. The fusion of optimized audio and video forgery detection methods will also be investigated to enhance the performance from single modality [28]. The proposed PSO-based deep architecture and key learning configuration search will be applied to other challenging vision processing tasks such as video/image generation [121] and captioning [122, 123].

REFERENCES

[1] Y. Li, Xin Yang, P. Sun, H. Qi, and S. Lyu. 2020. Celeb-DF: A large-scale challenging dataset for deepfake forensics. In *Proceedings of the IEEE International Conference on Computer Vision*, pp.3204–3213. IEEE.

[2] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, M. Nießner. 2019. FaceForensics++: learning to detect manipulated facial images. In *Proceedings of the IEEE International Conference on Computer Vision*, pp.1–11. IEEE.

[3] Y. Mirsky, T. Mahler, I. Shelef, and Y. Elovici. 2019. CT-GAN: Malicious Tampering of 3D Medical Imagery using Deep Learning. In *Proceedings of the 28th USENIX Security Symposium (USENIX Security)*, pp.461-478. IEEE.

[4] A. Chintha, B. Thai, S.J. Sohrawardi, K. Bhatt, A. Hickerson, M. Wright, and R. Ptucha. 2020. Recurrent convolutional structures for audio spoof and video deepfake detection. *IEEE Journal of Selected Topics in Signal Processing*, 14(5), pp.1024-1037.

[5] J. Wang, Y. Sun, and J. Tang. 2022. LiSiam: Localization Invariance Siamese Network for Deepfake Detection. *IEEE Transactions on Information Forensics and Security*, 17, pp.2425-2436.

[6] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia. 2020. Deepfakes and beyond: A survey of face manipulation and fake detection. *Information Fusion*, *64*, pp.131-148.

[7] U.A. Ciftci, I. Demir, and L. Yin. 2020. Fakecatcher: Detection of synthetic portrait videos using biological signals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp.1-17.

[8] T.T. Nguyen, Q.V.H. Nguyen, D.T. Nguyen, D.T. Nguyen, T. Huynh-The, S. Nahavandi, T.T. Nguyen, Q.V. Pham, and C.M. Nguyen. 2022. Deep learning for deepfakes creation and detection: A survey. *Computer Vision and Image Understanding*, 223, p.103525.

[9] T. Zhang. 2022. Deepfake generation and detection, a survey. *Multimedia Tools and Applications,* 81(5), pp.6259-6276.

[10] J. Carreira, and A. Zisserman. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.6299-6308. IEEE.

[11] G. Liu, C. Zhang, Q. Xu, R. Cheng, Y. Song, X. Yuan, and J. Sun. 2020. I3d-shufflenet based human action Recognition. *Algorithms*, 13(11), p.301.

[12] Z. Sun, Q. Ke, H. Rahmani, M. Bennamoun, G. Wang, and J. Liu. 2022. Human action recognition from various data modalities: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3), pp.3200-3225.

[13] L. Zhang, C.P. Lim, and Y. Yu, 2021. Intelligent human action recognition using an ensemble model of evolving deep networks with swarm-based optimization. *Knowledge-Based Systems*, 220, p.106918.

[14] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. 2018. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.6450-6459. IEEE.

[15] D. Zhang, C. Li, F. Lin, D. Zeng and S. Ge. 2021. Detecting Deepfake Videos with Temporal Dropout 3DCNN. In *Prcoeedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp.1288-1294.

[16] M. Majd and R. Safabakhsh. 2020. Correlational convolutional LSTM for human action recognition. *Neurocomputing*, 396, pp.224-229.

[17] G. Petmezas, G.A. Cheimariotis, L. Stefanopoulos, B. Rocha, R.P. Paiva, A.K. Katsaggelos, and N. Maglaveras. 2022. Automated Lung Sound Classification Using a Hybrid CNN-LSTM Network and Focal Loss Function. *Sensors*, 22(3), p.1232.

[18] L. Zhang, C.P. Lim, Y. Yu, and M. Jiang. 2022. Sound classification using evolving ensemble models and Particle Swarm Optimization. *Applied Soft Computing*, 116, p.108322.

[19] P. Dasari, L. Zhang, Y. Yu, H. Huang, and R. Gao. 2022. Human action recognition using hybrid deep evolving neural networks. In *Proceedings of International Joint Conference on Neural Networks,* pp.1-8. IEEE.

[20] J. Kennedy and R. Eberhart. 1995. Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, vol. 4, pp.1942-1948. IEEE.

