Northumbria University
NEWCASTLE

EVOLVING DEEP NEURAL NETWORKS FOR IMAGE AND AUDIO CLASSIFICATION USING SWARM-BASED OPTIMIZATION

SAMUEL JONATHAN SLADE

PhD

2023

EVOLVING DEEP NEURAL NETWORKS FOR IMAGE AND AUDIO CLASSIFICATION USING SWARM-BASED OPTIMIZATION

SAMUEL JONATHAN SLADE

A thesis submitted in partial fulfilment of the requirements of the University of Northumbria at Newcastle for the degree of Doctor of Philosophy

Faculty of Engineering and Environment

October 2023

Abstract

Fine-tuning and designing the architecture of deep learning models to attain optimal performance is a intricate and time-intensive endeavor demanding expertise and multiple iterations. Streamlining this process holds promise, with swarm optimization algorithms emerging as a potential solution. However, these algorithms can be susceptible to local optima and are greatly influenced by their initial conditions, particularly in scenarios involving transfer learning and intricate loss functions, where research remains limited. Moreover, the choice of the most suitable optimization algorithm can be task-specific, necessitating tailored solutions.

To tackle these challenges, the research undertook the following approach:

1. Environmental Particle Swarm Optimisation (EnvPSO) was introduced, a variant of particle swarm optimization, which leverages gradients derived from the anticipated solution space to optimize a multi-stream Convolutional Neural Network (CNN) for human action recognition. It introduces a novel layer strip-back parameter for determining the number of frozen layers during transfer learning.

2. Neural Inference Search (NIS) was then developed, a swarm algorithm that employs neural networks to predict velocities, enhancing the optimization of deep learning segmentation models with multi-loss functions. This approach addresses memory constraints observed in EnvPSO while introducing diverse velocity calculations.

3. Finally Cluster Search Optimisation (CSO) was proposed, a novel approach utilizing clustering of particle positions and historical data for hyper-parameter and architecture search in deep learning models designed for audio emotion classification. This method addresses the issue of randomness in velocity predictions from NIS, incorporating dynamic convergence monitoring. It advances the optimization process by simultaneously optimizing hyper-parameters and architecture.

These research initiatives build upon one another, commencing with EnvPSO, which enhances particle swarm optimization for hyper-parameter fine-tuning, primarily in the context of Human Action Recognition (HAR), by predicting the topology of the search space through convolution. Subsequently, NIS embeds a representation of the search space within a neural network to alleviate memory constraints observed in EnvPSO, introducing a more diverse set of velocity calcula-

tions. The pinnacle of our work, CSO, focuses on mitigating randomness in velocity predictions from NIS by introducing a clustering-based approach and dynamic convergence monitoring. This approach represents a significant advancement, optimizing both hyper-parameters and architecture.

Our research findings reveal the increased performance of EnvPSO, especially when optimizing an ensemble of multiple CNN streams for still image human action recognition. EnvPSO outperforms state-of-the-art models by a margin of 1.49% on the Willow7 dataset and 1.4% on the BU101 dataset. Additionally, NIS enhances the performance of the Deeplabv3 model, achieving notable improvements of 4.2%, 0.28%, and 0.3% over the best-performing models on the MESSIDOR, Freiburg Forest, and CamVid datasets, respectively. Furthermore, CSO excels in optimizing CNN-Bidirectional Long Short-Term Memory (BiLSTM) architectures for audio emotion classification, surpassing existing work by substantial margins of 2.9%, 4.4%, and 17.1% on the Emo-DB, SAVEE, and TESS datasets, respectively.

Contents

List of Figures

List of Tables

Acknowledgements

I am deeply grateful to William Buchanan at RPPTV for providing me with the opportunity to embark on this research project. Without his support and encouragement, I would not have been able to pursue my passion for exploring the optimization of deep learning models. I am also thankful for the guidance and mentorship of my supervisor, Li Zhang, who has been an invaluable source of knowledge and support throughout my PhD journey. Her expertise and willingness to help have been instrumental in helping me navigate the challenges of this research project. In addition, I would like to express my appreciation to Kamlesh Mistry for his professionalism and support as my primary supervisor during the final year of my PhD. His insights and guidance have been invaluable in helping me bring this research to fruition.

I am also grateful to the Intensive Industrial Innovation Program, which has provided me with the opportunity to conduct this research in partnership with Northumbria University and RPPTV. Without this program, I would not have been able to pursue my PhD and contribute to the field of deep learning optimization. I am thankful for the resources, support, and opportunities provided through this program, and I am committed to using my skills and knowledge to continue making meaningful contributions in the future.

Declaration

I declare that the work contained in this thesis has not been submitted for any other award and that it is all my own work. I also confirm that this work fully acknowledges opinions, ideas and contributions from the work of others. The work was done in collaboration with the RPPTV Ltd.

Any ethical clearance for the research presented in this thesis has been approved. Approval has been sought and granted by the Researcher's submission to Northumbria University's Ethics Online System on 3rd September 2021.

I declare that the Word Count of this thesis is 36,527 words.

Name: Samuel Jonathan Slade

Date: 26 October 2023

Chapter 1

Introduction

In recent years, deep learning has undergone significant evolution, becoming a dominant paradigm in machine learning with applications spanning natural language processing, audio, and image classification. Convolutional Neural Networks (CNNs) have played a pivotal role in this evolution, continually advancing and giving rise to models like Generative Pre-trained Transformer (Generative Pre-trained Transformer (GPT)), techniques including Generative Adversarial Networks (Generative Adversarial Network (GAN)s), autoencoders, and various specialized CNN architectures.

These deep learning models find utility in diverse domains, powering applications such as chatbots (e.g., GPT3, ChatGPT), machine-generated art (e.g., Stable Diffusion, StyleGAN), and autonomous driving (e.g., DeeplabV3, UNet). Many of these models employ transfer learning, a practice that involves fine-tuning pre-trained models for specific tasks. However, adapting pre-trained models involves intricate challenges encompassing hyperparameter tuning, architectural customization, and network state preservation, often necessitating extensive experimentation.

The process of parameter and architecture search can improve productivity and reduce the trial and error involved in deploying deep learning models. In light of this, the present work focuses on optimizing the hyperparameters and architecture of deep learning models across multiple domains. Specifically, this research aims to:

- Propose a novel optimization technique for optimizing the hyperparameters of transfer learning-

based deep learning models for HAR.

- Develop a second optimization algorithm for hyperparameter optimization of multi-loss deep learning models for semantic segmentation.
- Propose a third optimization method that can simultaneously optimize both hyperparameters and architecture for deep learning models for audio emotion classification.

This study addresses the complexities of deep learning optimization through a systematic research progression spanning three works, each building upon its predecessor's insights:

1.0.1 Work 1: EnvPSO - Hyperparameter Optimization for Human Action Recognition

The first work, EnvPSO, introduces a novel variant of Particle Swarm Optimization (PSO) tailored for hyperparameter fine-tuning, primarily applied to HAR. Notably, EnvPSO endeavors to predict the search space topology by leveraging convolution with a Gaussian function to smooth previous evaluations, allowing for gradient extraction that is added to the social and cognitive terms in the original PSO. This approach revealed a heavy memory requirement for the surface model with increasing dimensions.

1.0.2 Work 2: NIS - Optimizing Semantic Segmentation

The second work, NIS, diverges by embedding an abstract search space representation within a neural network to avoid the memory issues discovered in EnvPSO. It replaces traditional social and cognitive velocity updates with three velocity calculations, two derived from an LSTM-CNN model (Maximum Standard Deviation Velocity Prediction and Local Best Velocity Prediction). These predictions adapt based on limited history data of each particle's position and fitness. The third velocity calculation employs a novel n-dimensional whirlpool search, introducing diversity into optimization. Although initial optimization randomness is observed, it diminishes with accumulating training data. NIS aligns with the overarching goal of deep learning model optimization, with a focus on hyperparameter refinement.

1.0.3 Work 3: CSO - Neural Architecture Search for Audio Emotion Recognition

The culmination of this progressive journey is CSO, the third work. CSO specifically addresses the randomness issue observed in velocity predictions encountered in NIS by implementing a distinct strategy. To mitigate this randomness, CSO introduces a clustering-based approach, utilizing particle and position history clusters as velocity targets, replacing the original social and cognitive factors of PSO as with NIS. This approach aims to provide greater predictability in the optimization process. Furthermore, CSO introduces a constant global best search mechanism facilitated by a dynamic G_{thresh} calculation. Convergence is assessed by monitoring the summed distance between the top 50% of the fittest clusters. When convergence is deemed good, CSO prioritizes an exploitation strategy, whereas in cases of poor convergence, it shifts toward exploration. Importantly, CSO advances the optimization journey by simultaneously optimizing both hyperparameters and architectural components, marking a significant advancement in the optimization approach.

In summary, this research represents a structured progression in deep learning optimization, with each work contributing to the overarching objective of streamlining parameter and architecture search. These works share a common goal: enhancing the efficiency and effectiveness of deep learning models across diverse domains, with each iteration building upon the insights and strategies of its predecessors.

1.1 Background

1.1.1 Swarm Optimised Multi-stream CNN for Human Action Recognition

HAR aims to identify human actions from visual data. A good HAR model is important in many applications, such as detecting falls, recognizing violent behaviours, identifying theft and many other day-to-day activities in various sectors such as healthcare and security. Such potential benefits have led to significant interest in developing robust, accurate, and efficient HAR models. Recent HAR-based solutions cover three main data domains: 1) still images, 2) Red, Blue, Green (RGB) video streams, and 3) Red, Blue, Green, Depth (RGB-D) video streams. In this respect, video action recognition has attracted significant attention, which takes both spatial and tempo-

ral information into account for action classification. However, the extraction of optical flow information requires substantial additional effort, with significant computational cost and complexity.

In comparison with video HAR, still image HAR has limited sources of information, i.e., only containing spatial information without any temporal cues. In addition, because of viewpoint variations, background clutter, rotations, occlusions, large intra-class and small inter-class variations, still image HAR is a challenging task. Owing to inefficiency in extracting low-level features directly from whole images caused by the aforementioned distracting factors (e.g., cluttered scenes and complex actions), diverse high-level cues, such as human body, body parts, poses, objects, and scene contexts, have been extracted for enhancing performance of still image HAR in existing studies [1]. Traditional non-Deep Learning based methods derive such high-level cues through multiple pre-processing steps, which lead to high computational costs. As an example [2] extracted a combination of human pose and context information for still image HAR, while pose primitive based HAR was performed by [3]. [4, 5] extracted human body, objects, and human-object interaction, while [6, 7] derived human body, objects, and scene contexts for HAR. In addition, body parts, objects, and human-object interaction were used in [8, 9, 10], whereas [11, 12, 13] adopted human body, body parts, objects, and scene contexts.

In the literature, such high-level cues are characterized by using various low-level features for HAR. As an example, Gupta et al. [7] employed Histogram of Oriented Gradients (HOG), Global Image Feature (GIST), shape context, colour histogram, and edge distance features, while Li and Ma [14] adopted Scale-Invariant Feature Transform (Scale-invariant Feature Transform (SIFT)), HOG, and GIST features. A number of existing studies used both HOG and SIFT features, e.g., [2, 5, 11, 12, 15, 16, 17, 18]. Other studies employed purely HOG features, e.g., [3, 4, 8, 9, 10, 13], while SIFT features were used purely in [6, 19, 20].

However, such feature descriptors are subject to various drawbacks. As an example, although SIFT is invariant to scaling, rotation, and illumination changes, it is sensitive to threshold settings [21]. Owing to feature matching, it is computationally costly with large memory consumption [22, 23]. In comparison with SIFT, HOG is not scale and rotation invariant [24, 25]. Its performance degrades when dealing with regions cluttered with noisy edges [26]. Despite the generation of a basic low-dimensional spatial representation of a given image [27], GIST shows significant

limitations in capturing fine image details [28, 29]. In short, the low-level features extracted by traditional feature descriptors are susceptible to various drawbacks, limiting their discriminative capabilities in tackling still image HAR.

In comparison with traditional methods, deep Convolutional Neural Networks (CNNs) conduct hierarchical layer-wise feature learning in an end-to-end fashion without the requirement of complex computing pipelines. Their feature detectors (i.e., the filters in CNNs) are trainable and highly adaptive. Since the filters learn to adapt to new tasks, CNNs are able to learn bespoke features from a given data set automatically. The machine learned features in earlier layers are similar to those (e.g., edges and corners) yielded by SIFT and HOG descriptors, while the final layers in CNNs are able to produce comparatively more abstract high-level representations (e.g., eyes and wheels). Their efficiency has been ascertained in various HAR tasks in recent years [30, 31, 32, 33, 34, 35, 36]. Besides that, CNNs yield superior performances over those of traditional methods in solving diverse image classification tasks [37, 38, 39, 40]. While these modern CNN based approaches improve upon the more traditional handcrafted features, they nearly all focus on full video datasets to take advantage of the additional information within them. Unfortunately these approaches are extremely computationally expensive. Not only that, it seems intuitive that video data should not strictly be necessary to identify actions as it is possible for a person to identify most action categories from photographs.

Transfer learning techniques are often employed throughout these CNN based approaches. However, a good balance between preserving the generalisability of the earlier layers and re-training the later layers on the new data set must be found. Moreover the initial training settings such as learning rate and batch size have a considerable impact on the final performance of the CNN. Optimising these hyper parameter settings is challenging, involving expert knowledge and iterative exploration. It presents a high knowledge barrier that needs focused attention and time. The manual fine-tuning process of hyper parameters is thus undesirable, which we aim to overcome by using automated search methods.

While the use of a simple grid search can be exploited to identify aforementioned hyper parameters, it is inefficient as many iterations are necessary. In comparison with such brute-force methods, swarm intelligence algorithms offer capabilities in solving diverse single and multi-objective optimisation problems. Such evolving search algorithms are motivated by observations of natural

behaviours, such as ant colonies, beehives, and bird flocks. In this respect, one of the most prevalent algorithms is PSO. The PSO algorithm is robust for tackling diverse optimisation problems with fast convergence rates. However, owing to reliance on a global best leader, the PSO model is prone to being trapped in local optima.

The following gaps seem prevalent from within the literature:

- Current techniques typically rely on more complex data such as video however, good classification performance should be achievable through still images alone.
- Transfer learning requires knowledge on which layers should be frozen and which should be retrained and is currently a manual process.
- PSO is susceptible to local optima traps due to reliance upon the personal best and global position velocity targets.

1.1.2 Neural Inference Search for Multi-loss Segmentation Models

Segmentation methods form a key component in many vision related tasks, e.g. automated medical diagnosis, autonomous driving and robotic navigation, all of which stand to revolutionise many industrial sectors. Poor segmentation performance causes incorrect medical diagnosis and dangerous course trajectories leaving apprehension in the uptake of these innovations. Consistently accurate segmentation algorithms are required to meet the stringent safety standards of these systems. Unfortunately, no existing methods can satisfy this requirement.

The existing segmentation techniques range from traditional methods such as k-means clustering [39] to modern Deep Learning methods [41, 42]. Convolutional Neural Networks (CNNs) appear to be very successful for tackling segmentation problems with multiple semantic classes and complex shapes. Critically, this success depends upon selecting an appropriate loss function that sensibly measures the error and choosing appropriate hyper parameters for the gradient optimisation algorithm and loss function. If these factors are not met, then CNN training is unlikely to produce well-generalised models.

There are many avenues of exploration for improving CNN performance including gradient optimisation algorithms, architecture design, data prepossessing, optimising hyper parameters, loss function selection and transfer learning. Hyper parameter optimisation involves determining ideal

input parameters for a given problem by searching a range of valid inputs and evaluating them. A naive approach to this is to manually try input values and see which configurations yield good results; this is time consuming however, also requiring problem specific knowledge to select sensible values. More sophisticated approaches avoid these problems by employing algorithms to conduct the search in an automated and guided way. Often these algorithms are based on mechanisms found in nature, such as the bio-luminescent communication of fireflies and the flocking behavior of birds; formalised in the Firefly Algorithm (FA) and PSO respectively.

Typically these algorithms consists of a number of agents that occupy some position in the search space which is mapped to a defined range of input values. The fitness of each position is evaluated then passed through some heuristically defined algorithm which determines how each agent should move within the search space; establishing values for the next evaluation. This process is then iterated over until a good solution is found. These heuristic movement algorithms aim to enable a wide exploration of the search space ensuring areas containing good solutions are not overlooked whilst also balancing exploitation of previous agent positions with good fitness by fine tuning their positions. This automated approach lends itself well to optimal hyper parameter discovery for many problems including CNNs.

A CNN has three key constituent parts, the architecture, loss function and optimiser. The architecture consists of an carefully arranged set of operations and adjustable weights which when presented with input values produce some arbitrary output. By presenting the network with a structured dataset that contains some inputs and a related desired output, it is possible to compare the resultant random outputs from the network with the desired outputs defined in the dataset using a loss function. This loss function serves to measure the error produced by the prediction from the current network state for the given data sample. An optimiser can use this error to calculate a suitable gradient which is back-propagated through the network, adjusting the weights thus reducing the prediction error in the new network state. Through multiple iterations over each sample in the dataset a CNN embeds an internal representation which can accurately predict many of the dataset samples and hopefully perform well on related unseen data. As such it is clear that the exact arrangement of the architecture, the aspects measured by the loss function, and the way that the weights are adjusted by the optimiser can all severely affect the final learned internal representation of the CNN.

The configuration of these three components are problem specific i.e., time/sequence problems vs image/spatial localisation problems, where the former requires an architecture that can remember sequences and the latter needs architecture focused on handling spatial/colour relationships. Furthermore, the way loss functions measure error is also domain dependent; this being most starkly highlighted in the difference between regression and classification, where desired outputs can be any real number for the former and the latter outputs are often represented with binary values. As such, regression error is better measured by mean squared error, whereas classification error is more appropriately captured by cross-entropy. Misapplications of these loss functions can therefore cause a model to fail completely despite all other aspects of the CNN being well configured. Pertinently, the way in which a loss function calculates error necessarily alters the information encoded into the consequent loss value.

The optimisation algorithm used for the gradient calculation is also an important aspect to achieving a well generalised model. Most current research use one of the following optimisers: Stochastic Gradient Decent (SGD), Root Mean Square Propagation (RMSProp) or Adaptive Moment Estimation (ADAM). Each of these methods come with initial hyper parameters which require sensible settings to perform optimally. This is true even for those using adaptive learning rates such as ADAM. By applying a search algorithm we can automate the process of finding optimal hyper parameters for the specific problem at hand. Some loss functions also have initial parameters that need careful configuration to perform well. PSO is a rational choice for this kind of optimisation due to it's fast convergence rates across many diverse optimisation problems. Despite this PSO tends to get stuck in local optima of the search space due to a reliance on a global leader. Many PSO variants have been proposed to remedy this by introducing techniques that enhance exploration and exploitation.

The following gaps in research are noted as follows:

- Highly Accurate Segmentation performance is critical to industry uptake and it is clear that Segmentation models would benefit from optimal parameter settings for training CNN models to tackle image segmentation tasks but there is not much work that focuses in this specific area.
- Since the initial settings of the optimiser and in some loss functions can severely impact seg-

mentation performance and are often selected manually which requires specific knowledge or time consuming manual searches.

- There are few studies on the impacts of compounded loss functions on segmentation classification performance of CNN based networks.

1.1.3 Cluster Search Optimisation of Deep Learning Models for Audio Emotion Classification

Emotion classification is desirable across many application domains such as healthcare, security and human computer interaction. By employing accurate emotion detection these industries can take advantage of a more nuanced form of human behaviour monitoring to sense a patients discomfort, predict hostile actions or provide more emotionally sensitive automated responses. While these capabilities are seemingly desirable, achieving them to the required standard is fraught with many compounding challenges. Human emotion is experienced and expressed with a notoriously wide variance of individual differences that are heavily influenced by complex compounding experiences from their social life, cultural expectations, and significant traumatic events. Through these experiences peoples learned expressions of emotion are influenced resulting in huge variation in body language, vocal patterns and turn of phrase making it hard to capture all the emotions someone might expresses through a single sensory modality. To further compound the issue there are subtle biological factors that increase expressive variance in movement and intonation. Assuming issues with expression variance are solved, further problems are inherent within the subjective frame from which they are interpreted, compiling yet more variability and bias that influence an observers final classification of a specific emotional expression.

Many feature-based and Deep Learning methods have been employed to tackle these problems, approaching them from a lexical [43], prosodic [44], audio [45], or visual [46] perspective, sometimes using a hybrid combination of these modalities. The more prominent methods typically attempt audio emotion classification using a Deep Learning neural network. This audio based approach is likely chosen due to the availability of professionally acted audio emotion datasets, which while at the risk of oversimplifying the emotional representations, do help reduce problems with poor recording and expressive variation. With the undeniable success of Deep Learning approaches across many classification tasks, it is unsurprising they are regularly used for audio

emotion recognition solutions [47, 48]. In particular, the ability for CNN filters to capture general patterns within a dataset that are robust to noise is well established and a suitable approach to tackle the wide variance in emotional expression.

As with all CNN based approaches, the hyper parameter selection and structure of the network are major factors that determine the generalisability of CNN inference on unseen data. Due to the time consuming nature and expert knowledge required to perform these searches, they are thus an ideal target for automation and yet is rarely applied.

The following gaps in research are noted as follows:

- Audio emotion classification has significant challenges in emotional variance
- The architecture and hyper parameter optimisation of CNNs for Audio emotion classification is not common in the literature.

1.2 Research Motivations and Questions

1.2.1 Swarm Optimised Multi-stream CNN for Human Action Recognition

With a lack of modern research on still image human action recognition it is useful to evaluate newer CNN based approaches typically found in the video and 3D data domains such as using CNN models, multiple ensemble CNN streams, transfer learning and Swarm Optimisation techniques. For this research the following research questions are identified to elucidate on these areas:

- Can greater classification performance be achieved on still image human action recognition tasks by employing CNN and Swarm Optimisation techniques?
- Can the number of re-trainable layers in the transfer learning process be encoded such that it can be searched by optimisation algorithm?
- What methods can be employed to improve upon the PSO algorithm to reduce it's tendency to fall into local optima?

1.2.2 Swarm Optimised Multi-loss Segmentation Models

Applying Swarm Optimisation techniques across multiple domains is useful in understanding the kinds of performance increase that can be achieved. Additionally, the contribution toward model accuracy is heavily dependent on loss function choice and could potentially be further enhanced by employing multiple loss functions simultaneously. Since optimisation techniques are often problem specific, it is useful to investigate different adaptations of PSO and ways that reduce tendencies to fall into local optima traps for a specific problem domain. To this end the following research questions are posed:

- Which loss functions perform well on CNN based segmentation tasks and what benefit if any, can be achieved through compounding them?
- Noting the flaws inherent in PSO, Is it possible to enhance or replace the personal and global best search mechanisms that typically lead to local optima traps?

1.2.3 Swarm Optimised CNN models for Audio Emotion Classification

In this research, we aim to optimize the structure and hyperparameters of CNNs for audio emotion classification. This task presents unique challenges compared to image-based classification tasks, so it is important to investigate search mechanisms that are specifically designed for audio CNN classifiers. Additionally, the structure of a CNN can significantly impact its performance, so we aim to explore methods of encoding model structure for use with automated optimization techniques.

This motivates the following research questions:

- Can the structure and hyperparameters of a deep learning network be optimized to improve performance on audio emotion classification problems?
- What types of deep learning architectures are suitable for the audio emotion recognition task, and how can their structure be encoded for optimization?
- What kinds of search mechanisms can effectively adapt between global and local searches during the optimization process, avoiding traps in local optima and premature convergence?

1.3 Research Contributions

1.3.1 Contributions from Swarm Optimised Multi-stream CNN for Human Action Recognition

The contributions for this research are summarised as follows.

1. A new EnvPSO variant is proposed for automating the fine-tuning process of CNNs. Specifically, EnvPSO introduces three mechanisms to overcome stagnation, i.e., (1) a new optimisation parameter named Layer Strip-back, which determines the number of layers to be re-trained in the VGG19 networks during transfer learning; (2) non-linear functions for search coefficient generation which enable the search process to achieve a better balance between diversification and intensification; (3) an additional environmental term embedding a Gaussian fitness surface prediction, which guides the search process towards optimal regions. These three mechanisms work cooperatively to overcome stagnation and automate hyper parameter fine-tuning of CNNs.

2. An ensemble model with three CNN-based streams is proposed for tackling HAR with still images. Specifically, the first stream employs a VGG19 network with EnvPSO-optimised hyper parameters, which uses the original images as its inputs. The second stream adopts another VGG19 network with EnvPSO-optimised hyper parameters, which uses semantic segmentation masks yielded by Mask Recurrent Convolutional Neural Network (R-CNN) as inputs. Such extracted saliency maps from Mask R-CNN provide another modality of inputs, which in particular offer better efficiency in representing various action classes (e.g., JugglingBalls, SoccerJuggling, and SkateBoarding) for recognition in Human-Object Interaction. The third stream fuses both VGG19 networks trained with raw images and segmented masks, respectively, by using a flatten and concatenation layer before the fully connected layers. This fused CNN stream helps induce diversity in the learned feature sets extracted from raw images and segmented sailent regions. The final classification result for each image is obtained by calculating the mean average of the results from the three streams. The EnvPSO-optimised VGG19 networks with a variety of learning configurations yield better diversity and complementary characteristics to enhance ensemble model performance, as demonstrated in a series of empirical studies.

1.3.2 Contributions from Neural Inference Search for Multi-loss Segmentation Models

The main contributions of this research are as follows.

1. A new multi-loss function is proposed to capture multiple error measurements with respect to semantic segmentation, enhancing feedback during CNN training. Specifically, the mean of Cross Entropy, Focal and Dice losses is exploited. Cross Entropy loss provides measurement of the overall pixel-wise class accuracy. Focal loss introduces the term α to apply a weighting factor to the classes that are present or non-present in the Ground Truth (GT) masks, in order to prevent over-fitting. Additionally, it adds a γ term to weight the contribution of well classified examples, contrasting the error contribution of poorly classified ones. The Dice loss implements a soft version of a mIoU based measurement, which can better highlight discrepancy in shape and consistency of classification. Combining these loss functions provides multiple aspects of the error signal, which can be exploited for devising better training strategies.

2. To automate hyper parameter tuning, NIS is proposed. In particular, NIS incorporates three novel behaviours: (1) *Maximised Standard Deviation Velocity Prediction*, which employs a LSTM-CNN network to predict the velocity vectors that increase standard deviation of the particle positions, ensuring agents do not search in the same local areas; (2) *Local Best Velocity Prediction*, which predicts the velocity vectors that point to local areas of best fitness; (3) *n-dimensional Whirlpool Search*, which produces the velocity vectors via a dot product of an n-dimensional rotation matrix at a defined angle with a vector pointing toward the global best position. As such, the particles are forced to explore multiple dimensions of the search space. The contribution of these three behaviours are adjusted at every iteration with a novel scheduling function, namely the Staged Discrete Adaptive Wave formula ensuring a better trade-off between exploration and exploitation. This is achieved by leveraging discrete stages factored with a slowly decreasing or increasing sinusoidal function, allowing each behaviour to be disabled or emphasized whilst approaching global optimality. This scheduling function and the above three novel behaviours work synchronously to automate hyper parameter selection and overcome stagnation.

1.3.3 Contributions from Cluster Search Optimisation of Deep Learning Models for Audio Emotion Classification

The key contributions of this research are:

1. A new swarm-based search mechanism that takes advantage of the the search particles fitness and position history by performing clustering to determine areas of promising exploration. The top 50% of the fittest cluster centers are isolated as potential search targets indicating general areas of good fitness which can be used to inform the velocity updates of swarm particles. This reduces the likelihood of particles being trapped in local optima whilst performing exploration giving a wider search area at points of poor convergence.

2. The Cluster Distance Improvement metric is proposed which monitors the convergence of the clusters in contrast to the worst/best convergence state and the current convergence state. This facilitates selecting the type of search a particle will perform allowing, with more particles being assigned to exploration during low convergence and the rest being assigned to a global exploitation search. As the clusters converge more particles are selected to perform a global search. This allows the search to adapt dynamically to the current success of the search. The intuition here is that if the top 50% of clusters centers are distant, then there is no consensus for which area might yield the globally optimum position so it is better to distribute the search around those clusters. When the clusters are converging then there is more consensus about where the global optimum position may lay and so more particles should be assigned to search the current best position.