[21] W. Pu, J. Hu, X. Wang, Y. Li, S. Hu, B. Zhu, R. Song, Q. Song, X. Wu, and S. Lyu. 2022. Learning a deep dual-level network for robust DeepFake detection. *Pattern Recognition*, 130, p.108832.

[22] Z. Shang, H. Xie, Z. Zha, L. Yu, Y. Li, and Y. Zhang. 2021. PRRNet: Pixel-Region relation network for face forgery detection. *Pattern Recognition*, 116, p.107950.

[23] G. Wang, Q. Jiang, X. Jin, W. Li, and X. Cui. 2022. MC-LCR: Multimodal contrastive classification by locally correlated representations for effective face forgery detection. *Knowledge-Based Systems*, p.109114.

[24] B. Chen, T. Li, and W. Ding. 2022. Detecting deepfake videos based on spatiotemporal attention and convolutional LSTM. *Information Sciences*, 601, pp.58-70.

[25] H.H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen. 2019. Multi-task learning for detecting and segmenting manipulated facial images and videos, *arXiv:1906.06876*.

[26] S.Y. Wang, O. Wang, R. Zhang, A. Owens, and A.A. 2020. CNN-generated images are surprisingly easy to spot... for now. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp.8695-8704. IEEE.

[27] Z. Guo, L. Hu, M. Xia, and G. Yang. 2021. Blind detection of glow-based facial forgery. *Multimedia Tools and Applications*, 80(5), pp.7687-7710.

[28] Z. Almutairi and H. Elgibreen. 2022. A review of modern audio deepfake detection methods: Challenges and Future Directions. *Algorithms*, 15(5), p.155.

[29] M. Shan and T. Tsai. 2020. A cross-verification approach for protecting world leaders from fake and tampered audio. arXiv 2020, arXiv:2010.12173

[30] D.M. Ballesteros, Y. Rodriguez-Ortega, D. Renza, and G. Arce. 2021. Deep4SNet: deep learning for fake speech classification. *Expert Systems with Applications*. 184, p.115465.

[31] H. Khalid, M. Kim, S. Tariq, and S.S. Woo. 2021. Evaluation of an audio-video multimodal deepfake dataset using unimodal and multimodal detectors. In *Proceedings of the 1st Workshop on Synthetic Multimedia*, ACM Association for Computing Machinery, New York, NY, USA, pp.7–15. ACM.

[32] H. Khalid, S. Tariq, M. Kim, and SS. Woo. 2021. FakeAVCeleb: A novel audio-video multimodal deepfake dataset. In *Proceedings of the 35th Conference on Neural Information Processing Systems,* pp.1-14.

[33] X. Liu, X. Wang, M. Sahidullah, J. Patino, H. Delgado, T. Kinnunen, M. Todisco, J. Yamagishi, N. Evans, A. Nautsch, and K.A. Lee, 2023. Asvspoof 2021: Towards spoofed and deepfake speech detection in the wild. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31, pp.2507-2522.

[34] H. Xie, L. Zhang, L., C.P. Lim, Y. Yu, C. Liu, H. Liu, and J. Walters. 2019. Improving K-means clustering with enhanced Firefly Algorithms. *Applied Soft Computing*, 84, p.105763.

[35] D. Zhang, G. Ma, Z. Deng, Q. Wang, G. Zhang, and W. Zhou. 2022. A self-adaptive gradient-based particle swarm optimization algorithm with dynamic population topology. *Applied Soft Computing*, 130, p.109660.

[36] A. Zhang, H. Xu, W. Bi, and S. Xu. 2022. Adaptive mutant particle swarm optimization based precise cargo airdrop of unmanned aerial vehicles. *Applied Soft Computing*, 130, p.109657.

[37] W. Liang, Y. Zhang, X. Liu, H. Yin, J. Wang, and Y. Yang. 2022. Towards improved multifactorial particle swarm optimization learning of fuzzy cognitive maps: A case study on air quality prediction. *Applied Soft Computing*, 130, p.109708.

[38] Q. Liu, J. Li, H. Ren, and W. Pang. 2022. All particles driving particle swarm optimization: Superior particles pulling plus inferior particles pushing. *Knowledge-Based Systems*, 249, p.108849.

[39] B. Liu, M. Xu, L. Gao, J. Yang, and X. Di. 2022. A hybrid approach for high-dimensional optimization: Combining particle swarm optimization with mechanisms in neuro-endocrine-immune systems. *Knowledge-Based Systems*, 253, p.109527.