1.3.4 List of Publications

1. S. Slade, L. Zhang, Y. Yu, and C. P. Lim, "An evolving ensemble model of multi-stream convolutional neural networks for human action recognition in still images," *Neural Computing and Applications*, vol. 34, no. 11, pp. 9205–9231, 2022

2. S. Slade, L. Zhang, H. Huang, H. Asadi, C. P. Lim, Y. Yu, D. Zhao, H. Lin, and R. Gao, "Neural inference search for multiloss segmentation models," *IEEE Transactions on Neural Networks and Learning Systems*, 2023

1.4 Structure of the Thesis

Chapter 2 begins with a detailed overview of the PSO algorithm followed by a review of PSO variants that have been implemented in the literature for many varying tasks. This is proceeded by a review of works related to the relevant task domains to be investigated; Still Image Human Action Recognition, Semantic Segmentation and Audio Emotion Classification.

Chapter 3 describes a newly proposed PSO algorithm that attempts to incorporate environmental information from the search space to improve the optimisation capabilities of Traditional PSO. This technique is then employed to optimise three constituent CNN networks independently to form a Multi-stream Ensemble of CNNs to tackle still image Human Action Recognition tasks. This model is evaluated on a number of well known datasets against other state of the art methods. The proposed model and EnvPSO algorithm have been successfully published to a peer reviewed journal [49].

Chapter 4 introduces another novel PSO-like search algorithm Called NIS. This algorithm replaces components of the logic-based implementation of PSO with a neural network that predicts custom velocity components used to update particle positions. NIS is applied to optimise a number of existing Deep Learning segmentation models and evaluated on three Segmentation data sets and compared with state of the art methods to ascertain the resultant classification performance improvements.

Chapter 5 details a third novel search algorithm proposal called CSO which is employed to optimise the hyper parameters and architecture of the proposed block-based Deep Learning models for audio emotion classification problems. A number of results are provided on three well known Audio Emotion datasets with comparisons of models proposed by relevant work in the field.

Chapter 6 provides a summary of the overall findings from each of the three studies, the potential directions of future research and applications of the research for wider industry for the following 3 to 5 years.

Chapter 2

Related Works

2.1 Particle Swarm Optimisation

PSO is a useful swarm intelligence algorithm for solving optimisation tasks, such as optimal hyper parameter selection in CNNs [51, 52, 39, 53, 54]. The algorithm works on the assumption that multiple agents can usually find a solution close to the global optima by emulating swarming behaviours found in nature. Its search process is as follows. Firstly, a swarm population in a given search space is initiated. Each particle moves around in the search space by following local and global optimal signals (see Equation 2.1). A fitness function is used to evaluate the current position of each particle. Specifically, a new velocity is calculated using the inertia weight component, as well as the social- and cognitive-inspired terms. In particular, the social-inspired term establishes a tendency of agents to cluster together to exploit some promising regions of the search space. The cognitive-inspired term promotes a tendency of agents to investigate other optimal areas identified by each particle on its own. To achieve swarming behaviours, each particle records its position with the best fitness score as p_{best_i}, while the best solution found by the overall swarm is recorded as g_{best}. Subsequently, the cognitive-based term is formed as $r_1 c_1 (p^t_{best_i} - x^t_i)$, which specifically influences the extent an agent conducts search near its own personal best solution. The social-based term is defined as $r_2 c_2 (g^t_{best} - x^t_i)$, which dictates the extent an agent is compelled to search near the current global best solution. These terms are formalised in Equation 2.1:

$$v^{t+1}_i = w v^t_i + r_1 c_1 (p^t_{best_i} - x^t_i) + r_2 c_2 (g^t_{best} - x^t_i) \tag{2.1}$$

where v_i^{t+1} is the velocity of the ith particle at the $(t+1)$th iteration and w is the inertia weight defining the contribution of the particle's previous velocity v_i^t toward a new one generated in the next iteration. The personal best solution of particle i at the tth iteration is denoted as $p_{best_i}^t$, while the global best solution of the overall swarm at the tth iteration is represented as g_{best}^t. Parameters r_1 and r_2 are random factors sampled from uniform distribution $U(0,1)$, while c_1 and c_2 are acceleration coefficients that determine the contribution of cognitive- and social-based terms, respectively. The next particle position x_i^{t+1} is then obtained using Equation 2.2 by summing the current particle position x_i^t and new velocity v_i^{t+1}. The pseudo-code of the original PSO model is illustrated in Algorithm 1.

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{2.2}$$

Algorithm 1 The original PSO algorithm

```
Initialise the swarm size n_p
Initialise a swarm of particles
Initialise the search parameters w, c_1, and c_2
while t < t_max do
    for each particle i = 1, ...., n_p do
        if f(x_i^t) > f(p_best_i) then
            p_best_i = x_i^t
        end if
        if f(x_i^t) > f(g_best) then
            g_best = x_i^t
        end if
    end for
    for each particle i = 1, ...., n_p do
        Update each particle velocity using Equation 2.1
        Update each particle position using Equation 2.2
    end for
end while
return g_best
```

2.2 Variants of Particle Swarm Optimisation

The original PSO algorithm shows efficient search capabilities in tackling diverse optimisations problems. Nonetheless, owing to the guidance of single global best leader, the swarm tends to converge prematurely, leading to local optima solutions [55, 56, 57]. As a result, many PSO variants have been proposed to tackle the challenges. As an example, Fielding and Zhang [58] proposed a

Swarm Optimised Dense Block Architecture Ensemble (SODBAE) integrated with a PSO variant for image classification. The model was capable of devising CNN architectures with residual connections and dense connectivity to increase network diversity. Specifically, it employed adaptive acceleration coefficients generated using cosine annealing mechanisms to overcome stagnation. Two weight inheritance learning mechanisms were introduced to enable the devised CNN layers to inherit weights from previously optimised ones based on their positions and parameter matrix sizes, with the attempt to reduce computational costs. The model outperformed other state-of-the-art methods as well as manually designed deep networks in a case study with the Canadian Institute for Advanced Research (CIFAR)-10 data set.

Nobile et al. [59] proposed a Fuzzy Self-Tuning Particle Swarm Optimisation (FST-PSO) algorithm. It provided fully automated parameter configurations to each particle by integrating fuzzy logic into the PSO algorithm. Two linguistic variables were used to establish fuzzy membership functions, i.e., one for determining the distance between the current particle and global best position as 'close', 'medium'red, or 'far', while another for measuring fitness improvement of a particle between two successive iterations as 'worse', 'same', or 'better'. These linguistic variables were used in conjunction with a list of rules associated with the inertia weight, social and cognitive search coefficients, and lower/upper clamping values for velocity. Through dynamically adjusting these fuzzy variables, each particle was capable of exploring more promising search regions autonomously. Evaluated on 12 benchmark functions, FST-PSO illustrated fast convergence speed, while maintaining competitive performance, as compared with classical search methods, such as Differential Evolution (DE) and Artificial Bee Colony (ABC).

While this innovative approach holds promise for addressing dynamic optimization problems, it lacks an in-depth discussion of the rationale behind the chosen linguistic variables and fuzzy rules. Moreover, the evaluation primarily relies on benchmark functions, leaving the algorithm's real-world applicability and practical effectiveness unexplored. Despite impressive results in convergence speed and competitiveness against classical methods, such as Differential Evolution and Artificial Bee Colony, further research and validation in practical applications are needed to establish FST-PSO's broader effectiveness and relevance.

Tan et al. [40] proposed a PSO variant to optimise hyper parameters of CNNs as well as cluster centroids of Fuzzy C-Means Clustering (FCM) for skin lesion segmentation. PSO was combined

with helix and DE search mechanisms to increase search diversification. A spiral function was used to assign search coefficients to these search operations, while Simulated Annealing (SA) and Levy flight were employed to increase intensification. The model then assigned these local and global search operations in a cascading manner. It started with SA-based local exploitation, and then switched to other search strategies such as PSO, helix or DE actions when the search process became stagnant. In this way, the swarm performed multiple search actions simultaneously in each iteration, in order to diversify the search process. The model was used to not only optimise hyper parameters of pixelwise CNNs, but also fine-tune the cluster centroids of FCM. The optimised CNN and FCM components formed two separate ensemble models for lesion segmentation. Evaluated using three skin lesion data sets, i.e., Dermofit Image Library, PH2, and ISIC 2017, the devised PSO-based ensemble model illustrated significant superiority over other clustering and deep networks in lesion segmentation.

The enhancements introduced to the PSO algorithm, including the SA and Helix search mechanisms, offer significant improvements over the standard PSO approach. While traditional PSO can struggle with local optima, the SA method, implemented in the first iteration, brings an element of randomness and exploration to the search process. The transition probability (p) and temperature control (T) parameters provide an effective means to balance local exploitation and global exploration. Additionally, the Helix search, integrated in later iterations, enables particles to simultaneously follow global and local swarm leaders, thanks to its dynamic and randomized nature. This helps overcome local optima issues and diversifies the search process. Both SA and Helix strategies significantly enhance the PSO algorithm's robustness and effectiveness in optimizing various objective functions.

Singh et al. [60] proposed a Multi-level Particle Swarm Optimisation (MPSO) model for optimisation of architectures and hyper parameters of CNNs. The proposed model exploited the concept of multiple populations. Specifically, the initial swarm at level one was used for CNN architecture generation (i.e., identification of the most optimal settings of convolutional, pooling, and fully connected layers), while multiple populations at level two were subsequently used to optimise hyper parameters (e.g., number of filters, filter size, and number of neurons) of each CNN from level one. An adaptive inertia weight implemented by a sigmoid function was leveraged to balance diversification and intensification. Evaluated using five well-known data sets, including Differ-

ential Evolution (MNIST), CIFAR-10, and CIFAR-100, the devised model with optimal hyper parameters produced an impressive performance.

The hybrid MPSO-CNN approach offers an intriguing solution for the automated optimization of CNN architectures and hyperparameters, featuring a multi-level swarm structure that efficiently explores the search space. Its hierarchical approach allows for the optimization of both high-level CNN architecture and fine-grained hyperparameters, providing flexibility. The use of a fitness function to evaluate configurations is commendable for accurately measuring performance. The controlled decrease in the inertia weight balances exploration and exploitation. The ability to execute the algorithm in parallel across multiple datasets showcases scalability. However, while the method imposes search space constraints, additional details on the termination criterion are needed. More extensive experimentation and comparative studies are warranted to validate its effectiveness, and insights into the algorithm's convergence behavior would enhance understanding. Overall, the hybrid MPSO-CNN approach holds promise for automating CNN design, but further validation and refinement are essential.

Bai et al. [61] proposed the Sine Mapped Dynamic Weighted particle Swarm Optimisation (SDWPSO) algorithm for optimising weights and biases of a Backpropagation Neural Network (BPNN) for reliability prediction in engineering problems. A new position updating operation was proposed, where dynamic weights were used to adjust the proportions of contributions of the current position, the new velocity and the global best solution for position updating. The sine map with an adaptive control factor was used to adjust the inertia weight. Evaluated using 14 benchmark functions and reliability prediction of turbocharger and industrial robot systems, the model outperformed Support Vector Machine (SVM) and Artificial Neural Network (ANN) methods significantly.

The presented research introduces a novel approach to reliability prediction, combining BPNN and a modified Particle Swarm Optimization (PSO) variant called SDWPSO. The study's strengths lie in its innovative fusion of established techniques and its empirical validation in both benchmark functions and real-world applications, demonstrating superior performance compared to other optimization methods. However, limitations include the lack of a comprehensive comparison with state-of-the-art methods in reliability prediction and a need for further validation on diverse datasets. While promising, the study would benefit from more detailed explanations of the

SDWPSO algorithm and its advantages over standard PSO. Overall, this research offers a valuable contribution to the field, but its broader relevance and potential advantages require further exploration and validation

Lan et al. [62] developed a Hierarchical Sorting Swarm Optimiser (HSSO) to solve large-scale optimisation problems. HSSO incorporated a new learning strategy to sort the particles into a hierarchical structure based on fitness scores. Specifically, the particles were recursively sorted into groups containing solutions with promising or poor fitness values. Promising particles were used in each subsequent recursion. This hierarchical structure employed elite solutions with promising fitness scores to update the velocities and positions of worst-performing particles. In addition, the personal best solution in the cognitive term was also replaced with those promising solutions in the hierarchical structure. The mean position of the overall swarm was adopted in the social term as opposed to a global best position. Using 39 generic benchmark test functions, HSSO showed improved exploration and exploitation capabilities against those of Social Learning Particle Swarm Optimisation (SL-PSO), Competitive Swarm Optimiser (CSO_2), Efficient Population Utilisation Strategy Particle Swarm Optimisation (EPUS-PSO), Dynamic Multi-Swarm Particle Swarm Optimisation (DMS-PSO), and Multilevel Cooperative Coevolution (MLCC).

The paper introduces the Hierarchical Sorting Swarm Optimizer (HSSO) as a novel approach to tackle large-scale optimization problems, addressing the limitations of Particle Swarm Optimization (PSO). HSSO's hierarchical learning mechanism, dividing particles into good and bad hierarchies for accelerated learning, offers an innovative solution for maintaining swarm diversity and improving algorithm efficiency. The paper provides a clear and detailed algorithm explanation, along with a thorough analysis of time complexity, which contributes to a comprehensive understanding of HSSO's design and computational characteristics. The extensive experimental evaluation demonstrates HSSO's promising performance across various benchmark functions and dimensions, suggesting its versatility and scalability. However, for a more balanced review, it's important to acknowledge the need for deeper exploration of the algorithm's limitations and potential areas of improvement, as well as the consideration of additional comparison algorithms, including more recent or specialized methods. Furthermore, further empirical validation and real-world applications would enhance the paper's claims regarding HSSO's effectiveness in complex optimization tasks.

Han et al. [63] developed an Adaptive Gradient Multi-Objective Particle Swarm Optimisation (AGMOPSO) algorithm to address slow convergence and sub-optimal performance inherent in multi-objective optimisation problems. The main goal of multi-objective optimisation is to achieve a weighting of contribution across all the evaluation functions (objectives) by optimising some target variables. This ideal weighting is known as the Pareto-optimal set. A stock ticker Multi-Objective Gradient (stocktickerMOG) method was devised to approximate the optimal Pareto set of solutions. A unique self-adaptive flight mechanism which affected both social and cognitive terms was introduced. To achieve this, a fixed sized archive was updated with the global best position, provided that it was not dominated by any current entries in the archive. During each PSO iteration, Multi-Objective Gradient (MOG) was used to obtain gradient information so that the archive entries can be incremented toward the Pareto-optimal set. A unique self-adaptive flight parameter was calculated based on the distance between the closest and furthest particles corresponding to the swarm leader as well as the distance between the current particle and global best solution. This flight parameter was applied to each particle differently depending on its dominance state with respect to the current entries in the archive. This allowed each particle to dynamically adapt the amount of contribution from the social and cognitive terms. Evaluated on a series of established multi-objective benchmark functions (ZDT [64] and DTLZ [65]) using the Inverted Generational Distance (IGD) and spacing metrics, AGMOPSO achieved better diversity and accuracy as compared with seven multi-objective PSO algorithms as well as Non-dominated Sorting Genetic Algorithm II (NSGA-II) and Strength Pareto Evolutionary Algorithm 2 (SPEA2).

The Adaptive Gradient-Based Multi-Objective Particle Swarm Optimization (AGMOPSO) algorithm, as introduced in the paper, offers a novel approach to multi-objective optimization (MOPSO) by combining gradient-based optimization with self-adaptive parameters. AGMOPSO's adaptability and ability to balance exploration and exploitation make it a noteworthy candidate in the field. However, to establish its competitiveness and real-world applicability, further research and experimentation across diverse problem domains are necessary. Additionally, the algorithm's sensitivity to hyperparameters requires investigation to optimize its performance effectively.

Cai et al. [66] combined PSO with Particle Swarm Optimisation with Density Peaks Clustering (PDPC) to address the limitations in manual selection of initial cluster centroids and the influence of a distance cut-off parameter required by Density Peaks Clustering (DPC). The distance cut-

off parameter was determined by calculating the Gaussian distances between all data points and taking the mean value of the maximum and minimum Gaussian distances. Initial cluster centroids of DPC were selected using PSO, where the inverse product of density and distance was used as the fitness function. Evaluated using nine UCI benchmark data sets, PDPC showed great superiority in solving cluster centroid selection, yielding promising accuracy, precision, and recall scores in contrast to several methods, including K-means clustering, Improved K-means clustering, original DPC and density peak K-mediods.

The paper introduces the PDPC (Particle Swarm Optimization-based Density Peak Clustering) algorithm, which attempts to address some of the limitations in traditional density peak clustering methods. The incorporation of PSO into the clustering process is a novel idea and has the potential to enhance clustering results. Additionally, the method for determining the parameter dc, which has been a challenge in density-based clustering, is a noteworthy contribution. The algorithm's time complexity analysis shows that it is competitive with existing clustering methods, making it a practical choice for large datasets. However, the paper could benefit from a more in-depth explanation of the rationale behind certain parameter choices, such as the linearly decreasing inertia weight and time-varying acceleration coefficients, to help readers understand their impact better.

A PSO model embedded with multi-surrogate schemes was proposed by Hu et al. [67] for feature selection. The Multi-surrogate Multi-swarm Binary Particle Swarm Optimization (DBPSO) algorithm offers an innovative approach to binary particle swarm optimization. While it introduces intriguing concepts like multi-surrogate modeling and dynamic transfer functions, its complexity and increased computational requirements could pose challenges for practical use. DBPSO demonstrates competitive performance on various benchmark functions, showcasing its potential in optimization tasks. However, its sensitivity to a predefined parameter and the need for parameter tuning could be a drawback in real-world applications.

A multi-objective PSO combined with reinforcement learning was used by Zhang et al. [68] for multi-UAV path planning, where reinforcement learning was adopted to select different exploitative, exploratory or hybrid search behaviours. The MCMOPSO-RL algorithm, as presented in this paper, offers a compelling fusion of reinforcement learning and particle swarm optimization techniques for addressing multi-objective optimization challenges. This innovative approach

treats particles as reinforcement learning agents, allowing them to dynamically adjust their exploration and exploitation strategies based on their performance. This adaptability, coupled with the incorporation of a hybrid update mode, holds great promise for effectively navigating intricate optimization landscapes.A notable advantage of this approach is its ability to maintain the same computational complexity as traditional MOPSO, ensuring computational efficiency. The introduction of individualized Q-tables for each particle to inform its decision-making process is a noteworthy and original feature. This concept has the potential to enhance the algorithm's capacity to discover superior solutions across a wide range of problem domains.

Entangled Q-bits in quantum computing were integrated with PSO in Vaze et al. [69] for solving hybrid, shifted and rotated numerical optimisation problems. This study presents an exploration of image segmentation using a novel Quantum-Behaved Particle Swarm Optimization (QEPSO) algorithm, comparing its performance to several other nature-inspired optimization algorithms. The research tackles the challenging problem of multi-thresholding, particularly focusing on Otsu's method, in which QEPSO exhibits competitive results when compared to other algorithms. The authors provide a comprehensive evaluation, including mean and standard deviation analyses, along with box plots to assess algorithm stability and performance. The study's strengths lie in its systematic approach, thorough experimentation, and the development of a promising QEPSO algorithm. However, the absence of a deeper discussion regarding the choice of test images and the specific motivations behind the comparisons with other algorithms leaves room for a more robust contextualization of the results. Furthermore, while the study highlights QEPSO's stability and efficacy, a more comprehensive analysis of computational efficiency and the algorithm's applicability to various image types would provide a more holistic assessment of its potential in image segmentation tasks.

PSO with population aggregation measurement was integrated with GANs for facial image generation in Zhang and Zhao [70], where the aggregation measurement was used to ensure swarm diversity. This work presented an intriguing approach that integrates GAN with PSO to enhance image generation. The concept of improving PSO for GANs is innovative and holds potential for advancing the field. The experiments conducted on CelebA and Cifar10 datasets demonstrate promising results, particularly in terms of reducing the Frechet Inception Distance (FID) metric, which indicates improved image quality. While some aspects, like the lack of statistical signifi-

cance testing and the need for a more detailed theoretical explanation, could benefit from further development, the overall direction and potential of the proposed method are commendable and warrant further exploration.

Another PSO variant with a residual group-based encoding strategy and uniformly selected neighbouring promising indicators was implemented by Lawrence et al. [38] for residual CNN generation. The proposed resPSOCNN method presents an intriguing approach to deep neural network architecture generation, addressing some key limitations in existing methods by introducing residual connections and expanding the search space for network settings into the optimisation process. The proposed encoding strategy and search mechanism offer innovative solutions to improve network depth and accuracy. However, while the results show promising performance gains in several experiments, it's essential to consider the potential challenges and trade-offs associated with the approach, such as increased computational complexity due to the expanded search space. Further investigation and comparison with other state-of-the-art methods on a broader range of datasets and tasks would be valuable to assess the full potential and generalizability of resPsoCnn.

The aforementioned PSO variants are useful for tackling issues of premature convergence of original PSO, where stagnation is often attributed to non-optimal exploration and exploitation of the search processes. Many studies change the flight characteristics of the cognitive and social terms. These changes are often applied to the velocity updating operation, as defined in Equation 2.1, which plays a significant role in determining a particle's search behaviour. The velocity updating operation often incorporates certain new factors to affect the social and/or cognitive terms. In some cases, the inertia weight is adjusted as well, in order to obtain a delicate control of the velocity scale applied to each particle in each iteration.

These Particle Swarm Optimization (PSO) variants offer innovative approaches to optimization, each addressing specific challenges or domains. AGMOPSO stands out for its unique combination of gradient-based optimization and adaptability in multi-objective problems, while the Hybrid MPSO-CNN approach features a hierarchical structure for CNN architecture and hyperparameter optimization. PDPC fuses density peak clustering with PSO, introducing a novel method for determining crucial parameters but requiring further validation. The other variants demonstrate PSO's versatility in various applications, such as feature selection, clustering, and architecture design, yet often lack extensive real-world validation, necessitating further exploration of their

practical effectiveness and limitations.

2.3 Human Action Recognition

HAR has gained increasing research attention, owing to its broad range of real-life deployments such as healthcare, security, and surveillance [30]. As an example, Sharma et al. [71] presented an Expanded Parts Model (EPM) to tackle HAR problems. The model selected discriminative part templates with an associated scale space location and scored them using a novel SVM-like classifier. A unique scoring function was proposed, which promoted learning diverse spatial and descriptive image patches to best represent the action. The EPM model, when visualised, showed an interesting collage of class relevant image patches spatially overlayed atop the original image with non-relevant parts of the images remaining black. This gave an idea of how the classifier matched parts with relevant aspects in an image to optimise accuracy. Evaluated using the Stanford40 and Human Attributes (HAT) data sets, the EPM model in combination with Visual Geometry Group 16 model (VGG16)-based feature extraction achieved superior MAP scores as compared with nine other methods, including Spatial Pyramid Matching. Nevertheless, while the method exhibits promise by its adaptability to diverse datasets and its potential to complement deep learning methodologies, a more extensive comparative analysis against contemporary state-of-the-art techniques would enhance its academic rigor. The increase in training times for SVMs poses is notable and might benefit from strategies aimed at mitigating the computational overhead.

Zhang et al. [72] presented a part-based method called Minimum Annotation Effort (MAE) to handle still image-based HAR tasks. The model included two main components, i.e., delineation of the 'action mask' and a unique feature representation for action classification. Delineating the action mask required two steps, i.e., object parts generation and action mask discovery. To address the first issue, bounding-box based object proposals were obtained using unsupervised selective search and passed through a VGG16 network. A multi-max pooling technique was applied to the outputs from the last convolutional layer of the VGG16 network to yield object parts. To retrieve the action mask, an energy minimisation problem on a Markov random field was formulated. The solution produced a shared global parts model, a part model for each class and action-masks for each image. In addition, feature representation was conducted by applying Product Quantisation to the initial object proposals that had sufficient overlaps with the action mask. These formed the in-

puts to a one-vs-all linear SVM classifier for action classification. Evaluated using benchmark still image data sets (such as PASCAL VOC 2012, Stanford40, and Willow7), the model outperformed existing methods such as Regularised Max Pooling (RMP), Object Bank, Locality-constrained Linear Coding (LLC), and EPM.

This method for presents an innovative approach to address the challenging task without relying on human bounding-box annotations. By decomposing the problem into two subproblems—action mask delineation and feature representation—the authors introduce a part-based representation that effectively captures human-object interactions, demonstrating its superiority over existing state-of-the-art methods on the PASCAL VOC 2012 action dataset. The iterative learning of action masks, along with the fusion of object-based representations through product quantization, provides a robust framework for recognizing actions. While the method yields impressive results on various benchmark datasets, it still faces challenges in dealing with ambiguous action labels. Overall, this research offers a valuable contribution to the field of action recognition, emphasizing the importance of object context in understanding human actions.

Wang and Wang [73] proposed a Joint learning Hierarchical Spatial Sum Product Network (JHS-SPN) for HAR tasks. A novel feature representation scheme was introduced. Image patches were sequentially extracted from the images. Action features were established by extracting CNN features from these sampled image patches. The feature vectors were clustered and used to fine-tune a CNN model. Multiple SVMs were trained on these feature clusters to produce part activation vectors. JHS-SPN altered the original Sum Product Network (SPN) model by introducing hierarchical partitioning. It learned optimal channels by dividing an image and capturing deformable spatial relationships between object parts. Part activation vectors and spatial relationships were extracted from each image subdivision, in order to reduce the computation complexity. Based on the Willow7 action data set, JHS-SPN produced superior MAP scores as compared with those from EPM, Discriminative spatial Saliency (Dsal), and the interaction pairs method. Evaluated on the Stanford40 data set, JHS-SPN outperformed EPM, LLC, Object Bank, and Spatial Pyramid Matching methods.

This work presents an innovative approach to action recognition, particularly in handling spatial relationships between parts. The hierarchical partitioning of images into sub-images and the modeling of local pairwise spatial relationships within these sub-images offer a promising way to cap-

ture complex configurations of human body parts in different action classes. The experiments on the Willow 7 and Stanford 40 datasets demonstrate competitive performance compared to existing methods. However, while the HS-SPN shows advantages in certain scenarios, it may face limitations when applied to actions with more diverse and intricate spatial relationships. Additionally, the training process, involving multiple iterations for part discovery, might be computationally intensive and might require extensive hyperparameter tuning. Overall, the HS-SPN presents a novel perspective on action recognition, but its practicality and robustness in handling a wide range of action classes and variations remain areas for further investigation.

Li et al. [74] proposed attention-based transfer learning for image-video adaptation for both HAR and human interaction recognition. A new Human Interaction Image (HII) data set was introduced. Specifically, the method employed class-discriminative spatial attention maps and a Siamese EnergyNet structure for video classification. Class-discriminative spatial attention maps were generated for each video frame using a pre-trained CNN integrated with Gradient-weighted Class Activation Mapping (Grad-CAM). These attention maps were subsequently combined with word embedding vectors derived from the class description. The combined feature vectors were used as inputs to the Siamese EnergyNet. This network comprised four parallel dense CNN layers, which was optimised using both energy loss and triplet loss functions. To boost training efficiency, these four parallel dense CNN layers adopted four different types of inputs, i.e., a ground truth label, a false example, a positive example from a different video clip and an incorrect example with minor differences from the ground truth. The model produced competitive MAP scores on the UCF101 data set against 11 other state-of-the-art methods. Its superior performance on the HII data set was also demonstrated. This method addresses a significant challenge in action recognition by introducing a novel approach to domain adaptation from web images to video. The utilization of class-discriminative spatial attention maps generated by Grad-CAM is a noteworthy contribution, allowing the adaptation of pre-trained CNNs for video classification. The paper presents a well-structured study, exploring both unsupervised and supervised domain adaptation scenarios, and conducts experiments on relevant datasets, demonstrating competitive results, especially when the labeled video data is limited. However, while the approach showcases promising results, there is a need for a more in-depth discussion of its limitations and potential failure cases. Additionally, the paper could benefit from providing more insights into the computational requirements and scala-

bility of the proposed method. Nonetheless, the research offers valuable insights and an effective approach to address a real-world problem in action recognition domain adaptation.

Safaei [75] proposed an ensemble method combining Spatio-Temporal Convolutional Neural Network (STCNN) and Zero-shot Tensor Decomposition (ZTD) to solve Still Image HAR problems. A novel strategy for generating spatio-temporal features along with STCNN and ZDT models was formulated. A new large scale image data set, namely UCF-Star, was also introduced. The spatio-temporal feature extraction process was unique as the generated temporal information from still images did not inherently exist. To achieve this, the optical flow vectors across several frames were clustered into quantised groups. Taking an image and its corresponding motion clusters as labels, a CNN was optimised using a spatial loss function to classify the regions as probability distribution over the motion vectors. In effect, this produced vertical and horizontal predicted optical flow information. A 3-channel tensor was produced for each image by combining these optical flow predictions with a saliency map derived from a bottom-up ranking method. These spatio-temporal features were used to fine-tune a VGG16 network pre-trained on ImageNet forming the first part of the ensemble model, i.e., STCNN. The second part of the ensemble model was based on ZTD. It conducted HAR by forming action prototypes, applying Tucker decomposition and then performing classification by calculating the set of joint probability distributions between class labels and each test image. The STCNN and ZTD models were combined using Multiple Linear Regression (MLR). Evaluated using the UCFSI-101 (i.e., extracted frames from UCF101), Willow7, Stanford40, WIDER, and UCF-Star data sets, the MLR ensemble method integrating STCNN and ZTD outperformed Object Bank, LLC, and Multi Region CNN methods, significantly. The paper offers a thorough investigation into still image action recognition, particularly the integration of spatial and temporal information, which is a crucial aspect of the problem. The experiments and results provide valuable insights into the strengths of different representations and the advantages of the proposed STCNN model. However, some sections of the paper could benefit from more detailed explanations and clarity, particularly in describing the limitations of the approach and the rationale behind certain design choices. Nonetheless, the paper makes a significant contribution to the field by highlighting the importance of considering both appearance and motion cues in still image action recognition, and the reported performance improvements are noteworthy, especially in the context of body-motion actions.

Yu et al. [76] proposed Non-sequential Convolutional Neural Network (NCNN) to solve Still Image HAR tasks. The NCNN model added multiple parallel branches of convolutional layers to a pre-trained CNN, in order to separately learn spatial and channel-wise features. An end-to-end trainable ensemble of CNN models incorporating NCNN blocks was formed. This ensemble model was compared against traditional ensemble methods (e.g., majority voting, averaging, and weighted averaging) using three different voting strategies (e.g., tuning weight, hard, and soft voting schemes). An ensemble of VGG16, VGG19, ResNet50, VGG16_NCNN, VGG19_NCNN, and ResNet50_NCNN using the tuning weight voting scheme achieved the best performance on the Willow7 data set. This work provides valuable insights into their performance across different datasets. It is notable that the study demonstrates clear improvements over nonensemble models, showcasing the potential of these techniques in addressing complex recognition tasks.

Liu et al. [77] proposed loss guided activation for still image HAR tasks. A novel human mask loss was introduced for optimising a unique human localisation stream. This stream along with another action classification stream was appended to the final convolutional layer of an Inception-ResNet-v2 network. Such strategies enabled joint predictions on both human action classes and a human localisation heatmap, forcing the learned feature representations to focus on the most action-relevant human subjects in the image. The method showed great superiority over 7 other state-of-the-art methods on the MPII and Stanford40 data sets. This presents an intriguing approach to tackle the challenging problem of action recognition in still images by leveraging a human-mask loss guided activation network. The proposed architecture is well-structured and empirically demonstrated to outperform existing methods on benchmark datasets, showcasing the effectiveness of the human-mask loss in directing the network's attention towards humans in complex scenes. The visualization of activation maps adds valuable insights into the network's decision-making process. However, it is worth noting that the approach has limitations, particularly in scenarios where misleading or occluding objects dominate the scene or when there are indirect interactions with objects. These limitations highlight the ongoing challenges in action recognition, and future work should aim to address them, possibly by incorporating human-object interaction models.

Yan et al. [78] proposed multi-branch attention networks for still image HAR problems. The method leveraged the idea of human attention as applied to viewing images. To achieve this, a

soft attention mechanism was devised by adding two branches to a VGG16 model, one branch to capture scene level attention while another to handle region-level attention. A two-step alternating optimisation technique was introduced. The classification and region-level attention parameters were first trained before training those associated with scene-level attention. The method showed great performance on the PASCAL VOC 2012 and Stanford40 data sets. The incorporation of three branches, each specializing in target person region classification, region-level attention, and scene-level attention, demonstrates a thoughtful effort to capture both fine-grained contextual information and overall scene context. The utilization of alternating optimization during training is a clever strategy to manage the complex parameter space effectively. However, it's important to consider that the approach relies on the availability of bounding boxes in the initial training phase, which can be a limitation in real-world applications. Furthermore, the method's complexity may pose challenges in terms of computational resources and training time. Nonetheless, the authors' efforts to address these limitations in their experimental setting 2, where bounding boxes are not used during training, yield competitive results. Additionally, a more in-depth discussion of the computational demands and scalability of this approach would provide valuable insights for potential users.

2.4 Semantic Segmentation

Many research studies on semantic segmentation methods are available in the literature, e.g. [79, 80, 81], for autonomous driving and robotic navigation. Zhang et al. [82] investigated the effect of early and late fusion of multi-modal deep learning architectures to solve semantic segmentation for automated robotic navigation. They proposed a Complex Modality network (CMnet) which utilised a late fusion of two processing streams to handle both RGB images and supplementary features such as near-infrared images. The model was evaluated on the Freiburg Forest dataset and a newly introduced POLAR dataset. The results revealed that utilizing RGB images led to comparatively better outcomes on single stream architectures as compared with those from other sensory data. Further performance gains can be derived by using RGB with other data as multi-modal inputs for late fusion dual stream architectures.

Saire et al. [83] explored multi-task learning with deep learning for semantic segmentation by introducing three related auxiliary tasks to be solved simultaneously by a single CNN model. A

standard encoder-decoder network was adopted with predictive branches, one for the main segmentation task, while the others for distinct contour prediction tasks. Each branch used a combination of Cross Entropy and soft Intersection over Union (IoU) loss, which were weighted to control the contribution of the error signals. This multi-task learning approach was tested on various encoder-decoder architectures. The results revealed that those modified to solve the auxiliary tasks illustrated a better class segmentation performance. The authors present compelling evidence that MTL, especially when incorporating contour-based auxiliary tasks, enhances the precision of activation maps, mitigates overfitting, and fosters feature space robustification. However, it's worth noting that the increased training time associated with MTL, as indicated in their efficiency comparison, may pose practical challenges in real-world applications where computational resources are constrained. Additionally, while the study focuses on improving semantic segmentation, further exploration of the generalizability of these findings to other computer vision tasks could provide a more comprehensive understanding of the broader applicability of MTL in convolutional neural networks.

Islam et al. [84] proposed Gated Feedback Refinement Network (G-FRNet) for dense image labeling. Processing branches were inserted between each spatially distinct encoding and decoding layers. Each branch contained a Gate block linked to a refinement block, providing spatially relevant features for the decoder stages. Each stage was supervised by spatially matching GT images. The method was compared with a number of state-of-the-art CNN-based segmentation architectures. Improved IoU scores on several benchmark datasets were produced by the proposed method. This work reveals an innovative approach to dense image labeling, showcasing its effectiveness on multiple benchmark datasets. The introduction of gate units in the refinement process represents a novel contribution, allowing for the modulation of information passed through skip connections. This gating mechanism effectively filters out categorical ambiguity and enhances the network's ability to capture finer details and complex contextual patterns within images. The empirical results demonstrate that G-FRNet consistently outperforms several state-of-the-art semantic segmentation methods, achieving competitive mean Intersection over Union (IoU) scores with a significantly smaller number of parameters. While G-FRNet demonstrates remarkable performance improvements, it's important to note that its complex architecture might incur additional computational costs during training and inference, which could be a consideration for real-time

applications. Nonetheless, the incorporation of gate units to control information flow showcases a promising direction in the ongoing quest to improve dense image labeling accuracy.

Jègou et al.[85] proposed a new architecture, i.e. FC-DenseNet, by appending DenseNet with upsampling blocks for semantic segmentation. The work re-designed dense blocks so that they could perform the downsampling, bottleneck and upsampling operations in an encoder-decoder architecture. The key component was removing skip connections between upsampling blocks, reducing feature explosion during upsampling, while maintaining a good feature representation. Improved Global Accuracy and mIoU metrics were achieved on the CamVid dataset, as compared with those from eight existing state-of-the-art methods. This architecture builds upon the DenseNet's feature reuse concept and integrates it into an upsampling path. This design choice allows for the recovery of spatial information without suffering from a feature map explosion. The experiments conducted on the CamVid and Gatech datasets demonstrate significant improvements in semantic segmentation performance compared to other methods. While the results are indeed impressive, it's worth considering the computational complexity of training such deep networks and the potential need for extensive data augmentation and fine-tuning. Additionally, the authors acknowledge the absence of pretraining on large datasets, which could further enhance the model's performance. Nonetheless, the FC-DenseNet architecture presents an innovative and effective approach to semantic segmentation tasks, but practical considerations like computational resources and data requirements should be kept in mind when considering its adoption in real-world applications.

Badrinarayanan et al. [86] proposed SegNet for semantic segmentation based on the VGG16 architecture. They introduced a novel upsampling approach that used the pooling indices of the maxpooling steps from a VGG16 encoder to perform non-linear upsampling, removing the burden of learning to upsample. This method showed the best performance as compared with those of other well-known segmentation methods on the SUNRGB-D and CamVid datasets. The authors provide a detailed analysis of various decoder variants within the SegNet framework and compare them against other well-established segmentation architectures such as FCN, DeepLab, and DeconvNet. The empirical evaluation on road scene and indoor scene segmentation datasets reveals that SegNet achieves competitive performance in terms of global accuracy, class average accuracy, and mean intersection over union (mIoU). Notably, SegNet stands out as a memory-efficient

solution, particularly when compared to FCN, due to its use of max-pooling indices for upsampling, instead of storing full encoder feature maps. While the study's experimental findings are valuable, it could benefit from a deeper exploration of the trade-offs between memory efficiency and segmentation accuracy, shedding more light on scenarios where SegNet's advantages become more pronounced. Additionally, a more extensive comparison with recent state-of-the-art segmentation architectures and datasets could provide a more comprehensive assessment of SegNet's performance.

Besides deep learning methods, traditional heuristic methods were also used for image segmentation. Abdulla et al. [87] proposed a method to tackle optical disc segmentation by applying morphological operations to remove vasculature in the images, and enhancing the optical disc center via a Hough Transform. It also used the grow-cut algorithm to determine the optical disc boundary. The utilization of morphological operations, circular Hough transform, and the grow-cut algorithm showcases a novel combination of techniques. The method's robustness and commendable performance across various publicly available retinal image databases are certainly noteworthy. Achieving 100% OD detection accuracy in several datasets is a remarkable feat and highlights the algorithm's potential. Additionally, the methodology demonstrates robustness in handling challenging scenarios such as images with noise, poor illumination, and the presence of pathological structures. However, it's important to acknowledge that there are some cases where the algorithm fails to provide accurate segmentation, particularly in images with strong artifacts or certain pathologies. Therefore, while this approach is promising and competitive with existing methods, it may benefit from further refinement to address these challenging cases and enhance its overall applicability in clinical settings.

Morales et al. [88] proposed a solution for optical disk segmentation using the Principal Component Analysis (PCA) to produce a greyscale image with enhanced vasculature and optical disc. An inpainting technique was then applied to remove the vasculature, which was followed by a stochastic watershed algorithm and a stratified watershed algorithm. Finally, a geodesic transformation was used on the resultant watershed regions to determine which regions formed part of the optical disc. A comparison with existing studies indicated better results in several metrics such as Dice, mIoU, Global Accuracy, and Mean Absolute Difference. The presented method exhibits competitive performance, as demonstrated on the DRIONS database, with Jaccard and Dice co-

efficients of 0.8424 and 0.9084, respectively, accuracy of 0.9934, true positive fraction of 0.9281, false positive fraction of 0.0040, and mean absolute distance of 2.4945. While it successfully addresses challenges such as handling variations in image quality and contrast, the method is benchmarked against the second observer's annotations and an alternative marker-controlled watershed algorithm. The comparative results indicate that the proposed approach offers a strong balance between accuracy and robustness, showcasing its effectiveness in OD segmentation across multiple datasets. However, it is essential to consider other databases and assess the method's performance against a wider range of existing techniques to provide a more comprehensive evaluation.

2.5 Audio Emotion Recognition

To flesh out some key concepts and relevant background information for this work, a number of studies on audio emotion classification and their findings are introduced in the proceeding discussion.

One approach to audio emotion classification by Parry et. al. [89] looked at the effects of cross-corpus training as a way to improve classification performance. CNN, LSTM and CNN-LSTM models were trained using train data splits from either one or the combined collection of 6 key audio emotion datasets. Since the ground truth labels of each dataset vary, they are collected into three categories: positive, neutral and negative. The performance of these multi-corpus trained models were compared against the models trained on a single dataset, revealing that training with multi-corpus data led to better classification performance. Further analysis using t-SNE indicated that recurrent neural network architectures typically overfit training data producing poorer performance than CNN based structures. The selection of LSTM, CNN, and CNN-LSTM architectures is well-justified, as they are commonly employed in this domain. The experimental results clearly demonstrate the limitations of single-corpus training, showcasing the challenges of model generalization to out-of-domain data, which is a prevalent issue in real-world applications. The t-SNE visualizations add depth to the analysis, shedding light on the differences in learned representations among the models. However, while the paper convincingly highlights the superior performance of CNN-based models in cross-corpus settings, it leaves room for further exploration into the underlying reasons behind LSTM's poor performance. A more detailed examination of the vanishing gradient problem and its specific effects on LSTM's ability to generalize across different corpora

could provide a deeper understanding of the observed phenomena. Overall, the paper makes a significant contribution to the field of emotion recognition in speech, emphasizing the importance of model architecture and training paradigms.

Li et. al. [90] performed cross-lingual speech emotion recognition with a novel unsupervised cross-lingual Neural Network with Pseudo Multilabel (NNPM). They employ a Siamese Network with Self Attention (SNSA) that classifies labeled English audio data samples whilst outputting attention layer weights into an Dynamic External Memory vector. The network also outputs the features from the unlabeled audio data of a different language as a query vector. The external memory and query vectors are compared using a similarity metric that ultimately provides pseudo multi-label targets for the unlabeled data. The NNPM network is trained on labeled data and pseudo labeled data simultaneously. This method is tested against three state of the art models showing improvement in classification compared to the top performing model. While the introduction of Dynamic External Memory for handling source domain knowledge is a commendable innovation, the paper lacks a thorough exploration of the limitations and potential challenges of this approach. Additionally, the motivation behind the architectural choices in the SNSA and how these components improve feature extraction could be better elucidated. The pseudo-labeling technique for assigning labels to target domain data is a promising solution to the lack of target domain labels, yet the paper does not delve into the sensitivity of this approach to hyperparameter choices. Overall, while the proposed methodology is creative and effective, a more in-depth discussion of its nuances and potential pitfalls would enhance the paper's contribution to the field.

Kanwal et. al.[91] proposed DB-SCAN for Genetic Algorithms (DGA), a Genetic algorithm (GA) using Density-Based Spatial Clustering of Applications with Noise (DB-SCAN) for audio emotion recognition. This work replaces the random selection step of GA with DB-SCAN clustering of the evaluated solutions such that generated individuals not strongly belonging to any cluster are removed. By applying DGA to OPENSMILE features extracted from audio emotion data they were able to significantly improve the classification accuracy of SVM classification across 3 well known datasets. The proposed methodology is comprehensive and well-documented, and the experimental results demonstrate significant improvements in recognition accuracy, particularly in speaker-dependent scenarios. The use of density-based clustering for parent selection in the genetic algorithm is a noteworthy innovation, as it helps maintain population diversity and

contributes to the algorithm's success. However, while the paper provides compelling evidence of the algorithm's effectiveness, it lacks a deeper exploration of the algorithm's computational complexity, making it challenging to assess its scalability to larger datasets.

Hussain et. al. [92] introduced Wavegram and Wavegram-logmel features extraction of multi modal convolutional neural networks to solve audio visual emotion recognition problems. They pre-train 2 networks, the first is an audio CNN network that combines logmel based feature maps and Wavegram predictions to produce classification predictions. The second is a visual CNN network that perform facial land mark detection, bounding box regression, and class predictions. The predictions of these two networks are combined with a simple fusion layer using a standard fully connected block that is trained independently. They test the audio model and video model separately along with the combined audio-visual model showing classification improvements for all three modalities. Of these three, the audio model performed best for emotion classification. The use of Wavegram and Wavegram-Logmel features for audio representation, coupled with a one-dimensional CNN architecture with dilations, demonstrates an effective way to capture frequency information from speech waveforms. The incorporation of bounding box regression for quantifying visual transformations adds valuable fine-grained information. Furthermore, the performance evaluation on the SAVEE dataset, with comparisons against baseline models and state-of-the-art frameworks, suggests promising results. However, addressing potential challenges, such as real-world noise and variations in emotion expression, would enhance the system's practical applicability.

Haider et. al. [93] proposed Active Feature Selection (AFS) with SVM for audio emotion classification. Using Self Organising Maps to perform clustering on emobase and eGemaps features extracted using the openSMILE toolkit. Using a leave one out cross validation approach, each cluster is evaluated by training an SVM to determine which features provide the highest classification accuracy. The show AFS performs well on three audio emotion datasets when compared against three other feature selection techniques. The confirm that good classification accuracy on audio emotion datasets can be achieved with a selected reduced feature set. The selection of Unweighted Average Recall (UAR) as the evaluation metric is appropriate for addressing class imbalance concerns. The research demonstrates the potential of feature selection techniques, such as Fisher and ReliefF, to significantly reduce the dimensionality of feature sets while maintaining or even im-

proving classification performance. The novel Adaptive Feature Selection (AFS) method's ability to provide competitive results with fewer features is a notable contribution, although further exploration of combining features from different clusters is recommended. However, the study lacks a deeper exploration of the interpretability of the selected features and their relevance to emotion recognition, which could provide valuable insights into the underlying mechanisms. Additionally, the study could benefit from a more in-depth discussion of the computational efficiency of these methods, as reducing feature dimensionality should also consider real-time application feasibility in low-resource systems.

Chapter 3

Proposed EnvPSO Optimised Multi-stream CNN for Human Action Recognition

3.1 Introduction

In this research, we propose a new PSO variant for hyper parameter fine-tuning in a transfer learning setting for undertaking HAR tasks with still images. Denoted as EnvPSO, this PSO variant incorporates Gaussian fitness surface prediction and adaptive coefficients to accelerate convergence. It is used to optimise the hyper parameters of VGG19 deep networks, including the number of re-trained layers in the transfer learning process (denoted as Layer Strip-back), batch size, and learning rate. Moreover, motivated by the well-known two-stream CNN architecture proposed by Simonyan and Zisserman [31] where features extracted from multimodal inputs are used for action classification, we design a three-stream based ensemble model with multiple optimised VGG19 networks using EnvPSO for tackling HAR problems. Specifically, in the first stream, we employ an optimised VGG19 network with raw images as inputs. In the second stream, Mask R-CNN is first adopted to generate semantic segmentation masks for each input image. The yielded saliency maps are subsequently used as inputs for another optimised VGG19 network for action recognition. In the third stream, a fusion network is constructed by concatenating two VGG19 networks

configured in the same manner as the first and second streams. Each of the three CNN streams, denoted as Streams 1, 2, and 3, is optimised independently by the EnvPSO algorithm to identify optimal settings for the learning rate, batch size, and Layer Strip-back hyper parameters.

These three streams are then combined in an ensemble manner. The final classification results are obtained by taking the average of probabilistic class predictions from the three CNN streams. In other words, the class predictions generated by the optimised CNN streams are summed and divided by the number of streams to produce an average prediction for each input image. A high-level depiction of the proposed EnvPSO-optimised CNN ensemble model is provided in Figure 1.

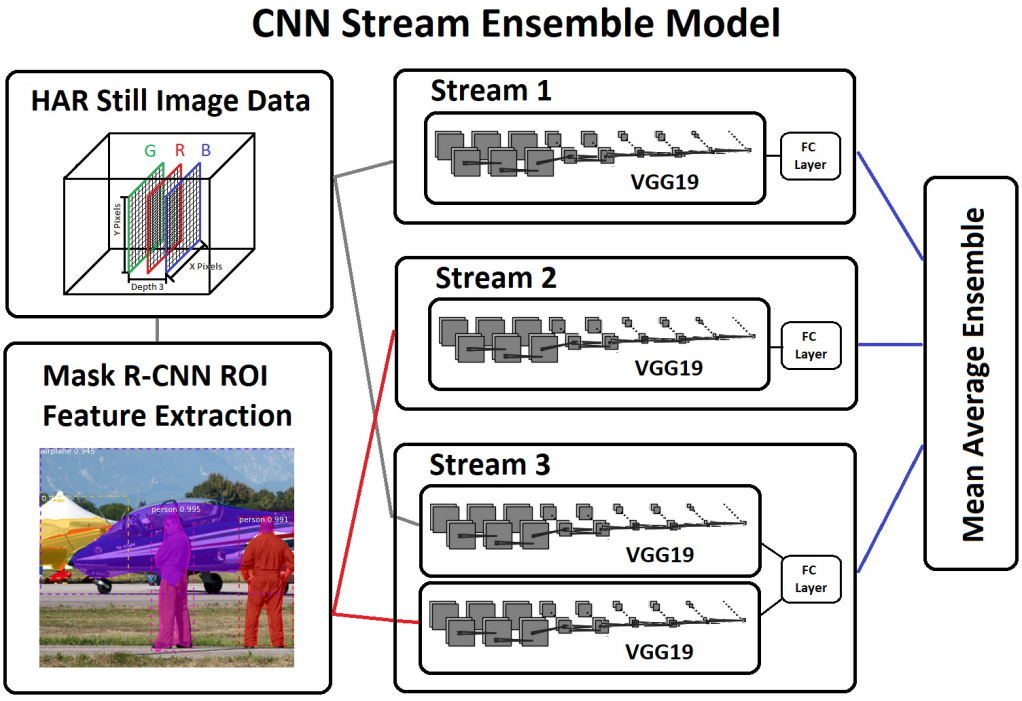


Figure 1: A high-level representation of the proposed CNN stream ensemble model with the raw images and segmented masks yielded by Mask R-CNN as inputs. Stream 1 employs an optimised VGG19 network with raw images as inputs. Stream 2 uses another optimised VGG19 network trained on the saliency maps yielded by Mask R-CNN. Stream 3 fuses a pair of optimised VGG19 networks with raw images and segmented masks as inputs, respectively. Each stream is individually optimised using EnvPSO to identify its optimal settings.

The VGG19 model was selected after conducting preliminary tests against EffcientnetB7, MobileNetV3Small, VGG19, Inception_v3, Resnet_v2, Densenet, Xception and Resnet50 on the HAR task. As a well known deep learning model it is also easy to employ through well establish packages such as PyTorch and TensorFlow along with freely available pretrained weights for imagenet.

It's simple structure also makes the Layer Strip-back implementation more approachable when compared to more complex architectures such as Resnet which have multiple internal branches and skip connections.

The proposed ensemble model comprises two main components, i.e., EnvPSO and EnvPSO-optimised CNN stream ensemble model. The CNN stream ensemble model is used to generate class predictions for HAR with still images as inputs. EnvPSO is used to optimise the hyper parameters of each CNN stream, i.e., the learning rate, batch size, and Layer Strip-back. Once the CNN streams are optimised, they are trained and used to generate class predictions which are subsequently summed and divided by the number of streams to produce an average prediction for each input image. We describe the key components in the following subsections, leading with the proposed EnvPSO variant. Then, the details of the EnvPSO-optimised CNN stream ensemble model are explained.

3.2 The Proposed EnvPSO Algorithm

As previously mentioned, PSO establishes two key elements that stimulate its swarm behaviours, i.e., the social and cognitive terms. The social term replicates a collaborative behaviour by influencing the search directions of particles towards the global best solution. The cognitive term guides each particle to move towards its personal best location. Instead of using the typical fixed coefficients for both terms, the aim here is to adaptively tune these values, enhancing the exploration and exploitation capabilities of the search particles. The standard PSO algorithm also does not take it's local fitness environment into account, which can be beneficial to complement both the social and cognitive terms to accelerate convergence.

Therefore, in this research, a new PSO variant, i.e., EnvPSO, is proposed. It incorporates a new environmental element that embeds Gaussian fitness surface prediction, and linear and exponential adaptive coefficients to balance between diversification and intensification. Specifically, linear and exponential functions are used to generate adaptive search parameters that allow the swarm to focus on global exploration in the beginning and local exploitation towards the end during the search process. In other words, adaptive functions are proposed to adjust both social and cognitive terms to gradually move from exploration to exploitation. To complement the social and cognitive

terms, a third environmental term is proposed, which estimates the fitness surface of the search space for an input function using a Gaussian distribution. It simulates particles to move towards more promising regions during the search process, in an attempt to accelerate convergence. Details of EnvPSO are shown in Algorithm 2.

Algorithm 2 The proposed EnvPSO algorithm

Initialise the swarm size n_p
Initialise a swarm of particles
Initialise the fitness array **A**
Initialise the fitness hyper-surface **S**
Initialise the search parameters w, c_1, and c_2
while $t < t_{max}$ **do**
  for each particle $i = 1, ..., n_p$ **do**
    if $f(x_i^t) > f(p_{best_i})$ **then**
      $p_{best_i} = x_i^t$
    end if
    if $f(x_i^t) > f(g_{best})$ **then**
      $g_{best} = x_i^t$
    end if
    Update **A** using Equation 3.5
  end for
  Update **S** using Equation 3.9
  for each particle $i = 1, ..., n_p$ **do**
    Update search coefficients c_1 and c_2 using Equations 3.1-3.2 or Equations 3.3-3.4, respectively
    Update each particle velocity using Equation 3.11
    Update each particle position using Equation 2.2
  end for
end while
return g_{best}

3.2.1 Adaptive Coefficients

As indicated in Equation 2.1, the standard PSO algorithm assigns constant values to the acceleration coefficients, i.e., c_1 and c_2, which guide the search process. In this research, the effects of adjusting these parameters during the search process are investigated. Specifically, linear and exponential functions for search coefficient generation are proposed. Equations 3.1-3.2 and Equations 3.3-3.4 define both linear and exponential formulae, respectively. Moreover, static coefficients are employed in EnvPSO by setting $c_1 = 2.5$ and $c_2 = 2.0$, for performance comparison purpose.

$$c_1 = c_{max} - \frac{c_{max} - c_{min}}{i_{max}} i \tag{3.1}$$

$$c_2 = c_{min} + \frac{c_{max} - c_{min}}{i_{max}} i \tag{3.2}$$

$$c_1 = \frac{c_{max} - c_{min}}{1 + e^{\frac{5}{i_{max}}(i - \frac{i_{max}}{2})}} + c_{min} \tag{3.3}$$

$$c_2 = \frac{c_{min} - c_{max}}{1 + e^{\frac{5}{i_{max}}(i - \frac{i_{max}}{2})}} + c_{max} \tag{3.4}$$

where $c_{max} = 2.5$ and $c_{min} = 0.5$, while i denotes the current iteration and i_{max} represents the maximum number of iterations. Figure 2 illustrates the adaptive search coefficients generated using Equations 3.1-3.2 and 3.3-3.4, respectively.

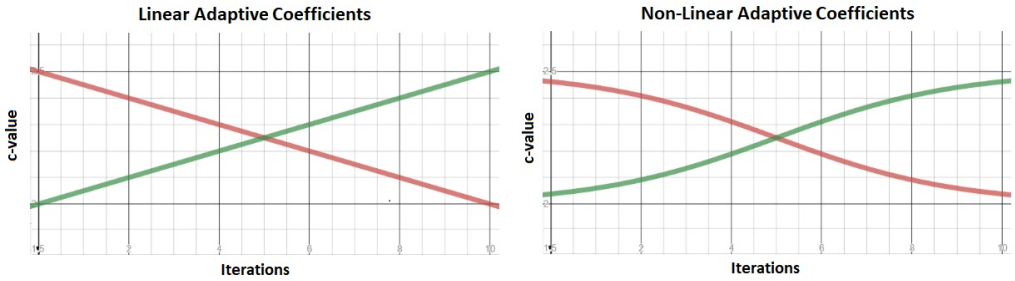


Figure 2: *Left:* Equation 3.1 (red line) generates the linear cognitive coefficient c_1 and Equation 3.2 (green line) generates linear social coefficient c_2.*Right:* Equation 3.3 (red line) generates exponential cognitive coefficient c_1 and Equation 3.4 (green line) generates exponential social coefficient c_2.

Such adaptive linear and exponential coefficients enable the swarm to focus on global exploration at the beginning of the search process and local exploitation towards the end.

Besides adaptive social- and cognitive-based terms, an environmental term pertaining to fitness surface estimation using Gaussian distribution is proposed, as explained in the following subsection.