[40] J. Lu, J. Zhang, and J. Sheng. 2022. Enhanced multi-swarm cooperative particle swarm optimizer. *Swarm and Evolutionary Computation*, 69, p.100989.

[41] H. Li, J. Li, P. Wu, Y. You, and N. Zeng. 2022. A ranking-system-based switching particle swarm optimizer with dynamic learning strategies. *Neurocomputing*, 494, pp.356-367.

[42] Y. Chen, L. Li, J. Xiao, Y. Yang, J. Liang, and T. Li. 2018. Particle swarm optimizer with crossover operation. *Engineering Applications of Artificial Intelligence*, 70, pp.159-169.

[43] B. Fielding and L. Zhang. 2018. Evolving image classification architectures with enhanced particle swarm optimisation. *IEEE Access*, 6, pp.68560-68575.

[44] D.R. Nayak, R. Dash and B. Majhi. 2018. Discrete ripplet-II transform and modified PSO based improved evolutionary extreme learning machine for pathological brain detection. *Neurocomputing*, 282, pp.232-247.

[45] T.Y. Tan, L. Zhang, and C.P. Lim. 2019. Intelligent skin cancer diagnosis using improved particle swarm optimization and deep learning models. *Applied Soft Computing*, 84, p.105725.

[46] L. Zhang and C.P. Lim. 2020. Intelligent optic disc segmentation using improved particle swarm optimization and evolving ensemble models. *Applied Soft Computing*, 92, p.106328.

[47] S. Slade, L. Zhang, Y. Yu, and C.P. Lim. 2022. An evolving ensemble model of multi-stream convolutional neural networks for human action recognition in still images, *Neural Computing and Applications*, *34*(11), pp.9205-9231.

[48] P.R. Lorenzo, J. Nalepa, M. Kawulok, L.S. Ramos, and J.R. Pastor, 2017. Particle swarm optimization for hyper-parameter selection in deep neural networks. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp.481-488. ACM.

[49] P.R. Lorenzo, J. Nalepa, L.S. Ramos, and J.R. Pastor, 2017. Hyper-parameter selection in deep neural networks using parallel particle swarm optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp.1864-1871. ACM.

[50] F.E.F. Junior, and G.G. Yen, 2019. Particle swarm optimization of deep neural networks architectures for image classification. *Swarm and Evolutionary Computation*, 49, pp.62-74.

[51] T. Lawrence, L. Zhang, K. Rogage, and C.P. Lim, 2021. Evolving deep architecture generation with residual connections for image classification using particle swarm optimization. *Sensors*, 21(23), p.7936.

[52] B. Baker, O. Gupta, N. Naik, and R. Raskar. 2017. Designing neural network architectures using reinforcement learning. In *Proceedings of International Conference on Learning Representations,* pp.1-18. Curran Associates, Inc.

[53] B. Zoph, V. Vasudevan, J. Shlens, Q.V. Le. 2018. Learning transferable architectures for scalable image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.8697-8710. IEEE.

[54] Y. Gu, Y. Cheng, C.P. Chen, and X. Wang. 2021. Proximal policy optimization with policy feedback. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 52(7), pp.4600-4610.

[55] Q. Shen, Y. Li, H. Jiang, Z. Wang, and T. Zhao. 2020. Deep reinforcement learning with robust and smooth policy. In *Proceedings of International Conference on Machine Learning*, pp.8707-8718. PMLR.

[56] R.S. Sutton, and A.G. Barto. 2018. *Reinforcement learning: an introduction*. MIT press.

[57] T.Y. Tan, L. Zhang, and C.P. Lim, 2020. Adaptive melanoma diagnosis using evolving clustering, ensemble and deep neural networks. *Knowledge-Based Systems*, 187, p.104807.

[58] L. Zhang, S. Slade, C.P. Lim, H. Asadi, S. Nahavandi, H. Huang, and H. Ruan. 2023. Semantic segmentation using Firefly Algorithm-based evolving ensemble deep neural networks. *Knowledge-Based Systems*, 277, p.110828.

[59] A. Fallahi, E.A. Bani, and S.T.A. Niaki, 2022. A constrained multi-item EOQ inventory model for reusable items: Reinforcement learning-based differential evolution and particle swarm optimization. *Expert Systems with Applications*, 207, p.118018.