3.2.2 Gaussian Fitness Surface Prediction

To further enhance the exploitation and exploration capabilities of PSO, a third environmental term is introduced to complement both social and cognitive-based terms in the velocity-updating formula. In essence, this new strategy adds an environmental awareness to particles by providing information on the function being evaluated. Since it is not possible to obtain the fitness scores of unevaluated positions in the search space, estimations can be made via fitness scores from the nearby previously evaluated positions. Using these estimations, a rough landscape of the

fitness surface for the input function can be determined. As the algorithm progresses, estimation of the complete fitness surface becomes more accurate. Based on this estimated surface, gradient information can be extracted to influence the velocity of a particle by pushing it along the direction toward fitter solutions. The extracted gradient information lays the foundation for the proposed third environmental term in accelerating convergence.

A pictorial example of this fitness surface is displayed in Figure 3. It shows how the landscape of estimated fitness surface changes over time when EnvPSO is used to solve a classic minimisation problem, i.e., the Ackley benchmark function. Initially, the landscape of estimated fitness surface (magenta) appears flat (when $i = 1$ in Figure 3). When particles explore and evaluate positions of

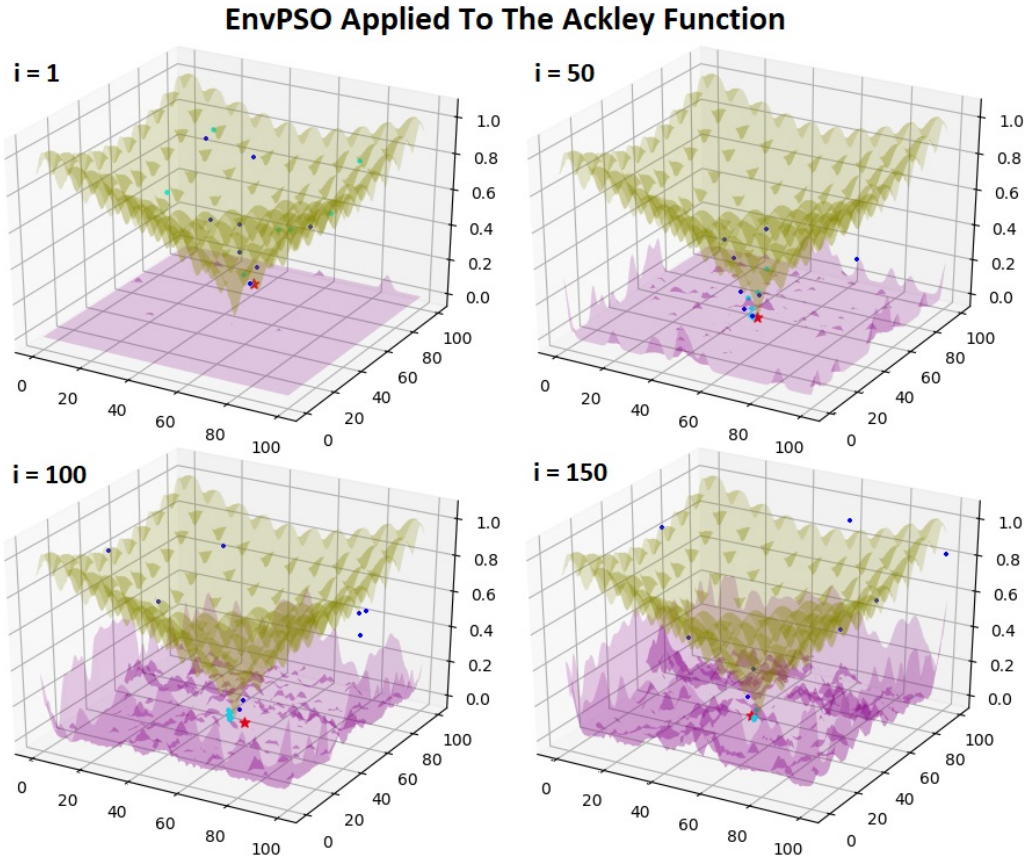


Figure 3: Variations of the Ackley function (yellow surface) and estimated Gaussian fitness surface (magenta surface) yielded by EnvPSO at iteration i=1, 50, 100, and 150. Blue points indicate current positions of particles, cyan dots show their historical personal best positions, while the red stars indicates the current global best position.

the input function (i.e., Ackley function), the associated gradient information in each dimension of the estimated fitness surface is extracted and exploited to influence their velocity. Notice that the estimated surface does not form a one-to-one representation pertaining to the input function.

Instead, the estimated surface is convolved with a dimensionally appropriate Gaussian kernel, in order to smooth the fitness landscape and provide a better approximation of the shape of the input function. This leads to appropriate gradient information to be utilised for influencing velocities of particles in the search process.

Specifically, the gradient information is generated by collecting all the currently evaluated positions in the search space and mapping them to a zero index n-dimensional integer array, where n represents the number of targeted hyper parameters. Mapping parameters with a continuous domain in this way requires an array of infinite size. To solve this problem, several equidistant points between the maximum and minimum values of the continuous domain are choosen to serve as indexes of a particular dimension. Once defined, each value in the fitness array **A** is initialised to zero. When a particle is evaluated, its fitness value $f(x_i^t)$ is stored in **A** at an index corresponding to its current particle position x, as defined in Equation 3.5.

$$\mathbf{A}(x_i^t) = f(x_i^t) \tag{3.5}$$

where x_i^t is the position of the ith particle at iteration t. After evaluating all particles in the current iteration, an n-dimensional fitness hyper-surface **S** is created by convolving a Gaussian filter over **A** using Equations 3.6, 3.7, 3.8, and 3.9. Firstly, Equation 3.6 is used to calculate the standard deviation of the Gaussian operation.

$$\sigma_d = \theta \times (max(V_d) - min(V_d)) \tag{3.6}$$

where σ_d is the standard deviation of dimension d with θ as a predefined smoothing factor. Then, σ_d is used in Equation 3.7 to generate the Gaussian kernel for convolution operations.

$$G_d(r) = \frac{1}{\sqrt{2\pi}\sigma_d} e^{-\frac{r^2}{2\sigma_d^2}} \tag{3.7}$$

where G_d is the Gaussian kernel for dimension d and its domain R is defined in Equation 3.8.

$$R = \{r \mid r \text{ is an integer, and } -4\sigma_d + 0.5 \leq r \leq 4\sigma_d + 1.5\} \tag{3.8}$$

The Gaussian kernel in the dth dimension is convolved sequentially along the dth axis of A as indicated in Equation 3.9.

$$S_d(\tau) = A(\tau) * G_d(\tau) \tag{3.9}$$

Before updating each particle's position and velocity, its current position x_i^t is used to index a point on the fitness hyper-surface $S_d(\tau)$ generated using Equation 3.9, from which the gradient information of the surface in each dimension is extracted. The gradient information is calculated using second-order finite central differences, as in Equation 3.10.

$$\Delta x_{id} = \frac{\mathbf{S}(x_{id} + h) - \mathbf{S}(x_{id} - h)}{2h} \tag{3.10}$$

where Δx_{id} is the gradient associated with dimension d of the ith particle at an indexed position x. Note that $x_{id} + h$ represents the proceeding neighbouring point of x_{id} at a predetermined distance h, while $x_{id} - h$ indicates an indexed position in the opposite direction. Since **S** is indexed with integers incrementing by 1, $h = 1$ is applied to obtain the adjacent position. Figure 4 shows the underlying procedure.

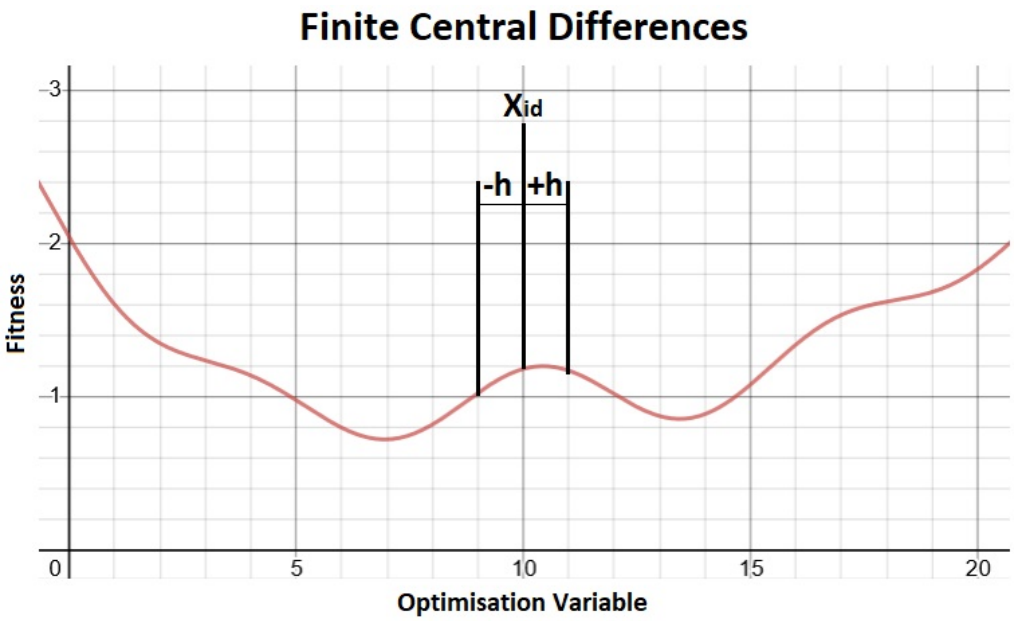


Figure 4: The application of finite central differences to an arbitrary function, where x_i refers to the ith particle and the red line represents the estimated fitness surface of the function, where $d = 0$ indicates a one-dimensional input. Here h is the step-size in Equation 3.10, which is set to 1 so that it lines up with the integer indexing scheme of A.

With the gradient information extracted from Equation 3.10, the environmental term Δx_i^t for the ith particle can be constructed, resulting in a vector of fitness gradient information with length d for velocity updating. Equation 3.11 is used to update each particle's velocity.

$$v_i^{t+1} = wv_i^t + r_1c_1(p_{best}^t - x_i^t) + r_2c_2(g_{best}^t - x_i^t) + \Delta x_i^t \tag{3.11}$$

Finally, the new particle velocity is used for updating its position using Equation 2.2. The proposed Gaussian fitness estimation surface equips the swarm with higher chances in exploring promising search regions, while reducing the risk of being trapped in local optima, in order to accelerate convergence.

The final PSO addition, namely Layer Strip-back, defines the number of CNN layers to be re-trained in the transfer learning process when EnvPSO is used to optimise network hyper parameters. An analysis is provided below.

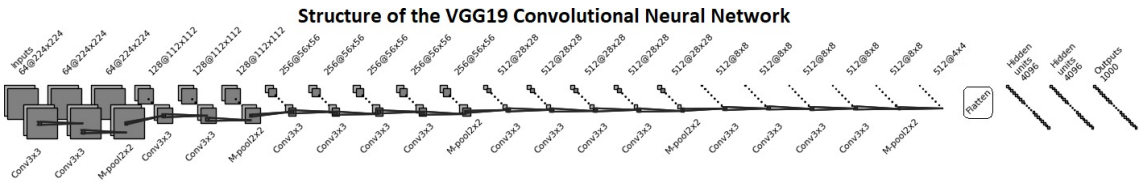


Figure 5: Layer configurations of the VGG19 network. The ImageNet pre-trained VGG19 models used in the proposed three streams are provided in the red Python package tensorflow.keras.application, which require an input shape of (224, 224, 3). Each network is adjusted by replacing the final three dense layers with three new counterparts, where the first and second dense layers have 1000 and 100 neurons, respectively, while the final output layer has neurons equivalent to the target classes in the training set.

3.2.3 Layer Strip-back

Three CNN streams are used to form the ensemble model. The first stream is based on a VGG19 [94] backbone pre-trained on the ImageNet data set. Its structure is displayed in Figure 5. To optimise matrix calculations and GPU memory allocation when training a pre-trained CNN for a new task, the number of layers to be re-trained can be manually selected. By reducing the number of trained layers, the number of required matrix calculations is also reduced, leading to economic use of computation cycles and GPU memory. Rather than manually determining the number of re-trained layers for transfer learning, the layer selection process is automated by creating a variable

called Layer Strip-back, which is presented in Figure 6.

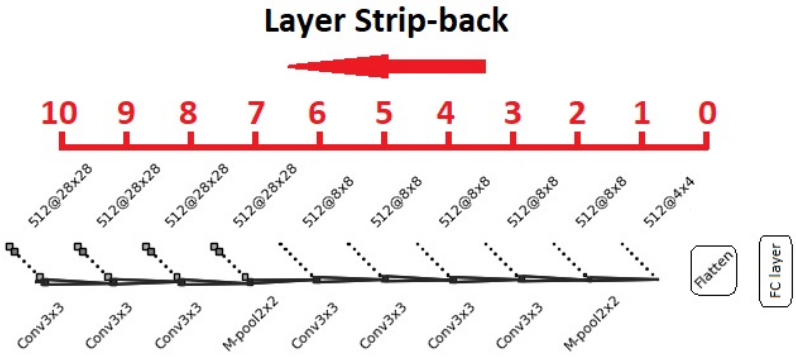


Figure 6: The Layer Strip-back parameter is applied to the VGG19 network. Note that zero indicates that no convolutional layer prior to the flatten layer needs to be re-trained.

This variable is assigned an integer value in a range of [0, 10], which determines the number of layers back from the final layer of the backbone network used for re-training. For instance, if the Layer Strip-back value is 2, then only the last two layers in the network need to be re-trained. This variable is automatically determined, like any other hyper parameters (i.e., the learning rate and batch size), during the optimisation process. After optimisation with EnvPSO, the proposed CNN ensemble model is used in a multi-stream form for HAR tasks.

3.2.4 The Multi-Stream Ensemble Model

We present the details of an ensemble model consisting of three CNN streams in the following subsections.

Stream 1 - VGG19 with Raw Images

The first stream is a VGG19 network [94] pre-trained on the ImageNet data set. Its structure is displayed in Figure 5. It is adjusted by replacing the original final three dense layers with three new fully connected dense layers, where the first dense layer has 1000 neurons, the second dense layer with 100 neurons and the final output layer has neurons equivalent to the target classes in the training data set. The input images are resized to $(224, 224, 3)$, in order to match the input shape of the first convolutional layer of the VGG19 network. An overview of this first stream is provided in Figure 7.

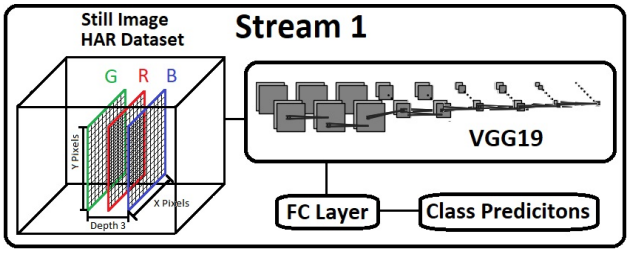


Figure 7: An overview of the first stream

Stream 2 - VGG19 with Mask R-CNN Features

The second stream is composed in a manner similarly to that of the first CNN stream, but differs by the input it receives. Instead of using the resized raw images as inputs, a pre-processing step is applied to the raw images to extract features via a Mask R-CNN [95] pre-trained on the MSCOCO data set. Mask R-CNN uses a Region Proposal Network (RPN) to propose candidate object bounding boxes. Classification and bounding box regression are then performed, while concurrently producing a binary segmentation mask for each class. This allows retrieval of the class probability, the bounding box offset and a binary segmentation mask for each detected object in a given input image. In addition, each detected class is represented by a particular shade. This pre-processing procedure using Mask R-CNN yields a resized grey-scale unsigned 8-bit integer image with class-encoded segmentation masks for all detected objects (see Figure 8).

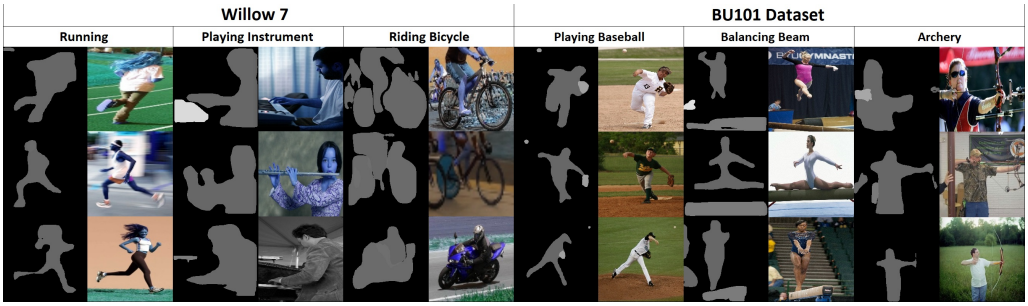


Figure 8: Examples from three of the 7 classes from the Willow7 data set as well as three of the 101 classes from the BU101 data set. Each column displays three examples of the grey scale images generated using Mask R-CNN and their corresponding raw images. Each grey shade represents a different class prediction for the region of pixels it covers in the raw image.

This output grey-scale image is used as the input to the VGG19 network in stream 2. In this way, we represent class categories as different shades, allowing previously identified class information to inform subsequent classification. Figure 9 illustrates the overview of this stream.

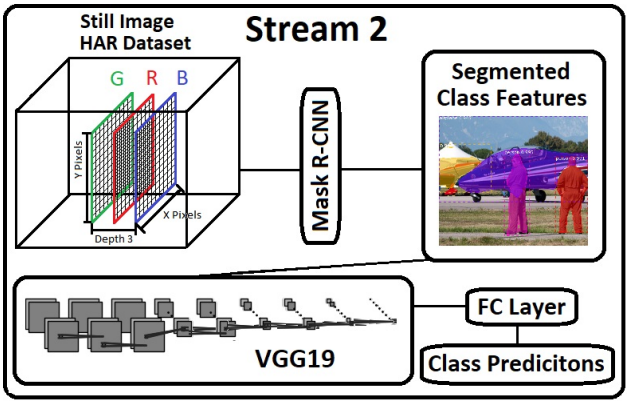


Figure 9: An overview of the second stream

Stream 3 - A Fusion of Streams 1 and 2

As indicated in Figure 10, the last stream fuses VGG19 networks in streams 1 and 2 together. It adds a flattening layer after the final convolutional layer of each network, and concatenates them to form an end-to-end trainable CNN. Its inputs are thus both raw images from stream 1 and pre-processed salient regional images from stream 2. These two types of input images are simultaneously used for training. Based on streams 1, 2 and 3, we construct an ensemble model to overcome bias and enhance performance.

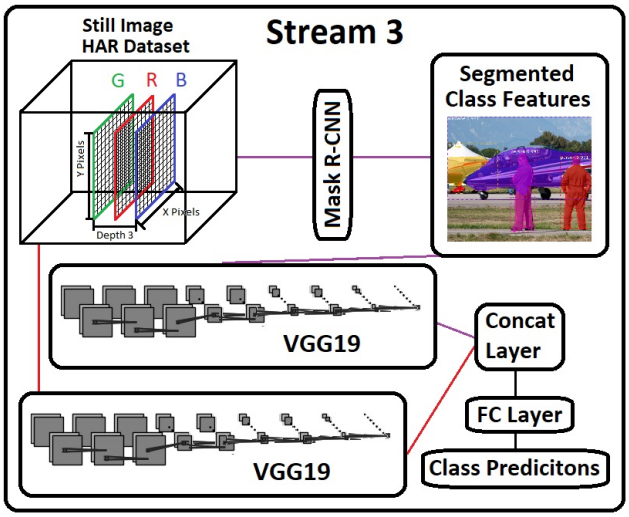


Figure 10: An overview of the third stream

Stream Ensemble Model

Each constituent stream is first optimised independently using EnvPSO to identify the optimal learning rate, batch size and Layer Strip-back settings. The search ranges of these hyper parame-

ters are shown in Table 1.

Table 1: Hyper parameter search ranges

Hyper parameter	Range
Batch Size	[8, 64]
Learning Rate	[0.001, 0.01]
Layer Strip-back	[0, 10]

During training, the target stream is trained for three epochs at each EnvPSO iteration. Then, it is evaluated based on a validation set to yield the class predictions. The MAP indicator is used as the fitness score pertaining to the particle's position in the search space. Once the optimisation process is completed, the optimal hyper parameters are used to train the corresponding CNN stream for 100 epochs. After training, the CNN models are evaluated using the test set, giving the final class predictions. Once all the streams are evaluated, they are ensembled by taking the mean average of predictions. We repeat this procedure for 10 trials and take the average results, in order to avoid randomness in CNN training. The mean MAP result over a set of 10 runs is used for performance comparison, as indicated in Figure 11. A learning rate range of 0.001-0.01 was selected to work with a dynamically scheduled reduction in learning rate. Specifically the learning rate is scheduled to reduce by a factor of 20% when the validation loss has not reduced in the past 3 epochs. Over the course of 100 epochs it is likely for the loss to fail this check causing the learning rate to be reduced to values more inline with standard static learning rates. This allows for larger changes to the model weights early on enabling a good initial weight configuration before more finely adjusting the weights with lower learning rates. This optimisation range is therefore set a little higher than might be typically expected from a static learning rate.

This multi-stream EnvPSO-optimised ensemble model is illustrated in Algorithm 3.

3.3 Experimental Studies

In this section, we evaluate the proposed three-stream ensemble model with EnvPSO-optimised hyper parameters using two HAR data sets, i.e. the Willow7 [17] and BU101 [96] data sets. To better understand the impact of additional contributions to original PSO, we evaluate each proposed strategy separately. Specifically, we compare the MAP scores of CNN models trained with hyper parameters optimised by PSO and EnvPSO using static, linear and nonlinear search coefficients in

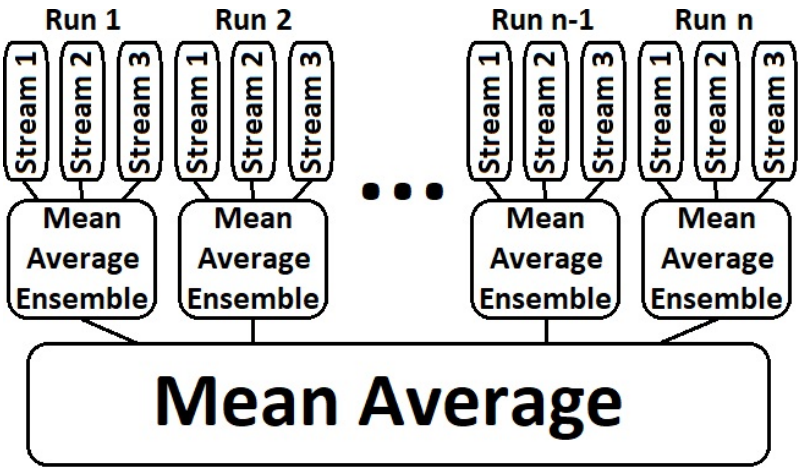


Figure 11: Construction of the ensemble model, where the class predictions of each stream are combined using the mean average.

each individual CNN stream as well as the ensemble model of every possible permutation of the streams. In addition, we compare the MAP results with those from other state-of-the-art existing methods.

The following settings are followed, in order to ensure consistency in experiments. Every CNN stream is trained with a stochastic gradient descent optimiser using a categorical cross-entropy loss function, as well as a Nestorov momentum of 0.01 and a decreasing learning rate that reduces by 1/5 when the validation loss does not improve over three consecutive epochs. The settings of static, linear and exponential search coefficients used in PSO and EnvPSO are shown in Table 2. The configurations are used to train PSO-optimised and EnvPSO-optimised CNN ensemble models on each data set. The trained ensemble models are subsequently evaluated on the unseen test set. We adopt the following settings throughout the experiments, i.e. population=10, maximum number of iterations=30, and dimension=3. In addition, a total of 10 runs are performed to construct 10 optimised stream ensemble models. The mean results of the 10 stream ensemble models are used for performance comparison.

3.3.1 Data sets

We use the following two key data sets that have been used in several related studies.

Algorithm 3 The Proposed Multi-stream CNN Ensemble Model with EnvPSO-Optimised Hyper parameters

runs $= 10$
run $= 0$
while run $<$ runs **do**
  for stream $= 1, 2, 3$ **do**
    Conduct hyper parameter optimisation for each stream using EnvPSO in Algorithm 2
  end for
end while
run $= 0$
while run $<$ runs **do**
  for stream $= 1, 2, 3$ **do**
    Train each optimised CNN model in each stream using a larger number (i.e.100) of epochs
    Evaluate the trained network using the test set for each stream
  end for
  Calculate the mean result of the three streams in each ensemble model to yield final class prediction
end while
Take mean average of the MAP results

Willow7

The Willow7 data set [17] consists of 7 classes containing 968 images extracted from Flickr. The classes are, 'Interacting with Computer', 'Photographing', 'Playing Instrument', 'Riding Bike', 'Riding Horse', 'Running' and 'Walking'. We employ the official train, validation and test data splits for each class category in our experiments.

BU101

The BU101 data set [96] comprises 23.8K manually filtered web images pertaining to actions from 101 classes. These class categories have a 1-1 correspondence with those of the UCF101 video action data set. Some example classes are, 'MoppingFloor', 'PullUps', 'Knitting', 'SkateBoarding' and 'Typing'. In addition, a total of 2769 images are taken from Stanford40, which share the same class categories (e.g. 'PlayingViolin' and 'Rowing') as those in UCF101. Each class in the BU101 data set contains 100-300 images extracted from the above sources. This data set does not have an official train/test data split. We use a train/validation/test split of 70/10/20, as adopted in other existing studies [74]. Specifically, we apply the above split to each class so that we obtain the same ratio of class samples to form train/validation/test sets.

Table 2: EnvPSO and PSO Settings

Method	Value
Static	Cognitive acceleration coefficient $c1 = 2.5$ Social acceleration coefficient $c2 = 2.5$ Inertia weight $w = 0.1$
Linear	Linear cognitive and social search coefficients generated using Equations 3.1 and 3.2 with $c_{max} = 2.5$ and $c_{min} = 0.5$, Inertia weight $w = 0.1$
Non-linear	Non-linear exponential cognitive and social search coefficients generated using Equations 3.3 and 3.4 with $c_{max} = 2.5$ and $c_{min} = 0.5$, Inertia weight $w = 0.1$

3.3.2 Evaluation of Human Action Recognition Models

The MAP metric is computed to determine the effectiveness of the EnvPSO-optimised CNN ensemble model. The mean results of 10 separate runs using the Willow7 and BU101 data sets are shown in Tables 3 and 4, respectively. The numbers in the first row of these tables refer to which streams are being ensembled to obtain the final predictions. Static, linear and non-linear refer to constant, linear and non-linear (exponential) search coefficients respectively.

Table 3: The mean MAP results over 10 runs for the CNN stream ensemble models using the Willow7 data set. (The '+' symbol indicates the streams that have been ensembled.)

Stream	PSO			EnvPSO			
	Static (%)	Linear (%)	Non Linear (%)	Static (%)	Linear (%)	Non Linear (%)	Stream Avg. (%)
1	64.6	**66.0**	68.4	69.2	72.8	76.2	69.53
2	49.3	58.1	59.0	60.7	63.0	72.4	60.42
3	62.2	61.0	67.3	**75.5**	75.7	76.5	69.70
1+2	62.4	65.9	66.5	66.0	74.7	71.4	67.82
1+3	**64.9**	**66.0**	**69.4**	73.6	**76.2**	**76.8**	71.15
2+3	59.2	64.0	64.3	70.1	71.4	75.5	67.42
1+2+3	63.5	65.1	67.5	70.3	76.0	73.3	69.28
MAP Avg.	60.87	63.73	66.06	69.34	72.83	74.59	
Total Avg		63.55			72.25		

In Tables 3 and 4, streams 1, 2 and 3 represent optimised VGG19 with raw images as inputs, optimised VGG19 with extracted Mask R-CNN salient features as inputs, and fusion of both streams 1 and 2, respectively. As illustrated in Tables 3 and 4, in all search methods, ensemble models with stream 1 or stream 2 combining with stream 3 achieve enhanced performances, indicating

Table 4: The mean MAP results over 10 runs for the CNN stream ensemble models using the BU101 data set. (The '+' symbol indicates the streams that have been ensembled.)

Stream	PSO			EnvPSO			Stream Avg. (%)
	Static (%)	Linear (%)	Non Linear (%)	Static (%)	Linear (%)	Non Linear (%)	
1	83.6	84.6	85.6	88.8	89.1	88.9	85.80
2	61.8	66.0	66.1	72.0	62.2	70.6	65.47
3	85.7	86.6	86.8	88.8	88.5	89.6	87.33
1+2	82.3	84.1	82.7	88.0	87.1	88.2	84.52
1+3	**86.6**	**87.1**	**87.5**	**89.6**	**89.5**	**89.7**	87.82
2+3	82.4	84.6	85.7	88.2	86.2	88.6	85.53
1+2+3	85.5	86.3	86.4	89.6	88.8	89.5	87.15
MAP Avg.	81.21	82.76	82.97	82.46	84.49	86.44	
Total Avg		82.31			84.46		

that additional diversity introduced by stream 3 offers significant advantage over those individual streams. In addition, ensemble models of streams 1 and 3 typically achieve the best performance with both data sets for nearly all search methods. The most effective configuration for both data sets is the ensemble model of streams 1 and 3 optimised by EnvPSO with non-linear adaptive coefficients, where the proposed strategies such as Gaussian fitness surface prediction and adaptive exponential coefficients work cooperatively to enhance local and global search capabilities, as compared with the original PSO algorithm.

Notably stream 2 shows poor performance in comparison with those of streams 1 and 3 for both the original and proposed PSO methods. This could be owing to a reduction of available information in the segmented mask image features, since many aspects of original images are removed including colour and local pixel information within the segmented areas and backgrounds. Despite this missing information, the networks still manage to classify over 50% of the class instances correctly using this method alone in most test cases. This suggests that processing raw images with Mask R-CNN is able to produce salient features that benefit the classification tasks. Stream 3, however, does not suffer from this problem as both inputs (i.e. Mask R-CNN extracted salient features and raw images) are combined through the two fused VGG19 networks, allowing the resulting networks to access more information. This is reflected in the results with stream 3 revealing the second highest stream average results of 69.70% and 87.33% for the Willow7 and BU101 data sets respectively. Ensemble models with streams 1 and 3 produce scores similar to or better

than those of stream 3, as indicated by the stream average results of 71.15% and 87.82% for Willow7 and BU101, respectively; the highest of all the stream average results. In other words, by ensembling streams 1 and 3, a consistent enhancement in performance with respect to both data sets is achieved.

Analysing the average results of static, linear and nonlinear coefficients for both original and proposed PSO algorithms reveals that the proposed non-linear exponential formulae for search coefficient generation contribute toward a more optimal configuration in exploration and exploitation pertaining to hyper parameter search. In other words, the results of both PSO and EnvPSO using adaptive exponential search coefficients show consistent enhancement in most test cases.

The average results for all EnvPSO-optimised streams are 72.25% and 84.46% for Willow7 and BU101, respectively. In contrast, the corresponding mean results of all the PSO-optimised streams are inferior, i.e. 63.55% and 82.31%, for Willow7 and BU101, respectively. The differences between these EnvPSO and PSO results are therefore 8.7% for Willow7 and 2.15% for BU101. These differences highlight the overall superiority of the EnvPSO optimised streams over those optimised by the baseline PSO method. Owing to adoption of the Gaussian surface prediction function, the search process of EnvPSO is better guided and is capable of exploring and exploiting optimal regions more thoroughly with better chances of attaining global optimality. In addition, Gaussian surface prediction in conjunction with adaptive exponential search coefficients further diversifies the search process with adaptive local and global search operations for hyper parameter search, while accelerating convergence. The resulting hyper parameters show greater efficiency in re-training VGG19 networks for undertaking HAR problems.

Hyper parameter Selection

We analyse the identified mean optimal hyper parameters for stream 1 CNN models as an example case study to indicate efficiency of the proposed EnvPSO model. Tables 5 and 6 show the selected mean hyper parameters for stream 1 CNN models over 10 runs for each search method on the Willow7 and BU101 data sets, respectively.

Referring to Table 5 for the Willow7 results, comparing EnvPSO and PSO in static, linear and nonlinear coefficient settings reveals that the average Layer Strip-back configurations identified by EnvPSO are consistently higher. Such higher Layer Strip-back settings from EnvPSO provide

Table 5: Average hyper parameters identified by each search method for stream 1 CNN models over 10 runs on the Willow7 data set

Variant	Batch Size	Learning Rate	Layer Strip-back	MAP (%)
EnvPSO stat	34.50	0.0051	4.7	69.2
EnvPSO lin	35.40	0.0056	5.9	72.8
EnvPSO nonlin	35.20	0.0053	6.1	76.2
PSO stat	39.40	0.0059	4.6	64.6
PSO lin	39.90	0.0036	4.6	66.0
PSO nonlin	36.90	0.0043	4.4	68.4

Table 6: Average hyper parameters identified by each search method for stream 1 CNN models over 10 runs on the BU101 data set

Variant	Batch Size	Learning Rate	Layer Strip-back	MAP (%)
EnvPSO stat	14.5000	0.0084	4.2	88.8
EnvPSO lin	15.1000	0.0082	5.6	89.1
EnvPSO nonlin	13.3000	0.0070	4.5	88.9
PSO stat	8.7000	0.0069	3.2	83.6
PSO lin	9.8000	0.0076	3.5	84.6
PSO nonlin	11.9000	0.0064	3.6	85.6

better capabilities for re-training the network on the new data sets without interfering with useful filter configurations in earlier layers. In comparison with larger and smaller learning rates yielded by PSO with constant and adaptive coefficients, EnvPSO produces moderate learning rates, leading to a better trade-off between performance and convergence speed. These optimal settings, i.e. larger Layer Strip-back configurations and moderate learning rates, account for the better MAP results from stream 1 CNN models from EnvPSO, as illustrated in Table 5.

The best configuration is EnvPSO with non-linear adaptive coefficients, producing a moderate mean learning rate and the highest Layer Strip-back setting amongst all methods. In contrast, the worst configuration is PSO with static coefficients, which yields a smaller mean Layer Strip-back setting with the largest average learning rate. Such settings result in a fast convergence to sub-optimal solutions as well as poor acquisition of the new domain knowledge and discriminative characteristics, as evidenced by the lower MAP results in Table 5.

Next we analyse the identified average hyper parameters of each search method for the stream 1 CNNs with respect to BU101 in Table 6. Again, the EnvPSO models with both static and adaptive coefficients produce larger Layer Strip-back settings than those from PSO. This further indicates that EnvPSO consistently identifies a stronger correlation between enhanced results and compara-

tively more re-training of network layers in the transfer learning process. The best configuration is EnvPSO with linear coefficients, which extracts the highest mean Layer Strip-back and batch-size settings, as well as a moderate average learning rate. Such optimal settings enable a sufficient re-training of network using new data set as well as better efficiency in extracting spatial patterns in each batch of this comparatively larger and more complex data set. On the contrary, PSO with static coefficients yields the smallest Layer Strip-back and batch-size settings, therefore the lowest performance amongst all methods. Since the training set of BU101 is larger than that of Willow7, there are larger numbers of batches in the BU101 training set than those in the Willow7 training set. Therefore, comparatively smaller batch sizes are identified by both EnvPSO and PSO for BU101 than those of Willow7.

In short, under both static and adaptive coefficient settings, EnvPSO selects higher Layer Strip-back configurations on average as compared with those yielded by PSO across both data sets for stream 1 CNN models. These findings suggest that EnvPSO is capable of optimising the Layer Strip-back parameters to fine-tune more CNN layers during re-training. Combined with moderate and higher average learning rates, EnvPSO is able to conduct better re-training of CNN streams and extract better new domain knowledge from the data samples, while providing better generalisation when evaluating using the unseen test samples without succumbing to overfitting or underfitting issues. Similar characteristics of identified hyper parameters are obtained for optimisation of VGG19 networks in streams 2 and 3, where EnvPSO yields larger Layer Strip-back and moderate learning rate configurations.

We now compare the devised CNN stream ensemble model using EnvPSO with adaptive exponential coefficients against state-of-the-art methods on both Willow7 and BU101 data sets, as shown in Tables 7 and 8, respectively.

Table 7 illustrates the comparison for the Willow7 data set. Each existing study shown in Table 7 employs the overall data set for evaluation. As illustrated in Table 7, our devised CNN stream ensemble model achieves an MAP score of 76.8%, outperforming all existing methods on the Willow7 data set. Our optimised three CNN streams illustrate significant diversity, as evidenced by the identified different Layer Strip-back and learning configurations. Such distinctive model settings enable the extraction of different internal feature representations, providing complementary properties to enhance ensemble model performance. In addition, the best baseline method is the

Table 7: HAR methods on Willow7

Studies	Methodology	MAP
Zhang et al. [72]	MAE	75.31%
Yu et al. [76]	Deep ensemble learning voting strategy (DELVS3) using tuning weight voting on 6 base deep learning methods	73.69%
Yu et al. [76]	DELVS2 using tuning weight voting on 3 deep learning models, i.e. VGG16_NCNN, VGG19_NCNN, and ResNet50_NCNN	71.89%
Delaitre et al. [10]	A locally order-less spatial pyramid bag-of-features model using action-specific body parts and object interaction representations	71.70%
Safaei and Foroosh [97]	Ranked saliency map and predicted optical flow + STCNN	71.60%
Safaei and Foroosh [97]	STCNN + intermediate feature space tensor Q	66%
Sharma et al.[71]	EPM with additional context (EPM + context)	67.60%
Sharma et al.[71]	EPM without context of 1.5x extension of bounding boxes	66.00%
Sharma et al.[98]	Discriminative spatial saliency with max margin classifier	65.90%
Wang and Wang [73]	SPN with classification by the Most Probable Explanation (MPE) method.	48.70%
Wang and Wang [73]	Flat Spatial SPN (FS-SPN).	65.30%
Wang and Wang [73]	Individual learning Hierarchical Spatial SPN (IHS-SPN).	71.30%
Wang and Wang [73]	Joint learning Hierarchical Spatial SPN (JHS-SPN). This method is the same as IHS-SPN except that it learns the weights of the shared edges and images between SPNs from two different classes.	71.70%
Wang and Wang [73]	Spatial Pyramid Matching as proposed by Lazebnik et al. [99]	63.70%
Ours	Multi-stream ensemble with EnvPSO-based hyper parameter optimisation	**76.80%**

MAE model [72], with an MAP result of 75.31%. This MAE model uses various techniques (such as Markov random field) to extract a contextual segmentation mask that links a person and the object being interacted with, in order to enhance classification performance. In our approach, we use a similar saliency extraction method based on Mask R-CNN, where the segmented regional images provide context for the person and related objects. Besides the above, other strategies such as

adoption of multiple types of inputs, hyper parameter fine-tuning of stream CNNs and ensembling mechanisms are able to enhance performance. Therefore, our approach leads to better robustness than those of [72].

The second-best baseline method is DELVS [76], where six base methods are embedded to yield 73.69% of mean MAP. The model proposes a tuning weight voting ensemble method to integrate the results of the following six base methods, i.e. VGG16, VGG19, ResNet50, VGG16_NCNN, VGG19_NCNN and ResNet50_NCNN. The ensemble method achieves promising performance by taking advantage of diverse deep networks and their potential to produce different internal representations with respect to training data. In comparison, our ensemble model achieves better diversification using both backbone networks and input data. EnvPSO is first used to devise optimal network and learning settings for each stream CNN model. Besides using original input images, salient segmented regional images yielded by Mask R-CNN are exploited as inputs in our CNN streams. In this way, our ensemble model incorporates distinctive base networks with different learning behaviours as well as diverse input channels for tackling HAR tasks.

Table 8: HAR methods on BU101

Studies	Methodology	MAP
Li et al. [74]	ResNet101 pre-trained on ImageNet	88.30%
Safaei and Foroosh [97]	STCNN	70.06%
Safaei et al. [100]	a two-stream spatio-temporal network (TSSTN)	72.8%
Alraimi [101]	VGG11 + visual word embedding.	81.70%
Alraimi [101]	VGG13 + visual word embedding. This configuration is the same as the VGG11 + visual word embedding with a different backbone network (VGG13).	77.80%
Alraimi [101]	VGG11 pre-trained on ImageNet	73.40%
Alraimi [101]	VGG16 + visual word embedding. This configuration is the same as the VGG11 + visual word embedding with a different backbone network (VGG16).	58.10%
Alraimi [101]	VGG16 pre-trained on ImageNet	56.60%
Safaei [75]	STCNN with prior knowledge	72.30%
Safaei [75]	ZTD with prior knowledge	71.16%
Safaei [75]	VGG13 pre-trained on ImageNet	70.02%
Safaei [75]	ZTD without prior knowledge	68.24%
Ours	Multi-stream ensemble with EnvPSO-based hyper parameter optimisation	**89.70%**

We subsequently compare our optimised CNN stream ensemble model with existing studies in

Table 8 for BU101. Since there is no official test/train split for the BU101 data set, Table 8 shows an estimated indication of model performance. EnvPSO-optimised CNN stream ensemble model achieves a mean MAP score of 89.7% indicating superior performance against those from existing methods. Owing to the optimised transfer learning process using EnvPSO supported by the Layer Strip-back parameter, our approach is able to fine-tune different numbers of re-trainable layers to better extract discriminative features and distinguish subtle variations of different action classes. Furthermore, we adopt a stream ensemble model incorporating diverse optimised base networks with both raw images and segmented salient regional proposals as inputs to diversify the ensemble operation. Our yielded CNN stream ensemble models therefore possess better robustness and diversity, as compared with those from the existing methods. In addition, [74, 101] employed ResNet101 and VGG11/13 models with embedding strategies and obtained promising performances. However, these models (and most of existing methods) employ a standard transfer learning process without applying any adaptive re-training mechanism to dynamically adjust the number of re-trainable layers. In addition, the use of automatic hyper parameter fine-tuning and/or salient regional features as additional input is not available in [74, 101]. These models also do not perform ensemble of distinctive optimised networks equipped with diverse learning options and different input contexts, therefore limiting the performance.

We present a theoretical analysis between EnvPSO and PSO, as follows. EnvPSO incorporates a new environmental term embedding a Gaussian fitness estimation surface as well as exponential adaptive coefficients to balance the search process and accelerate convergence. Specifically, the environmental term yielded from the gradient information of Gaussian fitness estimation surface adjusts the velocity of particles towards more promising search regions, leading to optimal discovery of hyper parameter configurations. As such, it produces streams with better generalisation capabilities. By implementing exponential adaptive coefficients, EnvPSO illustrates a greater ability to tailor its exploration and exploitation to overcome local optima traps, leading to efficient CNN streams with effective network and learning settings. Furthermore, the introduction of Layer Strip-back parameter provides a unique way to optimise the number of layers to be fine-tuned. These proposed mechanisms work cooperatively to mitigate premature convergence and account for superior performance of our proposed ensemble model. In contrast, standard PSO employs a single leader-based search process. Without the fitness estimation surface as additional guidance,

it is more likely to become stagnant, leading to sub-optimal hyper parameters. Such settings of comparatively less efficient Layer Strip-back configurations fail to train a sufficient number of CNN layers to form a better generalised representation of training data. As a result, it extracts limited domain knowledge, which in turn affects the performance of the resulting stream ensemble model.

On the other hand, using Mask R-CNN to generate class segmented images as a pre-processing step yields salient information for training VGG19 networks. Combining these pre-processed regional images and raw images as a 'multi-modal' input for CNN streams enriches spatial feature representations and better represents subtle variations between different action classes. Furthermore, incorporating multiple unique streams into an ensemble model enhances the overall performance by leveraging differences between the underlying learned representations present within different streams.

3.3.3 Evaluation using Benchmark Test Functions

To further examine the performance of EnvPSO, we present another evaluation using eleven benchmark functions, as shown in Table 9. Each benchmark function produces a unique shape that presents a challenging task to attain the global minima. In particular, we use seven unimodal functions of Sum Squares (Sumsqu), Zakharov, Sum of Different Powers (Sumpow), Sphere, Rosenbrock, Rotated Hyper-Elipsoid (Rothyp) and Dixon-Price, as well as four multimodal functions of Powell, Rastrigin, Griewank and Ackley.

Table 9: Benchmark Functions

Name	Range
Ackley	[-15, 30]
Dixon-Price	[-10, 10]
Griewank	[-600, 600]
Rastrigin	[-5.12, 5.12]
Rothyp	[-65, 65]
Rosenbrock	[-5, 10]
Sphere	[5.12, 5.12]
Sumpow	[-1, 1]
Zakharov	[-5, 10]
Sumsqu	[-5.12, 5.12]
Powell	[-4, 5]

From Table 3 and Table 4, the superior results of the proposed EnvPSO model in tackling HAR tasks indicate the benefits of adding a Gaussian Fitness Surface and nonlinear adaptive coefficients. To re-confirm the observation, we compare this version of EnvPSO with a number of classical search methods and PSO variants using the aforementioned benchmark functions. In addition to original PSO, the following methods are used for comparison, i.e. a modified PSO (MPSO) [102], Enhanced Leader PSO (ELPSO) [103], Dynamic Neighbourhood Learning PSO (DNLPSO) [104], Genetic PSO (GPSO) [105], Dragonfly Algorithm (DA) [106] and Ant Lion Optimisation (ALO) [107]. The settings of these methods are extracted from their original publications shown in Table 10.

Table 10: Experimental settings of the additional baseline methods

Name	Parameter Settings
MPSO [102]	time-varying acceleration coefficients and an adaptive inertia weight factor.
ELPSO [103]	$c_1 = c_2 = 2$, standard deviation of Gaussian mutation=1, scale parameter of Cauchy mutation=2, scale factor of DE-based mutation=1.2, and an adaptive inertia weight factor.
DNLPSO [104]	$c_1 = c_2 = 1.49445$, refreshing gap=3, regrouping period=5, and an adaptive inertia weight factor.
GPSO [105]	maximum velocity=0.6, inertia weight=0.9, acceleration constants $c_1 = 2.6, c_2 = 1.5$, crossover probability = 0.7, mutation probability = 0.3.
DA [106]	alignment factor=0.1, separation factor=0.1, enemy factor=1, cohesion factor=0.7, food factor=1 and an adaptive inertia weight factor.
ALO [107]	Using adaptive parameter settings.

Each search method terminates according to the total number of function evaluations, as defined by $Eval_{max} = population \times iter_{max}$ with $population = 50$ and $iter_{max} = 500$, while $dimension = 30$ is adopted in the experiment. To reduce the effect of random errors and other biases, we repeat each experimental run 30 times.

Table 11 illustrates the mean, minimum, maximum and standard deviation results over a set of 30 runs for all the test functions. As shown in Table 11, EnvPSO outperforms all the methods and achieves the best global minima in all the benchmark functions. The Wilcoxon rank sum test is conducted to evaluate the performance outcome statistically. As shown in Table 12, all the p-values except for two are lower than 0.05, ascertaining the statistically better performance of

Table 11: Evaluation results for the benchmark functions with dimension=30

		EnvPSO	PSO	DA	ALO	MPSO	DNLPSO	ELPSO	GPSO
Ackley	mean	**2.10E+00**	6.18E+00	7.55E+00	1.90E+01	1.72E+01	2.58E+00	1.49E+01	1.72E+01
	min	1.16E+00	3.04E+00	4.27E+00	1.90E+01	1.51E+01	**8.83E-03**	1.11E+01	1.58E+01
	max	**3.09E+00**	9.70E+00	1.19E+01	1.90E+01	1.87E+01	1.14E+01	1.60E+01	1.81E+01
	std	4.94E-01	1.85E+00	1.79E+00	**9.12E-03**	8.10E-01	2.38E+00	9.82E-01	5.57E-01
Dixon-Price	mean	**8.92E-01**	9.80E+00	1.31E+03	1.75E+06	7.92E+05	3.78E+02	1.43E+05	5.02E+05
	min	**6.67E-01**	6.78E-01	1.98E+01	1.07E+06	3.59E+04	6.68E-01	3.57E+04	1.56E+05
	max	**4.35E+00**	9.58E+01	1.14E+04	2.37E+06	1.61E+06	8.65E+03	2.44E+05	7.79E+05
	std	**7.74E-01**	2.65E+01	2.25E+03	2.77E+05	4.61E+05	1.58E+03	5.61E+04	1.46E+05
Griewank	mean	**2.16E-02**	3.59E-01	8.60E+00	5.85E+02	2.90E+02	4.05E+00	1.38E+02	2.87E+02
	min	1.31E-05	2.35E-02	1.90E+00	4.01E+02	1.52E+02	**1.28E-07**	6.44E+01	2.11E+02
	max	**1.00E-01**	1.43E+00	2.44E+01	6.89E+02	4.49E+02	7.61E+01	1.96E+02	3.73E+02
	std	**2.94E-02**	4.42E-01	5.79E+00	6.83E+01	7.19E+01	1.40E+01	2.71E+01	4.25E+01
Rastrigin	mean	**4.20E+01**	6.52E+01	1.18E+02	4.29E+02	3.37E+02	9.78E+01	2.71E+02	3.48E+02
	min	**1.89E+01**	3.09E+01	2.90E+01	3.81E+02	2.50E+02	2.99E+01	2.27E+02	3.10E+02
	max	**8.56E+01**	1.01E+02	2.51E+02	4.78E+02	4.26E+02	1.96E+02	3.20E+02	3.86E+02
	std	**1.46E+01**	1.71E+01	4.68E+01	2.16E+01	4.64E+01	4.49E+01	2.08E+01	1.86E+01
Rothyp	mean	**6.27E-05**	3.05E+00	5.74E+03	3.84E+05	1.89E+05	2.59E+03	9.60E+04	1.75E+05
	min	1.09E-05	8.67E-03	4.16E+02	3.00E+05	8.30E+04	**3.92E-07**	7.81E+04	1.32E+05
	max	**1.77E-04**	8.50E+01	2.08E+04	4.87E+05	3.76E+05	4.37E+04	1.21E+05	2.11E+05
	std	**3.73E-05**	1.55E+01	4.86E+03	4.41E+04	7.30E+04	8.74E+03	1.05E+04	1.98E+04
Rosenbrock	mean	**4.71E+01**	9.75E+01	3.52E+03	1.49E+06	3.37E+05	1.34E+02	1.07E+05	3.45E+05
	min	**6.25E-01**	1.59E+01	1.84E+02	5.76E+05	1.37E+05	2.71E+01	1.98E+04	1.26E+05
	max	**9.23E+01**	1.07E+03	1.87E+04	2.07E+06	6.38E+05	8.44E+02	2.26E+05	4.94E+05
	std	**3.32E+01**	1.88E+02	4.73E+03	3.74E+05	1.33E+05	1.51E+02	4.76E+04	9.16E+04
Sphere	mean	**1.28E-05**	2.66E-02	1.72E+00	1.70E+02	7.87E+01	1.36E+00	4.04E+01	8.97E+01
	min	2.18E-06	5.64E-03	1.97E-01	1.31E+02	2.37E+01	**4.08E-07**	2.11E+01	5.67E+01
	max	**4.84E-05**	8.12E-02	3.78E+00	1.96E+02	1.69E+02	1.92E+01	5.77E+01	1.27E+02
	std	**1.17E-05**	1.80E-02	1.16E+00	1.88E+01	3.36E+01	4.66E+00	8.27E+00	1.71E+01
Sumpow	mean	**3.17E-12**	1.05E-05	2.80E-05	6.65E-01	5.92E-01	1.18E-07	1.34E-02	1.38E-01
	min	3.71E-18	5.66E-07	**1.05E-51**	2.47E-01	3.80E-04	1.71E-21	3.22E-03	1.70E-02
	max	**4.44E-11**	3.49E-05	2.23E-04	1.10E+00	2.00E+00	2.68E-06	3.98E-02	5.32E-01
	std	**8.49E-12**	8.53E-06	5.44E-05	2.08E-01	5.79E-01	4.92E-07	9.32E-03	1.14E-01
Zakharov	mean	**5.30E+01**	1.35E+02	1.67E+02	6.98E+02	3.80E+02	1.16E+02	3.50E+02	4.44E+02
	min	**3.09E+01**	8.46E+01	5.77E+01	5.82E+02	2.61E+02	5.97E+01	2.94E+02	3.79E+02
	max	**7.36E+01**	1.98E+02	2.76E+02	7.50E+02	4.85E+02	2.71E+02	3.85E+02	4.79E+02
	std	**1.16E+01**	3.14E+01	5.62E+01	4.74E+01	4.42E+01	5.08E+01	2.02E+01	2.36E+01
Sumsqu	mean	**3.15E-05**	7.76E-02	3.69E+01	2.40E+03	1.19E+03	7.95E+00	5.55E+02	1.19E+03
	min	6.09E-07	6.33E-03	2.18E+00	1.49E+03	3.43E+02	**3.10E-08**	2.72E+02	7.62E+02
	max	**1.92E-04**	3.81E-01	1.10E+02	2.82E+03	2.20E+03	1.16E+02	8.34E+02	1.54E+03
	std	**4.24E-05**	9.17E-02	2.57E+01	2.94E+02	5.27E+02	2.49E+01	1.15E+02	2.09E+02
Powell	mean	**1.05E-03**	1.13E-01	1.12E+02	1.19E+04	6.10E+03	1.60E+01	1.46E+03	3.72E+03
	min	**3.08E-04**	7.34E-03	4.55E+00	6.51E+03	6.50E+02	2.27E-03	7.97E+02	2.26E+03
	max	**2.32E-03**	4.83E-01	4.35E+02	1.53E+04	1.52E+04	3.02E+02	2.43E+03	5.89E+03
	std	**5.29E-04**	1.24E-01	1.14E+02	2.91E+03	4.32E+03	5.47E+01	4.34E+02	9.82E+02

EnvPSO as compared with those of compared methods. The exceptions are for both Ackley and Rosenbrock landscapes, where the results of EnvPSO are statistically similar to those of DNLPSO and PSO respectively.

Table 12: The Wilcoxon rank sum test results for the benchmark functions over 30 runs

	PSO	DA	ALO	MPSO	DNLPSO	ELPSO	GPSO
Ackley	3.34E-11	3.02E-11	1.72E-12	3.02E-11	7.73E-01	3.02E-11	3.02E-11
Dixon-Price	2.20E-07	3.02E-11	3.02E-11	3.02E-11	4.20E-10	3.02E-11	3.02E-11
Griewank	1.43E-08	3.02E-11	3.02E-11	3.02E-11	2.39E-08	3.02E-11	3.02E-11
Rastrigin	2.00E-06	3.02E-11	3.02E-11	3.02E-11	2.83E-08	3.02E-11	2.02E-08
Rothyp	3.02E-11	3.02E-11	3.02E-11	3.02E-11	2.03E-07	3.02E-11	3.02E-11
Rosenbrock	5.01E-02	3.02E-11	3.02E-11	3.02E-11	1.75E-05	3.02E-11	3.02E-11
Sphere	3.02E-11	3.02E-11	3.02E-11	3.02E-11	1.61E-06	3.02E-11	3.02E-11
Sumpow	3.02E-11	3.02E-11	3.02E-11	3.02E-11	9.76E-10	3.02E-11	1.01E-08
Zakharov	3.02E-11	3.02E-11	2.92E-11	3.02E-11	9.76E-10	3.02E-11	1.33E-10
Sumsqu	3.02E-11	3.02E-11	3.02E-11	3.02E-11	2.19E-08	3.02E-11	3.02E-11
Powell	3.02E-11	3.02E-11	2.92E-11	3.02E-11	3.34E-11	3.02E-11	3.02E-11

3.4 Conclusion

In this section, we conducted a comprehensive evaluation of the proposed ensemble model for Human Action Recognition (HAR) tasks, optimized with Environmental PSO (EnvPSO). Through these experiments, we drew several noteworthy observations.

The ensemble model, comprising three distinct CNN streams, each individually optimized using EnvPSO, displayed notable performance improvements. This highlights the significance of incorporating diversity into feature representations for enhanced HAR accuracy.

EnvPSO, with its adaptive exponential coefficients and Gaussian fitness surface estimation, consistently exhibited advantages over conventional PSO in hyperparameter optimization. The adaptive search strategies played a beneficial role in this regard.

The introduction of the Layer Strip-back parameter provided valuable adaptability in fine-tuning the transfer learning process, contributing to improved generalization capabilities.

The incorporation of both raw images and salient segmented regional images as ”multi-modal” inputs yielded enriched spatial feature representations, resulting in enhanced performance in HAR tasks.

Our exploration extended beyond HAR to benchmark functions, where EnvPSO consistently demonstrated competitive performance.

In conclusion, The proposed ensemble model, refined through EnvPSO, presents a flexible and

promising solution for HAR challenges. Its ability to amalgamate diverse CNN streams with optimized hyperparameters, and leverage multi-modal inputs underscores its potential in HAR and Swarm Optimisation research. Moreover, EnvPSO's effectiveness across domains suggests its potential as a versatile optimization tool for complex problems.

3.5 Limitations of the Work

In spite of the promising findings and innovations presented in this research, it is essential to acknowledge several limitations that warrant consideration:

1. Data Specificity: This work primarily focuses on the evaluation of the proposed ensemble model and Environmental PSO (EnvPSO) optimization on two specific Human Action Recognition (HAR) datasets: Willow7 and BU101. While these datasets are well-established in the field, the findings may not directly extend to other HAR datasets with different characteristics, including variations in resolution, class distributions, or environmental conditions. Generalizing the results to a broader range of datasets requires caution.