[60] L. Zhang, C.P. Lim, and C. Liu. 2023. Enhanced bare-bones particle swarm optimization based evolving deep neural networks. *Expert Systems with Applications*, p.120642.

[61] X. He, K. Zhao, and X. Chu. 2021. AutoML: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212, p.106622.

[62] R. Elshawi, M. Maher, and S. Sakr. 2019. Automated machine learning: state-of-the-art and open challenges. *arXiv preprint arXiv:1906.02287*.

[63] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D.B. Tsai, M. Amde, S. Owen, and D. Xin. 2016. Mllib: machine learning in apache spark. *Journal of Machine Learning Research*, 17(1), pp.1235-1241.

[64] A.R. Jordehi. 2015. Enhanced leader PSO (ELPSO): a new PSO variant for solving global optimisation problems. *Applied Soft Computing*. 26, pp.401–417.

[65] Y. Zhang, L. Zhang, S.C. Neoh, K. Mistry and A. Hossain. 2015. Intelligent affect regression for bodily expressions using hybrid particle swarm optimization and adaptive ensembles. *Expert Systems with Applications*. 42(22), pp.8678–8697.

[66] L. Zhang, K. Wang, L. Xu, W. Sheng, and Q. Kang. 2022. Evolving ensembles using multi-objective genetic programming for imbalanced classification. *Knowledge-based Systems*, 255, p.109611.

[67] Q. Fan, Y. Bi, B. Xue, and M. Zhang. 2022. Evolving effective ensembles for image classification using multi-objective multi-tree genetic programming. In *Proceedings of Australasian Joint Conference on Artificial Intelligence*, pp.294-307. Cham: Springer International Publishing.

[68] P. Bosowski, J. Bosowska, and J. Nalepa. 2021. Evolving deep ensembles for detecting covid-19 in chest X-rays. In *Proceedings of IEEE International Conference on Image Processing,* pp.3772-3776. IEEE.

[69] J. Nalepa, M. Myller, L. Tulczyjew, and M. Kawulok. 2021. Deep ensembles for hyperspectral image data classification and unmixing. *Remote Sensing*, 13(20), p.4133.

[70] M. Pratama, W. Pedrycz, and E. Lughofer. 2018. Evolving ensemble fuzzy classifier. *IEEE Transactions on Fuzzy Systems*, 26(5), pp.2552-2567.

[71] G. Ngo, R. Beard, and R. Chandra. 2022. Evolutionary bagging for ensemble learning. *Neurocomputing*, 510, pp.1-14.

[72] L. Zhang, W. Srisukkham, S.C. Neoh, C.P. Lim, and D. Pandit, 2018. Classifier ensemble reduction using a modified firefly algorithm: An empirical evaluation. *Expert Systems with Applications*, 93, pp.395-422.

[73] X. Cai, L. Ye, and Q. Zhang. 2018. Ensemble learning particle swarm optimization for real-time UWB indoor localization. *EURASIP Journal on Wireless Communications and Networking*, 1, pp.1-15.

[74] R. Malhotra, and M. Khanna. 2018. Particle swarm optimization-based ensemble learning for software change prediction. *Information and Software Technology*, 102, pp.65-84.

[75] L. Hong, G. Wang, E. Özcan, and J. Woodward. 2023. Ensemble strategy using particle swarm optimisation variant and enhanced local search capability. *Swarm and Evolutionary Computation*, p.101452.

[76] W. Shafqat, S. Malik, K.T. Lee, and D.H. Kim, 2021. PSO based optimized ensemble learning and feature selection approach for efficient energy forecast. *Electronics*, 10(18), p.2188.

[77] C.J. Tan, S.C. Neoh, C.P. Lim, S. Hanoun, W.P. Wong, C.K. Loo, L. Zhang, and S. Nahavandi. 2019. Application of an evolutionary algorithm-based ensemble model to job-shop scheduling. *Journal of Intelligent Manufacturing*, 30, pp.879-890.

[78] H.E. Cagnini, S.C.D. Dôres, A.A. Freitas, and R.C. Barros, 2023. A survey of evolutionary algorithms for supervised ensemble learning. *The Knowledge Engineering Review*, 38, p.e1.

[79] E. Süli, and D.F. Mayers, 2003. An introduction to numerical analysis. Cambridge university press.

[80] C.J. Watkins, and P. Dayan, 1992. Q-learning. *Machine Learning*, 8, pp.279-292.

[81] K. Zhang, Z. Zhang, Z., Li, and Y. Qiao. 2016. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10), pp.1499-1503.