2. Computational Demands: The proposed ensemble model and EnvPSO optimization process can be computationally demanding. Training multiple CNN streams and conducting extensive hyperparameter optimization can consume significant computational resources and time. This limitation may restrict its practicality for applications with limited computing resources.

3. Overfitting Risk: Incorporating multiple input modalities can increase the risk of overfitting, particularly when dealing with smaller datasets. Effective regularization techniques and data augmentation strategies are crucial to mitigate this risk.

4. Ensemble Size: The research emphasizes the use of multiple CNN streams in the ensemble. However, determining the optimal ensemble size may vary depending on the dataset and task at hand. Exploring the ideal ensemble size for different scenarios is essential.

5. Comparison Metrics: The evaluation primarily relies on Mean Average Precision (MAP) as the performance metric. A comprehensive assessment should consider additional evaluation metrics and conduct a thorough analysis to provide a holistic view of the model's strengths and limitations.

6. Memory Requirement: Due to the way the prediction surface is modeled it becomes very memory intensive at higher dimensions. This requires careful implementation methods to navigate around.

In light of these limitations, it is important to approach the application of the proposed ensemble model and EnvPSO optimization with careful consideration of the specific dataset, task requirements, and available computational resources. Addressing these limitations and conducting further research in these areas will contribute to a more robust understanding of the applicability and limitations of the proposed approach in HAR and optimization tasks.

Chapter 4

Proposed Neural Inference Search for Multi-loss Segmentation Models

4.1 Introduction

We propose an NIS-optimised CNN model for image segmentation, which consists of two key components, i.e. NIS and a multi-loss function for CNN training. Optimal hyper parameters are first selected using NIS by training CNN with multiple losses combined into a single function. Specifically, we optimise the learning rate, momentum, and loss coefficients α and γ, with respect to balancing between loss effects of the classes that are present and non-present in the GT masks as well as weighting contributions of well/poorly classified examples, respectively. Using the identified hyper parameters, a new optimised CNN model is trained to produce pixel-wise probabilistic class predictions for semantic segmentation. The details of these components are described into the following subsections.

4.2 The Proposed Neural Inference Search Algorithm

The proposed NIS algorithm encompasses three unique search behaviours, i.e. LSTM-CNN based Maximised Standard Deviation Velocity Prediction for global exploration, LSTM-CNN based Local Best Velocity Prediction for search diversification and n-Dimensional Whirlpool search for local exploitation of the optimal regions. A Discrete Adaptive Wave function is formulated to pro-

Algorithm 4 The NIS algorithm

Initialise the swarm size S and particle positions
Initialise c_{d1} and c_{d2} using Equation 4.11
Initialise c_{u1} using Equation 4.12
Initialise Velocity Prediction Model (VPM)
Initialise training data array $\mathbf{A}_{data}$
while $t < T$ **do**
  Update θ_t using Equations 4.9-4.10
  Update $\mathbf{R}_\theta$ using Equation 4.8
  Collect input data from particle positions and fitnesses in array $\mathbf{A}_{input}$
  Get VPM predictions M from $\mathbf{A}_{input}$
  Update $\vec{u}_\sigma^t$ from M_0
  Update $\vec{u}_\beta^t$ from M_1
  Initialise target data array $\mathbf{A}_{gt}$
  for each particle $i = 1, ..., S$ **do**
    Update $\vec{u}_\theta^t$ using Equation 4.5
    Update velocity $\vec{v'}_i^{t+1}$ using Equation 4.1
    Update position $\vec{x}_i^{t+1}$ using Equation 4.2
    if $f(\vec{x}_i^{t+1}) < f(\vec{p}_{best_i}^t)$ **then**
      $\vec{p}_{best_i}^t = \vec{x}_i^{t+1}$
    end if
    if $f(\vec{x}_i^{t+1}) < f(\vec{g}_{best}^t)$ **then**
      $\vec{g}_{best}^t = \vec{x}_i^{t+1}$
    end if
    Generate and append target velocity vectors to $\mathbf{A}_{gt}$
  end for
  Combine $\mathbf{A}_{input}$ and $\mathbf{A}_{gt}$ and append to A_{data}
  Train VPM with A_{data}
end while
return $\vec{g}_{best}^t$

vide different emphasis of these three search behaviours at different search stages. The proposed velocity and position operations combining the above three search mechanisms scheduled by the Discrete Adaptive Wave function are defined in Equations 4.1 and 4.2, respectively.

$$\vec{v'}_i^{t+1} = r_1 c_{d1}(t)\vec{u}_{\sigma_i}^t + r_2 c_{d2}(t)\vec{u}_{\beta_i}^t + r_3 c_{u1}(t)\vec{u}_{\theta_i}^t \tag{4.1}$$

$$\vec{x}_i^{t+1} = \vec{x}_i^t + \vec{v'}_i^{t+1} \tag{4.2}$$

where $\vec{v'}_i^{t+1}$ and $\vec{x}_i^{t+1}$ define the velocity and position vectors of the i-th particle in the $t+1$-th iteration, respectively. In Equation 4.1, the velocity update operation consists of three behavioural terms as mentioned above. Specifically, the first component, i.e. $r_1 c_{d1}(t)\vec{u}_{\sigma_i}^t$, deals with exploration led by Maximised Standard Deviation Velocity $\vec{u}_{\sigma_i}^t$ to increase search territory. The second

term, $r_2 c_{d2}(t)\vec{u}^t_{\beta_i}$, manages local exploration/exploitation of promising optimal regions guided by Local Best Velocity $\vec{u}^t_{\beta_i}$, while the third term, i.e. $r_3 c_{u1}(t)\vec{u}^t_{\theta_i}$, provides an exploitation mechanism to emphasize search intensification of well-established optimal regions using an angle-driven search velocity $\vec{u}^t_{\theta_i}$. The first two vectors, i.e. the Maximised Standard Deviation Velocity $\vec{u}^t_{\sigma_i}$ and Local Best Velocity $\vec{u}^t_{\beta_i}$, are derived from the LSTM-CNN predictions, with the third being determined by a novel n-dimensional spatial spiral algorithm. These behavioural terms also contain a scheduling factor (c_{d1}, c_{d2} or c_{u1}) implemented by the Discrete Adaptive Wave function. The aim is to determine the overall velocity contributions of the associated behavioural terms in regard to the current iteration. In addition, parameters r_1, r_2 and r_3 contained in these terms are random scalar factors sampled from the uniform distribution $U(0.5, 1.5)$. They provide variations in the distance traveled between particles within the same iteration. As indicated in Equation 4.2, after defining $\vec{v'}_i^{t+1}$, the particle's next position can be found by simply adding the velocity to the current particle position.

The surrounding context of these velocity and position update operations is shown in Algorithm 4, the details of which are discussed in the following subsections.

4.2.1 Velocity Prediction Using a Neural Network Model

To generate the velocity vectors $\vec{u}_\sigma$ and $\vec{u}_\beta$, we employ an LSTM-CNN style network, named the VPM, as shown in Figure 12. Specifically, two LSTM modules in the LSTM-CNN network are firstly employed to extract the sequential information from the network inputs, i.e. historical information of the particle position and the associated fitness information defined as A_{input}. A convolutional layer is subsequently applied to tackle spatial relationships between the particles. Fully connected linear layers are used to make the final predictions through a sigmoid layer to constrain the values between 0 and 1. Using this architecture, the VPM with an LSTM-CNN architecture simultaneously predicts the $\vec{u}_\sigma$ and $\vec{u}_\beta$ vectors for each particle as M_0 and M_1 of a single matrix M with shape $(2, \mathrm{S}, \mathrm{D})$ where S is the swarm size and D is the dimensionality of the search space.

VPM training for velocity prediction with respect to global exploration is conducted from scratch, collecting and storing data samples at every iteration as A_{data}. Initially, the training data are sparse and the predictions rely on a few samples to produce the velocity vectors. As the search progresses,

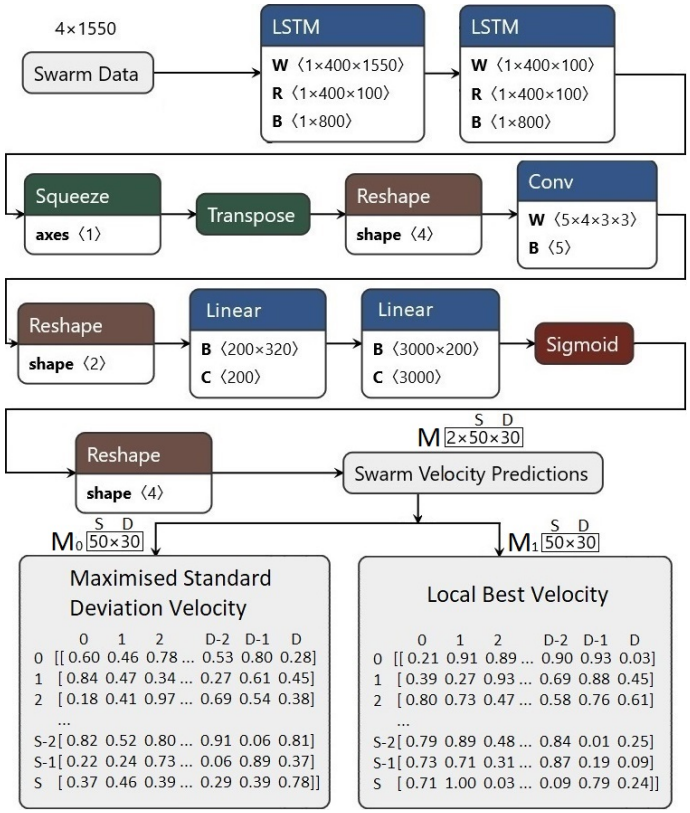


Figure 12: The VPM predicts velocity vectors for a number of particles (S) constrained within a D-dimensional search space. The Swarm Velocity Predictions M are split into M_0, i.e. the Maximised Standard Deviation Velocity predictions at index 0 axis 0, and M_1, i.e. the Local Best Velocity predictions at index 1 axis 0.

predictions improve since network training is conducted with comparatively more data collected at each iteration. Specifically, A_{input} is collected at the beginning of the loop where the VPM's input data consist of the previous particle positions from the last four iterations with their associated fitness scores, forming a tensor of shape $(1, 4, (\mathrm{D}+1) \times \mathrm{S})$. GTs for the target predictions of $\vec{u}_\sigma$ and $\vec{u}_\beta$ are stored in A_{gt}. Once the GTs of particle's velocity vector have been collected, A_{input} and A_{gt} are combined into a single data sample with shape $(2, \mathrm{S}, \mathrm{D})$ and stored in A_{data}. This growing data set serves to train the VPM before velocity prediction starts in the next iteration. The next two sections detail the GT collection processes for $\vec{u}_\sigma$ and $\vec{u}_\beta$, respectively.

4.2.2 Maximised Standard Deviation Velocity

The main intention of the Maximised Standard Deviation Velocity ($\vec{u}^t_{\sigma_i}$) in Equation 4.1 is to ensure the searched territory of the swarm sufficiently encompasses the entire search space in a distributed manner. As such, promising areas for future exploitation can be identified. This is

achieved by predicting global and local search velocity vectors using the VPM. To obtain the GT velocity vector required for prediction of $\vec{u}_{\sigma}^{t}$, we compare the standard deviation of the particle positions in the current iteration with that from the previous iteration. If the standard deviation of the current iteration is higher, then a vector based on the difference between the particle's current position and its most distant previous position in the last four iterations is generated. Otherwise, a mirrored vector is yielded. This is repeated for each particle, yielding a GT that corresponds to the predictions defined as M_0 in Figure 12, providing a tensor of shape (S, D). The previous particle positions from the last four iterations in conjunction with these GT velocity vectors serve as inputs and outputs respectively to train the VPM as described in the previous subsection; enabling Maximised Standard Deviation Velocity prediction. To extract Maximised Standard Deviation Velocity predictions, the VPM swarm prediction matrix M is indexed at 0 at axis 0 (M_0) yielding $\vec{u}_{\sigma}^{t}$, the predictions for every particle at iteration t, as indicated in Equation 4.3. This LSTM-CNN based Maximised Standard Deviation Velocity prediction guides the global search process starting from scratch and generates increasingly improved predictions to inform diversification of the swarm. As the search progresses, the predictions result in particles spreading out over the search space as indicated in Figure 13.

$$\vec{u}_{\sigma}^{t} = M_0 \tag{4.3}$$

4.2.3 Local Best Velocity

The second proposed search operation is the VPM based Local Best Velocity prediction ($\vec{u}_{\beta_i}^{t}$). This operation is conducted using the same VPM network, thus employs the same input training data (i.e. the previous particle positions from the last four iterations) as those used for Maximised Standard Deviation Velocity Prediction. The velocity vector target prediction aims to accelerate particle movements towards personal best positions, instead of global exploration as indicated in the first proposed action. The target prediction is the vector difference from the most recent best particle position and the most recent worst particle position. Target velocity vector predictions are collected for each particle giving a shape of (S, D) and then used for Local Best Velocity prediction training via the M_1 output tensor of the VPM. The rationale behind this method is that the network can predict vectors that lead to the local optimal position of each particle across the search space, allowing the swarm to both explore and exploit distinct regions of the search space

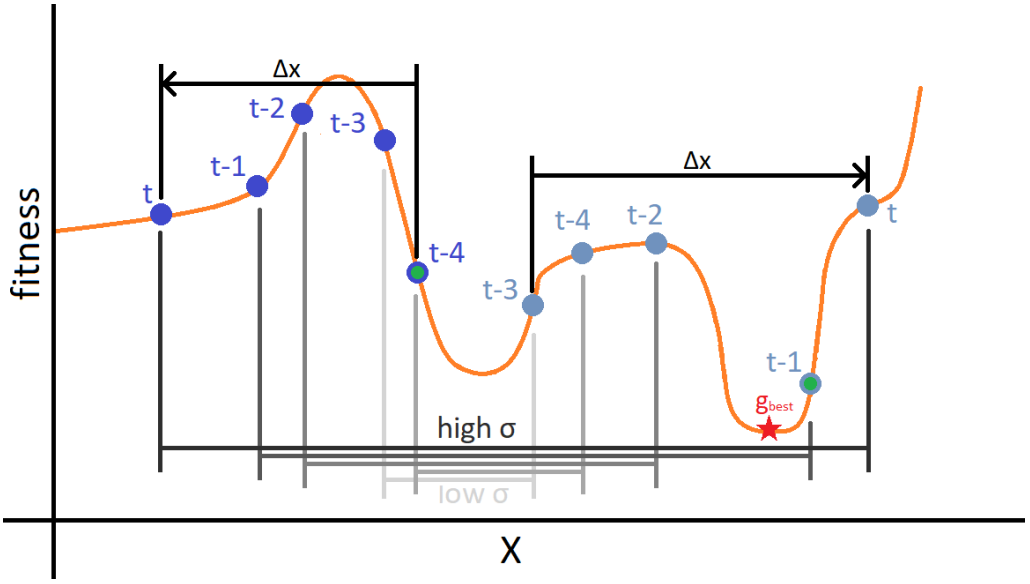


Figure 13: Since the standard deviation of the particle positions at t is higher than those at $t-1$, Δx is taken for each particle as velocity vectors for the Maximised Standard Deviation Velocity GT target prediction. By training on the these GT targets, the VPM causes particles to spread out in the search space (one-dimensional in this graph). Δx is the difference between the current particle position at iteration t and the most distant previous particle position within $t-4$ iterations.

independently. This process is shown in Figure 14. Similar to Maximised Standardized Deviation Velocity, Local Best Velocity predictions are taken from the VPM from the second index of M at axis 0 (M_1) yielding $\vec{u}_{\beta}^{t}$, the Local Best Velocity Prediction for every particle at iteration t, as indicated in Equation 4.4.

$$\vec{u}_{\beta}^{t} = M_1 \tag{4.4}$$

4.2.4 n-Dimensional Whirlpool search

The proposed Whirlpool search is an n-dimensional spiral-like search. It provides exploitation of promising areas in the search space by applying an iteratively decreasing angular rotation about a unit direction vector pointing toward the global best solution, as indicated in Equation 4.5.

$$\vec{u}_{\theta_i}^{t} = \vec{u}_{i}^{t} \cdot \mathbf{R}_{\theta}^{\intercal} \tag{4.5}$$

where $\mathbf{R}_{\theta}^{\intercal}$ is a transposed rotation matrix for rotating vectors by θ, $\vec{u}_{i}^{t}$ is the non-rotated vector

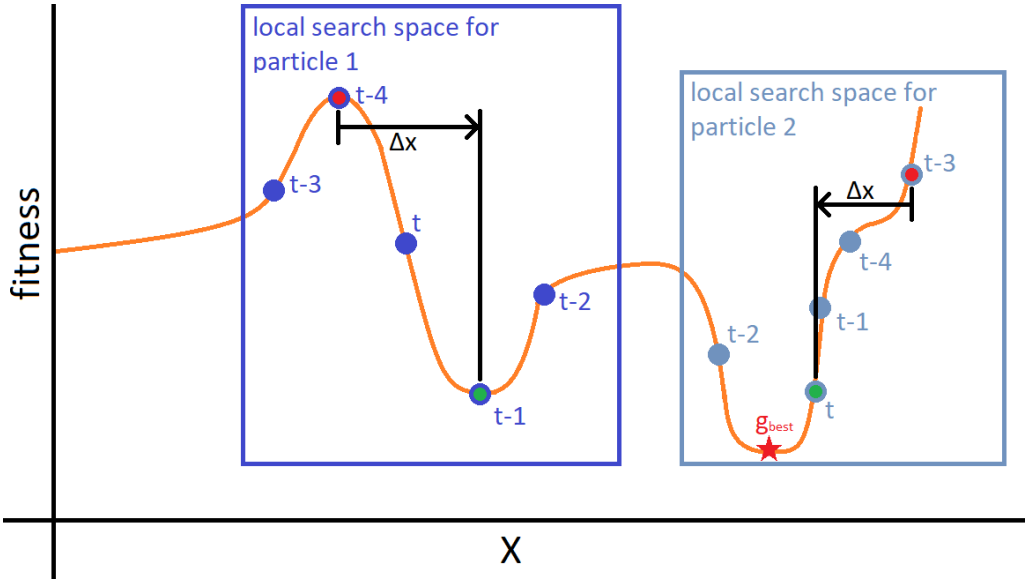


Figure 14: The influence of the VPM accelerates the particles to move towards personal best positions where t denotes the current iteration. The two boxes show two particles searching independently and how the velocity vectors (Δx) are collected and used as the training data in a one-dimensional search space.

pointing from the i-th particle position toward the global best position $\vec{g}_{best}^{t}$ at iteration t, and $\vec{u}_{\theta_i}^{t}$ being the resultant rotated vector. An example resultant rotation is displayed in Figure 15.

To increment particle positions towards the global best position, we define a direction vector, $\hat{u}_i^t$, with a variable magnitude which decreases over time to perform finer movements towards the end of the search. The largest possible movement is defined as the magnitude of the vector spanning from opposite corners of the search space ($\|\vec{b}_{up} - \vec{b}_{low}\|$). This magnitude is decreased by a cosine-based factor dependent on the maximum and current iterations. When combined with the direction vector $\hat{u}_i^t$, the complete angular rotation velocity vector ($\vec{u}_i^t$) is produced. This is formally expressed in Equation 4.6.

$$\vec{u}_i^t = \cos\left(\frac{t}{2T}\pi\right)\hat{u}_i^t \left\|\vec{b}_{up} - \vec{b}_{low}\right\| \tag{4.6}$$

where $\cos(\frac{t}{2T}\pi)$ is a decreasing factor. The direction vector $\hat{u}_i^t$ is obtained from the initial vector $\vec{u}_i^t$ by the division of its magnitude as in Equation 4.7.

$$\hat{u}_i^t = \frac{\vec{g}_{best}^{t} - \vec{x}_i^t}{\|\vec{g}_{best}^{t} - \vec{x}_i^t\|} \tag{4.7}$$

where $\vec{g}_{best}^{t}$ and $\vec{x}_i^t$ are the global best position and the i-th particle position at iteration t re-

spectively. Note that $\left\|\vec{g}_{best}^{t} - \vec{x}_{i}^{t}\right\|$ indicates the magnitude of the difference between $\vec{g}_{best}^{t}$ and $\vec{x}_{i}^{t}$.

As mentioned previously, the transposed rotation matrix $\mathbf{R}_{\theta}^{\intercal}$ defined in Equation 4.5 enables n-dimensional rotation of the initial vector $\vec{u}_{i}^{t}$ by a given angle θ. A new rotation matrix $\mathbf{R}_{\theta}$ is created at each iteration using two orthonormal vectors ($\hat{n}_1$ and $\hat{n}_2$) obtained through Gram-Schmidt Orthogonalization, as indicated in Equation 4.8.

$$\begin{aligned}\mathbf{R}_{\theta} = \mathbf{I} &+ \hat{n}_2 \otimes \hat{n}_1 - \hat{n}_1 \otimes \hat{n}_2 \sin\theta \\ &+ \hat{n}_1 \otimes \hat{n}_1 + \hat{n}_2 \otimes \hat{n}_2(\cos\theta - 1)\end{aligned} \tag{4.8}$$

with **I** being an identity matrix whose rows and columns equal to the dimensionalities of the search space, and $\otimes$ is the outer product operation. Besides that, θ is a dynamic angular value moving from $\frac{\pi}{2}$ to 0 in decreasing steps as shown in Equation 4.9.

$$\theta_t = \frac{\pi}{2}\left(0.8 - \frac{t}{T}\right) \tag{4.9}$$

where T is the total number of iterations. The factor of 0.8 ensures that θ has negative values in the last few iterations. This value is clipped to stay at 0 using Equation 4.10, ensuring no rotation occurs toward the very end of the search leading to a linear global best search.

$$\theta_{t+1} = \begin{cases} 0, & \text{if } \frac{\theta_t}{\pi} < 0 \\ \theta_t, & \text{otherwise} \end{cases} \tag{4.10}$$

This n-Dimensional Whirlpool search conducts angle-driven granular movements to exploit optimal regions around the global best solution to increase the chances of finding global optima.

4.2.5 Staged Discrete Adaptive Wave function

To maximise the exploration and exploitation capabilities of all three behavioural terms present in NIS (as defined in Equation 4.1), we introduce a function which adapts the contribution of each behavioral term based on sequential iteration ranges. These ranges, referred to as stages, allow each behaviour to be configured with a weighting factor in each stage to increase or decrease its contribution. This enables bespoke macro behaviours to be created in each defined stage.

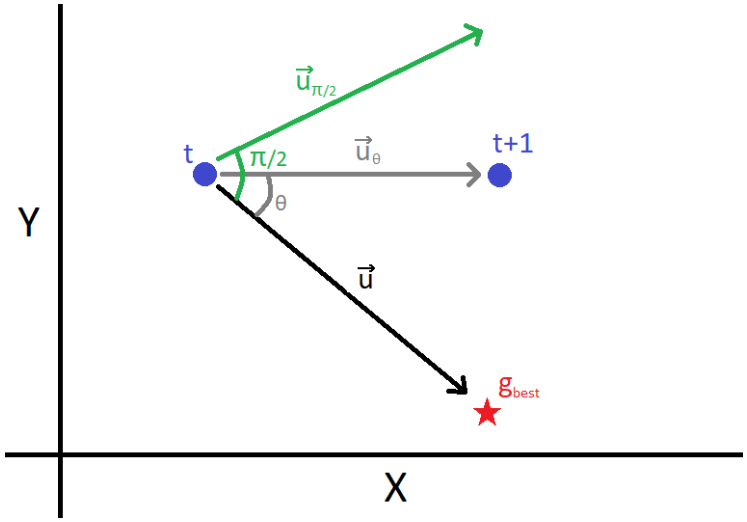


Figure 15: g_{best} refers to the global best position and t represents the current iteration. The angle θ slowly decreases from $\frac{\pi}{2}$ (green line) to 0 (black line). $\vec{u}$ indicates the initial direction vector and $\vec{u}_\theta$ is $\vec{u}$ rotated by θ. The magnitude of $\vec{u}$ is defined in Equation 4.6. The two-dimensional case is displayed here but Equations 4.5-4.10 show the generalisation to n dimensions.

Additionally, behavioural contributions to the overall velocity are increased or decreased via a sinusoidal function according to whether they are exploitative or exploratory, respectively. This ensures exploratory behaviours have a high contribution at the beginning of the search and no contribution near the end of the search, whereas exploitative behaviours do the opposite. Details of these mechanisms are shown in Equations 4.11 and 4.12.

$$c_d = \frac{1}{2}(1 + cos(\frac{t}{T}\pi))\Phi(t, T, s_1, s_2, s_3) \tag{4.11}$$

$$c_u = \frac{1}{2}(1 - cos(\frac{t}{T}\pi))\Phi(t, T, s_1, s_2, s_3) \tag{4.12}$$

where c_d and c_u are the functions for producing the increasing or decreasing behavioural term coefficients c_d1, c_d2, and c_u1 seen in Equation 4.1. $\frac{1}{2}(1 - cos(\frac{t}{T}\pi))$ and $\frac{1}{2}(1 + cos(\frac{t}{T}\pi))$ are the decreasing and increasing sinusoidal factors and Φ is the Discrete Adaptive Wave Function. These equations are displayed in Figure 16. The Discrete Adaptive Wave Function Φ is constructed through the addition of n_{th} summations of ψ which produces a function with discrete weighted stages to schedule NIS behaviours based on the current iteration t of the search algorithm. In Equation 4.13, we define three stages starting from 0 to the maximum iteration T in increments of $\frac{1}{3}T$.

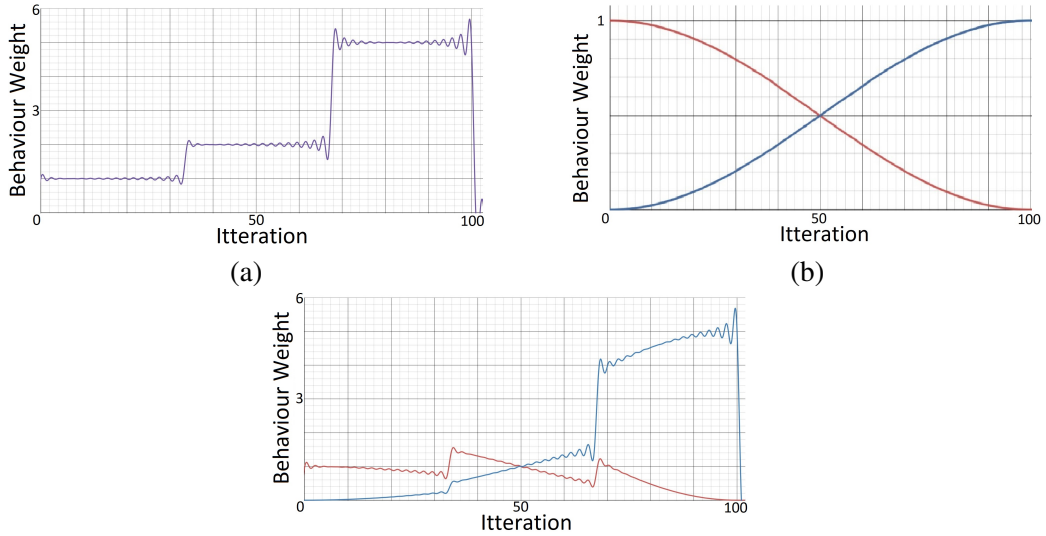


Figure 16: (a) Equation 4.13 where $s_1 = 1$, $s_2 = 2$, $s_3 = 5$ and $T = 100$. (b) Factors $(1 + cos(\frac{t}{T}\pi))$ (red) and $(1 - cos(\frac{t}{T}\pi))$ (blue) from Equations 4.11 and 4.12 respectively. (c) Equations 4.11 (red), and 4.12 (blue), i.e. the combination of (a) and (b).

$$\Phi(t, T, s_1, s_2, s_3) = \sum_{n=0}^{\frac{1}{3}T} \psi(n, t, s_1) + \sum_{n=\frac{1}{3}T+1}^{\frac{2}{3}T} \psi(n, t, s_2) + \sum_{n=\frac{2}{3}T+1}^{T} \psi(n, t, s_3) \tag{4.13}$$

Each stage has a corresponding weighting coefficient s_1, s_2 or s_3, which can be adjusted to increase or decrease the contribution of a particular behaviour depending on the iterations falling within a given stage. The configurations of these weightings for the proceeding sections are displayed in Table 13.

Table 13: Φ Settings

Coefficient	Behaviour	s_1	s_2	s_3
c_{d1}	$\vec{u}^t_{\sigma_i}$ Maximised Standard Deviation Velocity Prediction	1.0	0.5	0.0
c_{d2}	$\vec{u}^t_{\beta_i}$ Local Best Velocity Prediction	0.5	1.0	0.0
c_{u1}	$\vec{u}^t_{\theta_i}$ Whirlpool Search	0.0	0.5	1.0

where ψ is a function built upon $sinc$, often found in analogue to digital signal conversion. We

adopt this function as shown in Equation 4.14 for use with the discrete summation to enable an iteration-based scheduling as shown in Equation 4.13.

$$\begin{aligned} \psi(n,t,s) &= s \times \text{sinc}(\pi(t-n)) \\ &= s \times \frac{\sin(\pi(t-n))}{\pi(t-n)} \end{aligned} \tag{4.14}$$

As indicated in the velocity and position formulae in Equations 4.1 and 4.2, the Scheduled Adaptive Coefficients (c_{d1}, c_{d2}, and c_{d3}) are combined with the three previously defined behaviours ($\vec{u}^t_{\sigma_i}$, $\vec{u}^t_{\beta_i}$, and $\vec{u}^t_{\theta_i}$) from Equations 4.3, 4.4, and 4.5, respectively. The full algorithm of NIS is indicated in Algorithm 4. Owing to this Discrete Wave function, the proposed algorithm employs an adaptive emphasis of the aforementioned three search behaviours driven by neural network based velocity prediction and angle rotation-based search movement to balance between diversification and intensification.

4.3 The Proposed Multi-Loss Function

There are several popular loss functions for image segmentation. The Cross Entropy loss is commonly used for measuring the difference between GT and predicted masks. The Soft Dice loss adopts a Sørensen–Dice coefficient to measure similarity between two samples. The Focal loss is an adaptation of the Cross Entropy loss which embeds mechanisms to adjust the impact of loss contributions of well and poorly classified examples as well as those between the classes present and non-present in the GT masks.

To take advantage of the loss information from the aforementioned variants, we propose a new multi-loss function as shown in Equation 4.15. It combines the complementary error signals from Cross Entropy, Dice, and Focal Loss schemes. Such a multi-loss mechanism is able to provide compound loss indicators to advise the backpropagation process and adjust performance. In particular, to balance the effects of the well/poorly classified examples and contributions of the classes present/non-present in the GT masks, we optimise the γ and α coefficients in the focal loss function using the proposed NIS algorithm.

$$\text{ML} = \text{FL}(\gamma, \alpha) + \text{SDS} + \text{CE} \tag{4.15}$$

This approach simplifies the complexity of training machine learning models by combining distinct loss functions into a single scalar objective, a strategy that has become popular due to its practical benefits. This process shares similarities with the concept of Pareto optimality often seen in multi-objective optimization.

During backpropagation, gradients are computed with respect to the combined loss. These gradients, originating from the sum of individual loss gradients, act as guiding vectors in the high-dimensional parameter space, directing the model towards configurations that minimize the composite loss. This pursuit of optimality simultaneously respects the diverse objectives contained within the combined loss.

For example, when summing Focal, Cross Entropy, and Soft Dice Score losses, you harmoniously integrate various optimization criteria. These objectives interact intricately during backpropagation, where gradients, reflecting the local slope of each component's loss landscape, collaboratively steer parameter updates. Gradients from one loss function can influence updates in parameters that primarily affect other objectives. This dynamic interaction results in a delicate balancing act, where the optimization process continually strives to improve multiple objectives at once.

Importantly, this summation isn't a simple linear aggregation but a complex, non-linear interaction. Depending on how objectives interact, it can lead to scenarios where enhancing one objective might come at the expense of another, similar to trade-offs in multi-objective optimization. However, it also creates opportunities for synergy, where considering multiple objectives simultaneously enhances overall model performance.

Summing loss functions in this manner constructs a multi-dimensional optimization landscape, guiding the backpropagation process as it navigates the parameter space in pursuit of an optimal solution that balances competing objectives. Within this interplay, the model gains the ability to comprehensively address distinct optimization criteria while optimizing a unified scalar objective.

Specifically, the α loss coefficient balances the influence between the classes that are present and non-present in the GT masks at the pixel level. A higher α emphasizes the classes present in the GT mask, while a lower value shifts the emphasis towards those that are not present. Each column

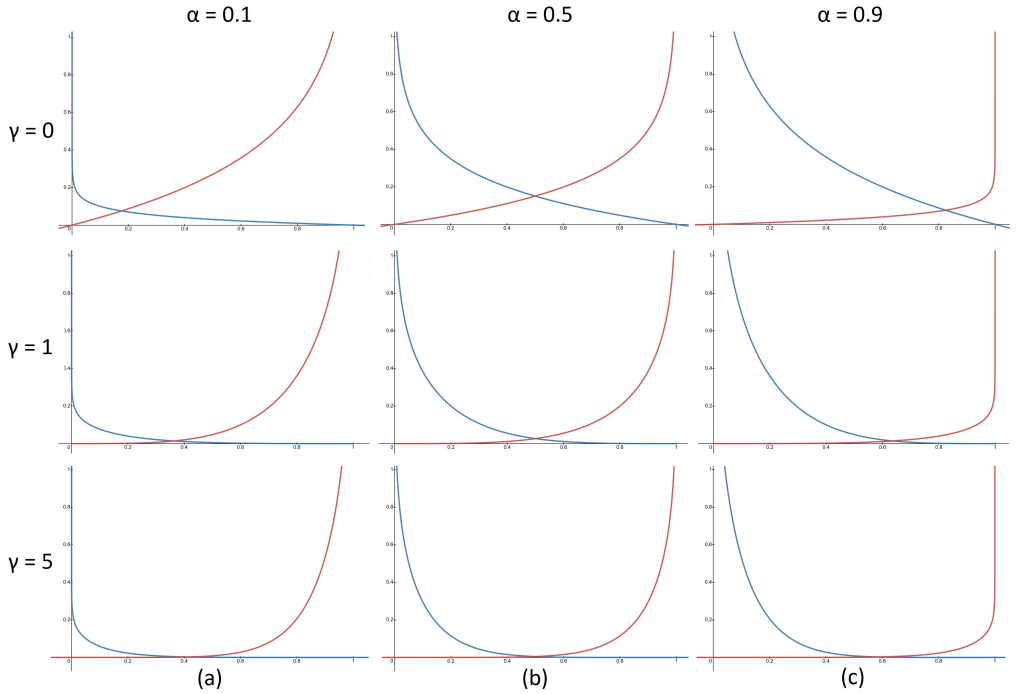


Figure 17: The blue and red lines respectively show the loss contributions of the classes that are present and non-present in the GT masks, respectively, for the Focal Loss. The graphs are arranged into three columns with varied α values and three rows with different γ values showing the resultant loss curves.

in Figure 17 indicates the impact of different α settings, where the blue and red lines show the loss contributions of the classes that are present and non-present in the GT masks, respectively. In each row, the loss graphs are generated using a fixed γ value (i.e. $\gamma = 0, 1,$ or 5), with (a) $\alpha = 0.1$, (b) $\alpha = 0.5$, and (c) $\alpha = 0.9$. To be specific, (a) indicates higher contributions of the classes that are not present in the GT masks, and (b) shows the balanced effects of both classes that are present and non-present, while (c) implies higher impact of the classes that are present.

The γ loss coefficient balances between pixels with true positive/true negative predictions and those with false positive/false negative predictions. We refer to classes with considerable true positive and true negative predictions as well classified, and those with substantial false positive and false negative predictions as poorly classified. A higher γ emphasizes the contribution of the well predicted classes at the pixel level, whereas a lower γ setting increases the contribution of the poorly classified classes by reducing the influence of well classified ones. Each row in Figure 17 shows the effects of different γ configurations.

As indicated in Figure 17, different settings of α and γ loss coefficients play significant roles

in the resultant loss function behaviours defined in Equation 4.15. We optimise these two hyper parameters along with the learning rate and momentum settings using the proposed NIS algorithm to further fine-tune model learning behaviours.

4.4 Experimental Studies

We employ three well-known semantic segmentation data sets, as well as mathematical numerical test functions to evaluate the proposed model against several baseline search methods.

4.4.1 Segmentation Data sets

We employ three data sets, i.e. CamVid, Freiburg and MESSIDOR, for evaluating segmentation models. These data sets are detailed in the following subsections.

The CamVid data set is a collection of images belonging to one of five video sequences taken from a car dashboard camera while driving. A total of 701 images have a resolution of 960x720 pixels. An official data split is provided comprising 369 training, 100 validation and 232 test samples which are used for evaluation. The GT annotations for each image have been created by manually assigning each pixel to one of 32 classes. To reduce the memory requirements and computational complexity, these 32 classes are compressed into 12 semantically similar classes as indicated below, i.e. Void, Sky, Building, Pole, Road, Pavement, Tree, SignSymbol, Fence, Car, Pedestrian, and Bicyclist.

The Freiburg Forest data set contains 366 images from a robot mounted camera as it navigates through a forest. It contains RGB, Depth, NIR, NRG, NDVI and EVI images with GT annotations with a resolution of 882x490. This study uses the RGB images and GT masks with classes of object, trail, grass, tree, vegetation and sky. The tree and vegetation classes are combined since the latter is not used in the test set. An official split of 230 training and 136 test samples is provided.

The MESSIDOR data set contains 1200 colour retinal images and binary segmentation masks at multiple resolutions (1440x960, 2240x1488, and 2304x1536) for segmentation and detection of optic discs. This study resizes GTs and images to 640x480 using an 80-20 train-test split.

4.4.2 Loss Function Evaluations

We explore NIS-devised networks in combination with different loss functions and indicate efficiency of the proposed combined multi-loss scheme.

To be specific, each model was trained with one of the existing three loss functions, i.e. Cross Entropy, Dice, and Focal loss, commonly used for semantic segmentation tasks, as well as a diverse combination of these loss functions, including the newly proposed one. The performance of these loss functions was evaluated across seven established segmentation models, i.e. FCN [79], DeeplabV3 [42], Unet++ [80], Linknet [108], LR-ASPP[81], MAnet[41], and PSPnet[109]. Each model was trained for 50 epochs using SGD with a learning rate of 0.01 and a momentum of 0.5. A train-validation split was taken from the original training sets to preserve the true test data. Four commonly used segmentation metrics were calculated from the test set for comparison, i.e., Dice Score, Global Accuracy, mIoU and Mean Class Accuracy (MCA). These results can be found in Table 14.

Loss Function Comparison

We present the segmentation results for the CamVid data set using each of the three loss functions in Table 14. As indicated in Table 14, the top ten results for each metric for this data set are mostly distributed between FCN, DeeplabV3 and Unet++ with the remaining good results being attained by MAnet. The top result (63%) for the Dice score is obtained by DeeplabV3 using the Dice loss function. For the global accuracy rates, the top result (87.6%) is shared jointly between FCN with Cross Entropy loss and DeeplabV3 with Focal loss. The DeeplabV3 with Dice loss model achieves both the highest mIoU (52.4%) and MCA (63.3%) scores. It is clear that the most consistently accurate models are DeeplabV3, FCN and Unet++. Regarding loss functions, models trained with Dice loss often provide the highest results. To further assess the effectiveness of the combination of different loss functions, we use the three best-performing networks for subsequent experiments.

Multi-Loss Function Results

Since each of the previously used loss functions capture unique aspects of the error present in the data set, further investigation was conducted to determine if the benefits provided by each loss

Table 14: The results for Dice Score, Global Accuracy (GA), mIoU and MCA for models trained with different losses on the CamVid data set with the top 10 results highlighted

Model	Cross Entropy	Dice	Focal	Dice (%)	GA (%)	mIoU (%)	MCA (%)
FCN	✓	×	×	**61.1**	**87.6**	**50.2**	**62.7**
	×	✓	×	**62.0**	**86.8**	**51.3**	**62.4**
	×	×	✓	**61.3**	**86.6**	**50.5**	**63.1**
DeeplabV3	✓	×	×	**57.7**	**84.6**	**46.3**	**59.6**
	×	✓	×	**63.0**	**87.5**	**52.4**	**63.3**
	×	×	✓	**60.6**	**87.6**	**49.8**	**63.0**
Unet++	✓	×	×	**54.3**	81.4	**43.5**	**57.8**
	×	✓	×	**55.2**	79.6	**44.3**	**56.3**
	×	×	✓	**55.0**	**83.8**	**44.8**	**58.1**
Linknet	✓	×	×	43.1	83.5	34.5	49.0
	×	✓	×	33.4	79.3	27.5	38.2
	×	×	✓	36.8	81.8	30.4	45.2
LR-ASPP	✓	×	×	41.3	71.5	31.1	42.4
	×	✓	×	41.6	78.2	32.6	42.5
	×	×	✓	42.9	79.7	33.6	44.6
MAnet	✓	×	×	47.0	**84.1**	38.1	53.2
	×	✓	×	**51.3**	**84.6**	**42.2**	**53.7***
	×	×	✓	47.9	**85.5**	38.8	**53.7***
PSPnet	✓	×	×	39.7	58.4	28.5	37.8
	×	✓	×	44.0	63.9	32.2	42.0
	×	×	✓	39.8	58.7	28.7	38.0

* represents a shared tenth place result

function can be exploited simultaneously. Toward this end, the three best-performing networks, i.e., DeeplabV3, FCN, and Unet++, were trained with every combination of loss functions on the CamVid data set using the same training regimen discussed previously. These results are provided in Table 15. The results indicate that models trained with all three loss functions typically produce the top results, otherwise yielding the second best results. The lowest results are almost consistently associated with models trained with single loss functions. Of these models, DeeplabV3 and FCN networks typically outperformed Unet++ with higher metric scores, leading to top results. These observations justify using DeeplabV3 and FCN and the combined multi-loss function for further evaluation of NIS optimisation on CNN segmentation models.

Table 15: Dice Score, Global Accuracy (GA), mIoU and MCA for models trained with different combinations of losses on the CamVid Data set

Model	Cross Entropy	Dice	Focal	Dice (%)	GA (%)	mIoU (%)	MCA (%)
FCN	✓	×	×	61.1	87.6	50.2	62.7
	×	✓	×	62.0	86.8	51.3	62.4
	×	×	✓	61.3	86.6	50.5	63.1
	✓	✓	×	63.6	87.6	52.8	64.5
	✓	×	✓	61.5	87.2	50.6	63.2
	×	✓	✓	**64.4**	88.1	53.6	**66.0**
	✓	✓	✓	64.3	**89.2**	**53.8**	65.6
DeeplabV3	✓	×	×	57.7	84.6	46.3	59.6
	×	✓	×	63.0	87.5	52.4	63.3
	×	×	✓	60.6	87.6	49.8	63.0
	✓	✓	×	63.5	87.8	52.6	65.7
	✓	×	✓	61.4	87.9	52.9	66.4
	×	✓	✓	63.2	87.4	52.2	65.0
	✓	✓	✓	**64.3**	**89.5**	**53.7**	**67.6**
Unet++	✓	×	×	54.3	81.4	43.5	57.8
	×	✓	×	55.2	79.6	44.3	56.3
	×	×	✓	55.0	83.8	44.8	58.1
	✓	✓	×	61.8	83.1	50.0	**64.2**
	✓	×	✓	57.7	84.3	46.4	60.5
	×	✓	✓	60.0	81.4	47.6	61.4
	✓	✓	✓	**62.3**	**86.2**	**51.0**	**64.2**

4.4.3 Segmentation Model Evaluation

In this section, we employ the proposed NIS model for hyper parameter identification of the best-performing networks, i.e. DeeplabV3 and FCN. In particular, the multi-loss function identified earlier is used as the fitness function, which integrates Cross Entropy, Dice, and Focal loss measures. FA and PSO are utilized as the baseline methods for optimal hyper parameter selection, across the three test data sets. The experimental setup is firstly explained. We then analyze the results and discuss notable patterns with respect to each data set. A summary of the combined results from all data sets is also provided. The selected hyper parameters are presented and discussed with regard to their effect on model performance.

To evaluate each segmentation network, NIS, PSO and FA are employed to identify the optimal settings of learning rate, momentum, γ and α in the multi-loss function. In particular, γ and α are the loss coefficients for the Focal loss function, as discussed earlier. The identified hyper parameters are then used to train the model on the combined training and validation sets, before

being evaluated with the test set. The optimal hyper parameter identification process is performed five times. We present and analyze the mean results over five runs for DeeplabV3 and FCN with respect to each data set in Subsection 4.4.3.

As a reference, the default results of both DeeplabV3 and FCN without hyper parameter optimisation are also provided. In the default experimental settings, instead of the multi-loss function, a standard Cross Entropy loss is used. The networks are trained with an SGD optimiser with a default learning rate of 0.01 and a default momentum of 0.5. The final results of each network are obtained by taking the average of five runs.

The following settings remain constant throughout all experiments. All algorithms use a population of 10, a maximum iteration of 20 and a set of 5 runs. Each fitness evaluation trains each CNN model for two epochs before evaluation. The search ranges for the optimisation targets are shown in Table 16. The devised DeeplabV3 and FCN models with optimal settings are both trained with the SGD optimiser for 50 epochs, along with a weight decay of 0.005 and a batch size of 4.

Table 16: hyper parameters targeted for optimisation

Hyper-Paremeter	Lower Bound	Upper Bound
Learning Rate	0.0001	0.01
Momentum	0.0	1.0
α (loss parameter balancing between the classes that are present and non-present in the GT masks)	0.15	0.99
γ (loss coefficient weighting contributions of well and poorly classified examples)	1.00	5.0

Results

The Dice score, Global Accuracy, mIoU and MCA are used to measure the performance of the NIS optimised Multi-Loss CNN models. Evaluation results on the CamVid, Freiburg Forest and MESSIDOR data sets for the devised DeeplabV3 and FCN models are shown in Tables 17 and 18, respectively. We also analyze the selected hyper parameters in Section 4.4.3.

Tables 17 and 18 depict that NIS yields a superior performance over the standard PSO and FA methods across all three test data sets for both networks. The search strategies of NIS contribute toward improved hyper parameter selection, thus increase model prediction accuracy across all

Table 17: The mean results of four common metrics over 5 runs for the NIS-optimised DeeplabV3 model on three data sets

Data set	Search	Dice (%)	GA (%)	mIoU (%)	MCA (%)
CamVid	Default	61.0	84.9	51.8	63.6
	FA	72.8	85.1	59.7	70.2
	PSO	78.5	90.0	66.7	74.8
	NIS	**79.8**	**90.5**	**68.3**	**76.3**
Freiburg	Default	85.3	93.4	73.0	83.2
	FA	88.2	93.8	80.4	86.5
	PSO	88.3	**94.2**	80.7	86.7
	NIS	**89.0**	94.1	**81.4**	**87.7**
MESSIDOR	Default	81.1	99.3	74.4	72.0
	FA	92.5	99.7	87.6	90.5
	PSO	83.6	99.5	76.6	77.7
	NIS	**95.6**	**99.8**	**92.1**	**93.9**

four metrics. Furthermore, the models trained with optimised hyper parameters perform better than those trained with typical default configurations. Specifically, models trained with default settings use a single Cross Entropy loss function with default learning settings, while the devised networks in this work are equipped with an optimised multi-loss function in combination with more effective learning settings. As such, the latter shows great efficiency in learning from the backpropagation process to adjust the performance with customized learning behaviours. These observations are empirically shown across all three data sets, indicating that using NIS hyper parameter optimisation together with a multi-loss function provides benefits across multiple problems and model structures.

The improved performance gained from using a multi-loss function results from the combination of diverse and unique loss calculation mechanisms. Each constituent loss function measures the error between the prediction and the GT differently making the error signal inherently more informative. This leads to better error correction during training as compared with using a single loss function, yielding improvements in the predictive capability of the resulting model. In addition, optimised parameters γ and α enable the multi-loss function to well balance the impact of well/poorly classified instances and contributions of the classes present/non-present in the GT masks, in order to avoid overfitting.

Table 18: The mean results of four common metrics over 5 runs for the NIS-optimised FCN model on three data sets

Data set	Search	Dice (%)	GA (%)	mIoU (%)	MCA (%)
CamVid	Default	51.4	84.3	42.6	54.6
	FA	70.6	83.6	57.4	67.3
	PSO	74.9	87.6	62.5	71.3
	NIS	**79.5**	**90.5**	**68.1**	**76.4**
Freiburg	Default	62.8	90.1	55.7	62.2
	FA	85.9	92.5	77.1	84.5
	PSO	77.7	87.8	68.4	75.9
	NIS	**86.7**	**92.7**	**78.0**	**85.5**
MESSIDOR	Default	81.0	99.2	76.6	80.8
	FA	90.6	99.7	86.5	90.3
	PSO	85.8	99.6	80.2	82.4
	NIS	**94.6**	**99.8**	**90.3**	**94.8**

Hyperparameter Selection

An overview of the hyper parameter selection results of each optimisation method across all three data sets are provided in Tables 19 and 20 for the DeeplabV3 and FCN models, respectively. Each table displays the optimised learning rate (Lr), momentum, α and γ configurations.

Cross referencing the results from Tables 17, 18 and 19, 20, we observe that lower settings of learning rate, γ, and α, in combination with higher momentum parameters, produce improved accuracy for every metric across all data sets and models. NIS obtains such preferred hyper parameter configurations effectively, in contrast to FA and PSO which are comparatively more sensitive to swarm initialization.

High momentum and low learning rate constitute a typical hyper parameter setting for most CNNs. It is therefore unsurprising to see this trend reflected in the results. While most of the algorithms identify a high mean momentum, NIS additionally identifies the importance of a low average learning rate as well, contributing to the high accuracy metrics in the generated networks. This lower average learning rate allows a network to make finer adjustments to the weights during training. It provides greater variation and granularity in the potential internal network representations of the data set, allowing for higher accuracy. High momentum places emphasis on the overall trajectory towards a potential global optimum. This pseudo inertia effect reduces the likelihood of weights being trapped in local optima or go beyond global optimum solutions.

Configurations of low γ combined with low α loss coefficients adapt the focal loss function in a way that a CNN can focus more on classes not present in GT during training (i.e. where the class GTs are zero). Specifically, weights are adjusted to produce less incorrect predictions as opposed explicitly adjusting weights with regard to further enhance correct predictions. In a multi-class situation (e.g. for the CamVid and Freiburg data sets), this can lead to a more stable approach to training as the network can focus on filtering irrelevant signals/noise from images and compensate for class imbalance in the data set. With respect to the MESSIDOR data set, we notice that in some test cases, the highest metric measures have the lowest learning rates combined with slightly higher α and γ values. Since a foreground-background binary segmentation process is performed in MESSIDOR, the balancing effect from low α and γ values is less influential due to the lower number of classes. The lower learning rates can therefore take precedence, focusing on finer weight adjustments, along with higher momentums with sufficient capabilities in jumping out of local optima.

Table 19: Average hyper parameters identified over 5 runs using search methods based on the DeeplabV3 model on three data sets

Data set	Search	Lr	Momentum	α	γ
CamVid	FA	0.0093	0.8297	0.4625	3.9188
	PSO	0.0093	0.8274	0.6043	3.4761
	NIS	0.0059	0.8707	0.3786	1.9099
Freiburg	FA	0.0072	0.8008	0.4230	1.5410
	PSO	0.0067	0.8636	0.4866	2.8988
	NIS	0.0054	0.7514	0.3506	2.0119
MESSIDOR	FA	0.0079	0.8732	0.6860	2.1909
	PSO	0.0083	0.9053	0.7007	3.1876
	NIS	0.0036	0.8345	0.4448	2.4869

To further analyze the optimisation algorithms, a detailed examination of the identified hyper parameters and the results from devised DeeplabV3 networks, is conducted. Tables 17 and 19 show the segmentation results and identified hyper parameters using DeeplabV3, respectively. Results for the CamVid data set indicate that both PSO and FA select high learning rate, momentum and γ values with moderate α values with Dice scores of 78.5% and 72.8% respectively. Both models perform worse than NIS, which selects high momentums with low α, γ and learning rate configurations with a Dice score of 79.8%. The combination of moderate α with high γ coefficients increases the error signals from both the most correctly and incorrectly classified examples whilst reducing the error signals for the moderately classified examples. Such loss function settings

impose harsh punishment/reward for the extremes, causing class-level accuracy rates to stagnate near 50%. This pushes the network to learn a representation of the data set that has less variance in accuracy between classes present and non-present in the GT masks. It leads to a less informative error signal, producing instability in the network during training and encouraging under-fitting. By combining high learning rates and momentums with the above loss function settings, the under-fitting issue is simply compounded. The reason the models with such settings do not fail completely is likely owing to the other two loss functions compensating for poor configurations identified by PSO and FA.

The results for the DeeplabV3 model trained on the Freiburg forest data set reveal again that NIS selects high momentums with low α, γ and learning rate settings with a Dice score of 89%. Similarly, PSO again selects high learning rate, momentum and γ values with moderate α coefficients with a Dice score of 88.3%. Meanwhile, FA selects high learning rates, high momentums, low γ and moderate α values with a Dice score of 88.2%. FA stands out as it selects low/moderate α values and very low γ parameters, which should lead to good results. However, because of the extracted high learning rates and momentums, the performance boost is minimal over PSO. While the variance in Dice scores across models is minimal for this data set, we can still observe that high momentums with low α, γ and learning rate settings produce better results.

Table 20: Average hyper parameters identified over 5 runs using search methods based on the FCN model on three data sets

Data set	Search	Lr	Momentum	α	γ
CamVid	FA	0.0077	0.5421	0.5848	3.9476
	PSO	0.0084	0.6732	0.4753	3.2803
	NIS	0.0065	0.8214	0.2815	2.6709
Freiburg	FA	0.0052	0.7462	0.6782	2.7889
	PSO	0.0052	0.4689	0.7493	3.7681
	NIS	0.0030	0.8105	0.3356	1.5221
MESSIDOR	FA	0.0073	0.6850	0.2335	2.5713
	PSO	0.0076	0.6550	0.4807	3.6714
	NIS	0.0024	0.8584	0.2420	2.3641

The results for the MESSIDOR data set for the DeeplabV3 model further reinforce the superiority of high momentum, low α, low γ and low learning rate hyper parameter configurations, which are again preferred by NIS producing a Dice score of 95.6%. Yet again PSO selects high learning rate, momentum, γ and α values, yielding a Dice score of 83.6%. FA selects high learning rate,

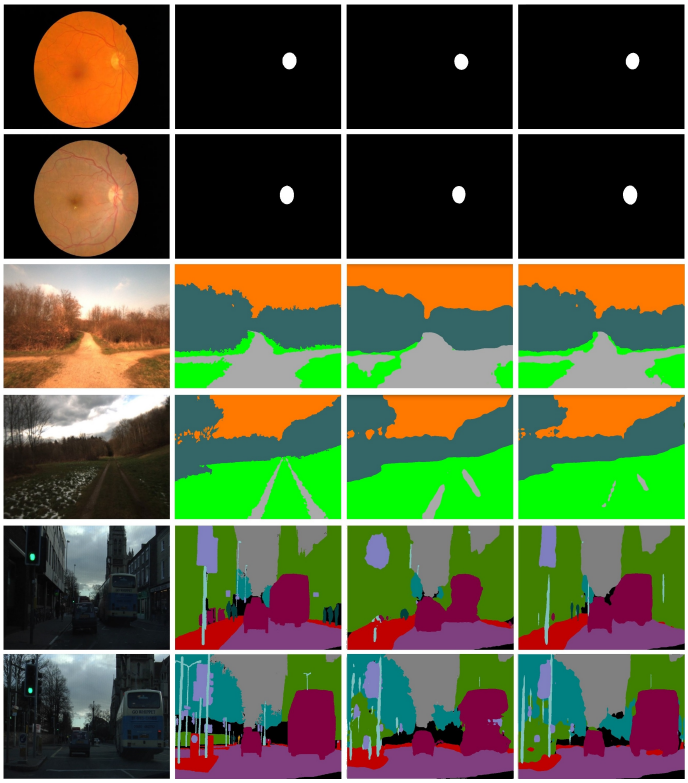


Figure 18: Example segmentation results (from left to right, input images, GT masks, outputs of NIS-optimised FCN and NIS-optimised DeeplabV3)

momentum, and α configurations and low γ coefficients, giving a Dice Score of 92.5%. Although FA selects high learning rate and α settings, they are somewhat lower than those identified by PSO. These FA-optimised configurations together with significantly reduced γ values improve the performance over those of PSO but have worse performance in comparison with those of NIS.