[82] X.S. Yang. 2010. Firefly algorithm, levy flights and global optimization. *Research and Development in Intelligent Systems.* 26, pp.209–218.

[83] S. Mirjalili. 2016. Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications*, 27(4), pp.1053-1073.

[84] X.S. Yang and S. Deb. 2009. Cuckoo search via Lévy flights. In *Proceedings of World Congress on Nature and Biologically Inspired Computing,* pp.210-214. IEEE.

[85] S. Mirjalili, A. Lewis and A.S. Sadiq. 2014. Autonomous particles groups for particle swarm optimization. *Arabian Journal for Science and Engineering*. 39(6), pp.4683–4697.

[86] S. Mirjalili and S.Z.M. Hashim. 2010. A new hybrid PSOGSA algorithm for function optimization. In *Proceedings of International Conference on Computer and Information Application,* pp.374-377.

[87] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, D. 2017. Grad-cam: visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp.618-626. IEEE.

[88] I. Demir and U.A. Ciftci. 2021. Where do deep fakes look? synthetic face detection via gaze tracking. In *Proceedings of ACM Symposium on Eye Tracking Research and Applications*, pp.1-11. ACM.

[89] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen. 2018. MesoNet: a compact facial video forgery detection network. In *Proceedings of IEEE International Workshop on Information Forensics and Security*, pp.1-7. IEEE.

[90] H.H. Nguyen, J. Yamagishi, and I. Echizen. 2019. Capsule-forensics: using capsule networks to detect forged images and videos. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pp.2307-2311. IEEE.

[91] R. Wang, F. Juefei-Xu, L. Ma, X. Xie, Y. Huang, J. Wang, and Y. Liu. 2019. FakeSpotter: A simple yet robust baseline for spotting ai-synthesized fake faces. In *Proceedings of International Joint Conference on Artificial Intelligence*, pp.3444-3451. ACM.

[92] H. Liu, X. Li, W. Zhou, Y. Chen, Y. He, H. Xue, W. Zhang, and N. Yu. 2021. Spatial-phase shallow learning: rethinking face forgery detection in frequency domain. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.772-781. IEEE.

[93] H. Zhao, W. Zhou, D. Chen, T. Wei, W. Zhang, and N. Yu. 2021. Multi-attentional deepfake detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.2185-2194. IEEE.

[94] J. Yang, A. Li, S. Xiao, W. Lu, and X. Gao. 2021. MTD-Net: learning to detect deepfakes images by multi-scale texture difference. *IEEE Transactions on Information Forensics and Security*, 16, pp.4234-4245.

[95] P. Zhou, X. Han, V.I. Morariu, and L.S. Davis. 2017. Two-stream neural networks for tampered face detection. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp.1831-1839. IEEE.

[96] Y. Li and S. Lyu. 2019. Exposing deepfake videos by detecting face warping artifacts. 2019. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp.46-52. IEEE.

[97] J. Fridrich and J. Kodovsky. 2012. Rich models for steganalysis of digital images. *IEEE Transactions on information Forensics and Security*, 7(3), pp.868-882.

[98] D. Cozzolino, G. Poggi, and L. Verdoliva. 2017. Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection. In *Proceedings of ACM Workshop on Information Hiding and Multimedia Security*, pp.159-164. ACM.

[99] B. Bayar and M.C. Stamm, 2016. A deep learning approach to universal image manipulation detection using a new convolutional layer. In *Proceedings of ACM Workshop on Information Hiding and Multimedia Security*, pp.5-10. ACM.

[100] N. Rahmouni, V. Nozick, J. Yamagishi, and I. Echizen. 2017. Distinguishing computer graphics from natural images using convolution neural networks. In *Proceedings of IEEE Workshop on Information Forensics and Security*, pp.1–6. IEEE.

[101] T.S. Gunawan, , S.A.M. Hanafiah, M. Kartiwi, N. Ismail, N.F. Zabah, and A.N. Nordin. 2017. Development of photo forensics algorithm by detecting photoshop manipulation using error level analysis. *Indonesian Journal of Electrical Engineering and Computer Science*, *7*(1), pp.131-137.

[102] Z. Zhang, C. Mal, B. Ding, and M. Gao. 2021. Detecting manipulated facial videos: a time series solution. In *Proceedings of the International Conference on Pattern Recognition,* pp. 2817-2823. IEEE.

[103] K.W. Kim, H.G. Hong, G.P. Nam, and K.R. Park. 2017. A study of deep cnn-based classification of open and closed eyes using a visible light camera sensor. *Sensors*, 17, 7. p.1534.

[104] Y. Li, M. Chang, and S. Lyu. 2018. In ictu oculi: exposing ai created fake videos by detecting eye blinking. In *Proceedings of IEEE International Workshop on Information Forensics and Security*, pp.1–7. IEEE.

[105] D. Güera, and E.J. Delp, 2018. Deepfake video detection using recurrent neural networks. In *Proceedings of IEEE International Conference on Advanced Video and Signal based Surveillance,* pp.1-6. IEEE.

[106] S.J. Sohrawardi, A. Chintha, B. Thai, S. Seng, A. Hickerson, R. Ptucha, and M. Wright. 2019. Towards robust open-world detection of deepfakes. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pp.2613-2615. ACM.

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

[108] L. Li, J. Bao, T. Zhang, H. Yang, D. Chen, F. Wen, and B. Guo. 2020. Face x-ray for more general face forgery detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.5001-5010. IEEE.

[109] Y. Xu and S.Y. Yayilgan. 2022. When handcrafted features and deep features meet mismatched training and test sets for deepfake detection. *arXiv preprint arXiv:2209.13289*.

[110] X. Li, Y. Lang, Y. Chen, X. Mao, Y. He, S. Wang, H. Xue, Q. Lu. 2020. Sharp multiple instance learning for deepfake video detection. In *Proceedings of the ACM International Conference on Multimedia*, pp.1864-1872. ACM.

[111] B. Zi, M. Chang, J. Chen, X. Ma, and Y.G. Jiang, 2020. Wilddeepfake: A challenging real-world dataset for deepfake detection. In *Proceedings of the ACM International Conference on Multimedia*, pp.2382-2390. ACM.

[112] S. Mirjalili. 2015. Moth-Flame optimization algorithm: A novel nature-inspired heuristic paradigm, *Knowledge-Based Systems*. 89, pp.228-249.

[113] Q. Chen, Y. Chen and W. Jiang. 2016. Genetic particle swarm optimization-based feature selection for very-high-resolution remotely sensed imagery object change detection. *Sensors*, 16(8), p.1204.

[114] M. Nasir, S. Das, D. Maity, S. Sengupta, U. Halder and P.N. Suganthan. 2012. A dynamic neighborhood learning based particle swarm optimizer for global numerical optimization. *Information Sciences*. 209, pp.16-36.

[115] D. Pandit, L. Zhang, S. Chattopadhyay, C.P. Lim and C. Liu. 2018. A scattering and repulsive swarm intelligence algorithm for solving global optimization problems. *Knowledge-Based Systems*. 156, pp.12-42.

[116] A. Kazem, E. Sharifi, F.K. Hussain, M. Saberic and O.K. Hussain. 2013. Support vector regression with chaos-based firefly algorithm for stock market price forecasting, *Applied Soft Computing*. 13(2), pp.947-958.

[117] A.H. Gandomi, X.S. Yang, S. Talatahari and A.H. Alavi. 2013. Firefly algorithm with chaos, *Communications in Nonlinear Science and Numerical Simulation*, 18, pp.89-98.

[118] S.H. Yu, S.L. Zhu, Y. Ma and D.M. Mao. 2015. A variable step size firefly algorithm for numerical optimization. *Applied Mathematics and Computation*. 263, pp.214-220.

[119] L. He and S. Huang. 2017. Modified firefly algorithm based multilevel thresholding for colour image segmentation. *Neurocomputing,* 240, pp.152-174.

[120] H. Wang, W. Wang, X. Zhou, H. Sun, J. Zhao, X. Yu and Z. Cui. 2017. Firefly algorithm with neighborhood attraction. *Information Sciences*. 382, pp.374-387.

[121] S. Liu, J. Ye, S. Ren, and X. Wang. 2022. Dynast: dynamic sparse transformer for exemplar-guided image generation. In *Proceedings of European Conference on Computer Vision*, pp.72-90. Springer, Cham.

[122] P. Kinghorn, L. Zhang and L. Shao. 2017. Deep learning based image description generation. In *Proceedings of International Joint Conference on Neural Networks*, pp.919-926. IEEE.

[123] P. Kinghorn, L. Zhang and L. Shao. 2018. A region-based image caption generator with refined descriptions. *Neurocomputing*, 272, pp.416-424.