A similar observation is also obtained for the hyper parameter selection using the FCN model. Tables 18 and 20 illustrate the segmentation performances of optimised FCN models and hyper parameters identified by the three search methods, i.e. PSO, FA and NIS. The empirical results from the three data sets reinforce the correlation of high performances with the high momentum, and low α, γ and learning rate hyper parameter configurations which are preferred by NIS. On the contrary, the settings, such as high α, γ and learning rate with low momentum hyper parameters, obtained by PSO and FA in most test cases, are more likely to result in local optima traps, therefore leading to less competitive results.

In summary, NIS selects low α, γ, learning rate parameters, and high momentum settings, produc-

ing superior performance in comparison with those of FA and PSO, when evaluated on DeeplabV3 and FCN networks across three benchmark segmentation data sets. In light of this, the use of a multi-loss function and the scheduling of novel exploitation and exploration behaviours through the Discrete Adaptive Wave Function introduces delicate and thorough loss measurements and diversified search behaviours to enhance segmentation efficiency.

With respect to optimisation strategies, we notice that FA struggles with exploration, due to the focus of agents moving toward the area of the highest light intensity, requiring careful initialisation of FA positions to provide suitable exploration. On the other hand, the exploration behaviour of PSO is also restricted owing to a lack of adaptive balance of cognitive and social scheduling. This can cause large areas of the solution space being unexplored, placing more reliance on favourable initialisation of particle positions. NIS solves this by introducing iteration-based scheduling of novel exploitation and exploration behaviours via the proposed Discrete Adaptive Wave Function. This function leverages heuristic coefficients to maximize or minimize different search behaviours in three separate stages. In the early search stage, the first proposed Maximised Standard Deviation Velocity Prediction mechanism extends the search territory with an attempt to thoroughly spread the particles in the search space. The second search action, local best velocity prediction, leads the particles towards the independent personal best positions in the subsequent search iterations. The third proposed Whirlpool search strategy implements an n-dimensional spiral-like search action to fine-tune search exploitation of promising areas and to bypass local minima traps. The experimental results indicate the effectiveness of the proposed search strategies in avoiding early stagnation. Figure 18 illustrates the example outputs of NIS-optimised FCN and DeeplabV3 networks.

In Table 21, we present a comparison of the NIS-optimised DeeplabV3 network against state-of-the-art models on the three data sets, owing to the efficiency of the devised DeeplabV3 network. Most of the selected existing studies in Table 21 employed the official train-test splits for the above data sets as in this research. The empirical results indicate that the optimised networks in this work yield improved performance in comparison with those of other deep networks across multiple metrics.

Table 21: Comparison with other Reported Results

Method	Dice (%)	GA (%)	mIoU (%)	MCA (%)
MESSIDOR				
Abdulla et al. [87]	93.39	**99.89**	87.9	-
Morales et al. [88]	89.50	99.49	82.28	-
Kumar et al. [110]	84.56	-	-	-
Rehman et al. [111]	85.1	98.8	74.7	-
Zahoor and Fraz [112]	90.3	99.1	84.4	-
Fan et al. [113]	91.96	97.7	86.3	-
NIS-Deeplabv3	**95.6**	99.8	**92.1**	**93.9**
Freiburg Forest				
FC-DenseNet67[83]	-	-	74.47	-
FCN8[83]	-	-	80.05	-
ParseNet[83]	-	-	80.89	-
FastNet[83]	-	-	81.00	-
CGBNet[83]	-	-	81.04	-
SegNet[83]	-	-	81.12	-
Zhang et al.[82]	-	92.07	79.87	-
NIS-Deeplabv3	**89.0**	**94.1**	**81.4**	**87.**7
CamVid				
Segnet [86]	-	90.40	60.10	71.20
Dilation + FSO [114]	-	-	66.12	-
LRN [115]	-	-	61.7	**77.2**
G-FRNet [84]	-	-	68.0	-
FC-DenseNet103 [85]	-	**91.5**	66.9	-
DPDB-Net [116]	-	85.2	54.7	-
NIS-Deeplabv3	**79.8**	90.5	**68.3**	76.3

4.4.4 NIS Evaluation using Benchmark Test Functions

To validate the contributions of the Discrete Adaptive Wave Function scheduling and exploration/exploitation behaviours of NIS, we evaluate it against a total of 12 search methods in solving mathematical test functions, including PSO, FA [117], Random Search (RA) [118], Memetic Algorithm (MA) [119], Jaya [120], Gravitational Search Algorithm (GSA) [121], Genetic Algorithm (GA) [122], Flower Pollination Algorithm (FPA) [123], Cuckoo Search (CS) [124], Arithmetic Optimisation Algorithm (AOA) [125], Adaptive Random Search (ARS) [126], and Autonomous Particle Groups PSO (AGPSO) [127].

A total of nine benchmark functions are utilized, i.e. Ackley, Dixon Price, Powell, Rastrigin,

Table 22: Evaluation results for the benchmark functions with dimension=30

		RA	MA	Jaya	GSA	GA	FPA	CS	AOA	ARS	PSO	FA	AGPSO	NIS
Ackley	mean	1.82E+01	1.73E+01	1.91E+01	1.61E+01	1.91E+01	1.59E+01	1.88E+01	1.96E+01	1.78E+01	1.81E+01	1.96E+01	1.54E+01	**5.39E+00**
	min	1.77E+01	1.58E+01	1.78E+01	1.49E+01	1.85E+01	1.37E+01	1.75E+01	1.83E+01	1.66E+01	1.76E+01	1.86E+01	1.42E+01	**2.81E+00**
	max	1.85E+01	1.83E+01	1.97E+01	**1.67E+01**	1.97E+01	1.74E+01	1.92E+01	2.01E+01	1.82E+01	1.84E+01	1.99E+01	1.69E+01	1.97E+01
	std	2.09E-01	5.77E-01	3.71E-01	4.32E-01	3.27E-01	9.35E-01	3.88E-01	4.03E-01	3.69E-01	**1.75E-01**	2.60E-01	5.05E-01	4.79E+00
DixonPrice	mean	6.26E+05	3.13E+04	1.42E+06	1.69E+06	9.34E+04	6.28E+04	8.50E+05	1.62E+06	5.56E+05	4.13E+03	1.64E+06	5.26E+02	**2.37E+01**
	min	3.93E+05	1.84E+04	6.87E+05	5.11E+05	1.07E+03	3.51E+04	3.72E+05	9.30E+05	3.64E+05	2.35E+03	7.65E+05	2.80E+02	**4.41E+00**
	max	8.93E+05	4.40E+04	2.23E+06	2.28E+06	7.11E+05	1.11E+05	1.20E+06	2.32E+06	7.32E+05	5.67E+03	2.22E+06	1.05E+03	**7.23E+01**
	std	1.25E+05	6.10E+03	3.46E+05	3.82E+05	1.62E+05	2.17E+04	1.98E+05	3.28E+05	9.89E+04	7.65E+02	3.34E+05	1.54E+02	**1.70E+01**
Powell	mean	7.78E+08	8.78E+08	1.44E+12	1.60E+13	1.60E-02	1.41E+06	3.04E+10	3.18E+13	5.36E+08	1.24E+03	2.97E+12	1.99E+01	**2.78E-05**
	min	2.06E+07	2.20E+07	6.15E+08	2.43E+09	2.33E-03	1.88E+03	1.42E+08	1.22E+10	7.25E+06	9.72E+01	6.03E+09	9.10E+00	**1.01E-09**
	max	3.62E+09	4.17E+09	1.07E+13	1.22E+14	5.18E-02	1.78E+07	1.28E+11	2.82E+14	2.43E+09	5.85E+03	1.84E+13	4.24E+01	**3.73E-04**
	std	8.07E+08	8.90E+08	2.02E+12	2.92E+13	1.06E-02	3.19E+06	3.39E+10	6.63E+13	5.71E+08	1.13E+03	4.44E+12	7.95E+00	**8.84E-05**
Rastrigin	mean	3.37E+02	3.40E+02	4.01E+02	2.10E+02	3.80E+02	2.93E+02	3.61E+02	4.29E+02	3.31E+02	1.73E+02	4.22E+02	1.11E+02	**7.16E+01**
	min	3.01E+02	2.91E+02	3.33E+02	1.65E+02	3.20E+02	2.59E+02	3.19E+02	3.85E+02	3.00E+02	1.38E+02	3.75E+02	**1.59E+01**	3.55E+01
	max	3.70E+02	3.63E+02	4.50E+02	2.40E+02	4.12E+02	3.19E+02	3.98E+02	4.62E+02	3.47E+02	1.94E+02	4.60E+02	1.79E+02	**1.18E+02**
	std	1.54E+01	1.73E+01	2.48E+01	1.73E+01	1.87E+01	1.51E+01	1.97E+01	2.17E+01	1.27E+01	**1.39E+01**	1.88E+01	7.00E+01	2.22E+01
Rosenbrock	mean	4.29E+05	8.25E+04	1.02E+06	1.43E+06	5.29E+04	5.50E+04	7.70E+05	1.32E+06	3.85E+05	1.66E+05	1.45E+06	9.22E+04	**2.29E+02**
	min	2.19E+05	5.15E+04	2.97E+05	4.34E+05	4.19E+03	2.67E+04	2.21E+05	5.20E+05	2.10E+05	1.36E+05	8.78E+05	2.65E+04	**8.78E+01**
	max	6.10E+05	1.11E+05	1.76E+06	2.08E+06	1.86E+05	9.02E+04	1.08E+06	1.93E+06	4.88E+05	1.97E+05	2.06E+06	1.21E+05	**5.74E+02**
	std	1.09E+05	1.26E+04	3.35E+05	3.60E+05	4.88E+04	1.77E+04	2.14E+05	3.50E+05	6.18E+04	1.25E+04	3.24E+05	2.04E+04	**1.15E+02**
Rotated Hyper Ellipsoid	mean	2.23E+05	6.39E+04	3.47E+05	3.98E+05	2.66E+05	5.91E+04	2.54E+05	3.97E+05	2.12E+05	2.04E+04	3.89E+05	7.85E+03	**6.06E+02**
	min	1.79E+05	4.39E+03	2.49E+05	2.76E+05	2.04E+05	3.64E+04	1.58E+05	2.97E+05	1.51E+05	1.44E+04	3.05E+05	4.11E+03	**1.55E+02**
	max	2.70E+05	1.09E+05	4.38E+05	4.67E+05	3.13E+05	9.17E+04	3.18E+05	5.03E+05	2.49E+05	2.48E+04	4.84E+05	1.49E+04	**1.98E+03**
	std	2.51E+04	2.43E+04	4.61E+04	4.19E+04	2.85E+04	1.46E+04	3.22E+04	4.67E+04	1.92E+04	2.86E+03	3.55E+04	2.65E+03	**4.44E+02**
Sphere	mean	1.03E+02	1.05E+02	1.51E+02	7.69E+00	1.40E+02	2.60E+01	1.19E+02	1.73E+02	1.01E+02	9.20E+00	1.71E+02	5.65E+00	**2.09E-01**
	min	7.41E+01	8.19E+01	1.20E+02	4.16E+00	1.09E+02	1.65E+01	7.39E+01	1.34E+02	7.22E+01	7.07E+00	1.43E+02	4.45E+00	**6.09E-02**
	max	1.22E+02	1.21E+02	1.83E+02	1.23E+01	1.59E+02	3.89E+01	1.44E+02	2.04E+02	1.17E+02	1.11E+01	2.01E+02	6.79E+00	**4.80E-01**
	std	1.09E+01	1.06E+01	1.53E+01	2.20E+00	1.25E+01	5.65E+00	1.59E+01	1.65E+01	8.68E+00	9.81E-01	1.52E+01	5.36E-01	**1.11E-01**
SumSquares	mean	1.43E+03	1.30E+03	2.23E+03	7.88E+01	1.56E+03	3.59E+02	1.59E+03	2.43E+03	1.35E+03	1.34E+02	2.40E+03	7.60E+01	**1.78E+00**
	min	1.17E+03	1.08E+03	1.75E+03	3.14E+01	9.39E+02	2.25E+02	1.18E+03	1.49E+03	7.81E+02	1.04E+02	1.84E+03	5.39E+01	**4.35E-01**
	max	1.70E+03	1.50E+03	2.76E+03	1.35E+02	2.10E+03	5.51E+02	1.97E+03	3.01E+03	1.56E+03	1.70E+02	2.90E+03	9.20E+01	**4.09E+00**
	std	1.34E+02	1.17E+02	2.37E+02	2.04E+01	2.30E+02	7.15E+01	2.17E+02	3.12E+02	1.54E+02	1.76E+01	2.80E+02	9.57E+00	**9.80E-01**
Zakharov	mean	4.12E+02	2.44E+02	7.18E+02	8.35E+08	3.59E+02	4.24E+02	5.39E+02	4.91E+08	3.71E+02	2.30E+08	5.11E+07	2.20E+02	**6.71E+01**
	min	2.61E+02	1.74E+02	4.08E+02	1.48E+03	2.13E+02	3.25E+02	2.13E+02	6.17E+02	2.63E+02	5.92E+02	6.14E+02	2.91E+01	**2.17E+01**
	max	5.38E+02	3.47E+02	1.02E+03	1.01E+10	6.72E+02	5.28E+02	7.40E+02	4.26E+09	4.52E+02	3.97E+09	1.00E+09	6.95E+02	**1.49E+02**
	std	5.93E+01	4.57E+01	1.45E+02	1.85E+09	8.86E+01	6.26E+01	1.25E+02	8.72E+08	4.91E+01	7.47E+08	1.93E+08	1.68E+02	**3.50E+01**

Table 23: The Wilcoxon rank sum test results for the benchmark functions over 30 runs

	Ackley	DixonPrice	Powell	Rastrigin	Rosenbrock	RotHyp	Sphere	SumSquares	Zakharov
RA	1.07E-07	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
MA	1.07E-07	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
Jaya	9.06E-08	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
GSA	1.07E-07	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
GA	8.35E-08	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
FPA	1.07E-07	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
CS	1.07E-07	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
AOA	1.55E-09	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
ARS	1.07E-07	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
PSO	1.07E-07	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
FA	4.18E-09	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
AGPSO	1.07E-07	3.02E-11	3.02E-11	**7.73E-02**	3.02E-11	3.02E-11	3.02E-11	3.02E-11	7.22E-06

Rosenbrock, Rotated Hyper-Ellipsoid, Sphere, Sum Squares, and Zakharov. Each benchmark function provides a unique challenge for optimisation algorithms by defining various shapes intended to trap the algorithm in local optima. The details of these numerical optimisation functions are provided in Tan et al. [39]. We adopt the following experimental configurations for evaluating these test functions, i.e. 50 particles, 500 iterations and 30 dimensions. Each algorithm is executed for 30 trials. The minimum, maximum, standard deviation and mean average results are summarized in Table 22.

The results from Table 22 indicate that NIS performs better than all other methods for all the benchmark functions. This further indicates the strength of the three behaviours and the Discrete

Adaptive Wave scheduling scheme. In particular, the effectiveness is highlighted by the performance of NIS as compared with those of PSO and AGPSO. To ensure statistical validity of these results, the Wilcoxon rank sum test is performed, and the results are presented in Table 23. The statistical test results confirm that the p-value is smaller than 0.05 for nearly all test functions. The only exception is the Rastrigin function, where NIS and AGPSO achieve statistically similar results.

4.5 Conclusion

In this study, we introduced NIS, a novel optimization algorithm with it's constituent Discrete Adaptive Wave Function and Velocity Prediction Model. We evaluated the performance of this approach in the contexts of semantic segmentation and mathematical test functions.

A model selection process for semantic segmentation tasks was conducted, evaluating various CNN architectures, including FCN, U-Net, SegNet, and DeepLabv3, among others. These architecture selections were made due to their established track record of success in image segmentation tasks. Notably, among these, DeepLabv3 and FCN consistently demonstrated superior performance across multiple metrics such as Dice, GA, mIoU, and MCA.

The semantic segmentation tasks, utilizing datasets such as CamVid, Freiburg Forest, and MESSIDOR, showed promising results. Several loss functions were employed, including Dice loss, Cross-Entropy loss, and Focal loss. Through careful optimization of the Learning Rate, Momentum, α and γ hyper parameters, NIS consistently outperformed traditional optimization methods like PSO and FA on these segmentation task. This suggests that NIS has the potential to enhance the performance of segmentation models.

NIS demonstrated its robustness and effectiveness in optimizing challenging functions. While it generally outperformed other methods, we acknowledge the need for further investigations and comparisons to provide a comprehensive assessment.

The success of NIS optimisation can be attributed to both the Discrete Adaptive Wave Function scheduling scheme, which intelligently balances exploration and exploitation throughout the optimization process combined with the learned Local Best and Maximum Standard Deviation Velocity updates, as determined by the Velocity Prediction Model.

In conclusion, this research suggests that NIS combined with multi-loss functions, holds promise for optimizing deep learning models for semantic segmentation. While the results are promising, further research and refinement are needed to unlock its full potential and explore its applicability in broader domains.

4.6 Limitations of the Work

The proposed algorithm, NIS (Neural Inference Search), and the associated multi-loss function are innovative and offer several advantages. However, like any research work, they also have certain limitations:

1. Sensitivity to Initializations: The effectiveness of NIS can depend on the initial positions of the particles in the swarm. In some cases, poor initializations may lead to convergence to suboptimal solutions. Strategies for improving the robustness of the algorithm to different initializations may be needed.

2. Limited Generalization: The NIS algorithm relies on the training of a neural network (VPM) to predict velocity vectors. The generalization of this network to different problem domains or data distributions may be limited. Fine-tuning or retraining of the network may be necessary for new problem instances.

3. Manual Configuration: The Discrete Adaptive Wave function introduces stage-dependent coefficients to control the contributions of different search behaviors. Configuring these coefficients for specific problems may require manual intervention and domain knowledge. An automated mechanism for adjusting these coefficients could enhance usability.

4. Scalability: The scalability of NIS to extremely high-dimensional optimization problems is not well-demonstrated. Handling problems with hundreds or thousands of dimensions may pose challenges for the algorithm.

5. Dependency on Data: The VPM in NIS relies on sequentially accrued training data to predict velocity vectors which can be variable and may not always produce suitable data samples to generalise from. This indicates high levels of randomness in the initial stages which in some ways are beneficial for exploration, but is not particularly reliable.

6. Limited Interpretability: Neural networks used in NIS are often considered as ”black-box” models, making it challenging to interpret the decision-making process of the algorithm. Understanding why certain solutions are chosen or how the network generalizes can be difficult.

In conclusion, while NIS and the associated multi-loss function offer a novel approach to optimization and image segmentation tasks, they come with certain limitations that should be considered when applying them to real-world problems. Addressing these limitations and conducting further research can help refine and extend the applicability of these techniques.

Chapter 5

The Proposed Cluster Search Optimisation of Deep Learning Models for Audio Emotion Classification

5.1 Introduction

CSO is proposed to tackle optimisation of deep learning models for audio based emotion classification. CSO is used to perform an architecture and hyper parameter search upon three base model architectures: 1DCNN, BiLSTM, and CNN-BiLSTM. The learning rate, number of blocks and number of filters are optimised for the CNN-based models, i.e. 1DCNN and CNN-BiLSTM, whereas the optimisation targets for the BiLSTM model are the learning rate, number of BiLSTM layers, and number of hidden units. The details of the proposed CSO algorithm and optimisable deep learning models are described in the following subsections.

5.2 The Proposed Cluster Search Optimised CNN models for Emotion Detection

Inspired by PSO, we propose the CSO algorithm. It replaces the typical PSO velocity vector calculation with a conditional statement based on the novel Global Search Particle Selection Threshold G_{thresh} for the selection between exploitation and exploration based search strategies. The ex-

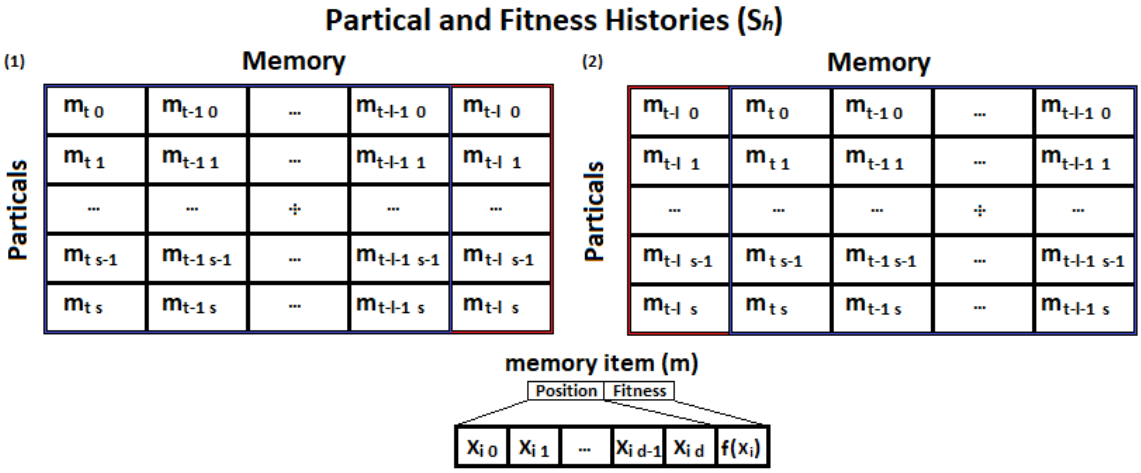


Figure 19: $\mathbf{S}_h$ is defined with a memory length l, swarm size s and at iteration t. Each memory item m is construed of a particle position x and fitness value $f(x)$ for some particle i with search space dimentionality d. (1) shows the initial state of $\mathbf{S}_h$ before rolling and (2) indicates the state after rolling. The roll essentially pushes all elements back by one index along the memory axis with the last index rolling around to the front. This allows for the last memory position to be overridden by the newest position fitness histories.

ploitation search employs a typical global best position approach whereas the exploratory search guides particles toward positions extracted via clustering the fitness and position history of all previous particles. These changes to PSO are highlighted in the corresponding velocity calculation for CSO shown in Equation 5.1.

$$\vec{v}_i^{t+1} = \begin{cases} 0.7\vec{v}_i^t + (\mathbf{C}_{x_j} - x_i^t), & \text{if } i < G_{\text{thresh}} \\ 0.7\vec{v}_i^t + (\vec{g}_{\text{best}}^t - x_i^t), & \text{otherwise} \end{cases} \tag{5.1}$$

where $\vec{v}_i^t$ is the current velocity vector for particle i at time t and thus $\vec{v}_i^{t+1}$ is the resultant velocity for the same particle at the next iteration. In similar fashion to PSO, x_i^t is the current particle position and g_{best_i} is the current global best position. This search action introduces a matrix of cluster centroids $\mathbf{C}_x$ obtained by employing OPTICS clustering [128] on all previous particle positions and fitnesses $\mathbf{S}_h$ (see Figure 19). Since OPTICS clustering is density based, it can dynamically determine the initial number of clusters, is robust to varying densities and is resilient to noise/outliers. As such it is a strong candidate for discovering structures within a dynamically changing and highly variable sample set like that which is present in the resulting solution space

Algorithm 5 The CSO algorithm

Initialise swarm position and fitness history $\mathbf{S}_h$
Initialise OPTICS clustering algorithm c_{optics}
Initialise best/worst cluster distances c_b, c_w
while $t < T$ **do**
  Roll $\mathbf{S}_h$ values (See Figure 19)
  Get cluster centers matrix $\mathbf{C}_x$ and cluster fitness vector $\vec{c_f}$ by clustering swarm history $c_{\text{optics}}(\mathbf{S}_h)$
  Sort $\mathbf{C}_x$ and $\vec{c_f}$ according to decreasing fitness
  Calculate cluster distance c_d using Equation 5.3
  if $c_b > c_d$ **then**
    $c_b = c_d$
  end if
  if $c_w < c_d$ **then**
    $c_w = c_d$
  end if
  Calculate Global Search Particle Selection Threshold G_{thresh} using Equation 5.5
  for each particle $i = 1, ..., S$ **do**
    Calculate velocity $\vec{v}_i^{t+1}$ as in Equation 5.1
    Calculate distance $\vec{x}_i^{t+1}$ as in Equation 5.2
    if $f(\vec{x}_i^{t+1}) < f(\vec{g}_{\text{best}}^t)$ **then**
      $\vec{g}_{\text{best}}^t = \vec{x}_i^{t+1}$
    end if
    Store fitness and position in $\mathbf{S}_h$
  end for
end while
return $\vec{g}_{\text{best}}^t$

from optimising the hyper-parameters and structure of machine learning models. This allows better identification of distinct areas of good or bad fitness within the solution space via the resultant cluster centers. More specifically these cluster centers are defined as $\mathbf{C}_x \in \mathbb{R}^{n \times d}$ where n is the number of clusters and d is the search space dimensionality. Therefore $\mathbf{C}_{x_j}$ is the j-th cluster centroid vector where j is a particle assignment index such that $j = i \mod d$. This indexing scheme ensures particles performing an exploration search will target the fittest cluster centroids first. It also enables multiple particles to be assigned to the same clusters when the swarm size is larger than the number of clusters. The 0.7 factor was established through trial and error based testing. The resultant velocity vector is then applied to the standard PSO position update shown in Equation 5.2.

$$\vec{x}_i^{t+1} = \vec{x}_i^t + r_i \vec{v}_i^{t+1} \tag{5.2}$$

In this Equation $\vec{x}_i^t$ is the current particle position, $\vec{x}_i^{t+1}$ is the next particle position and r_i is a

uniformly distributed random value sampled from $U(0, 1)$. The full operation of this algorithm is described in Algorithm 5. The specifics of the G_{thresh} calculation and particle assignment are described in the proceeding subsections.

5.2.1 Global Search Particle Selection Threshold

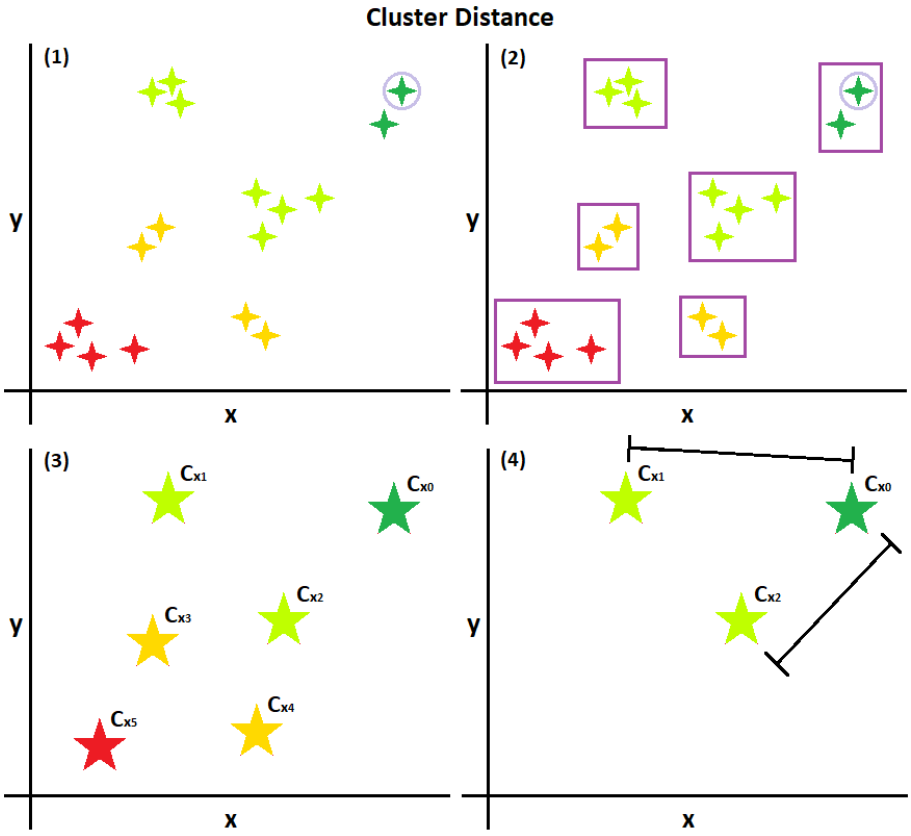


Figure 20: The cluster distance is calculated according to the sequential steps highlighted in the four images. (1) A collection of previously evaluated particle positions with fitnesses indicated by colour, where red indicates low fitness and green refers to high fitness. (2) OPTICS based clustering of the position and fitness to assign cluster labels for each sample. (3) Determination of centroid position and fitness of each cluster. (4) The worst 50% of the clusters are removed with the remaining clusters used to calculate the final cluster distance as in Equation 5.3.

G_{thresh} enables dynamic selection of particles so they either follow the global best solution or a cluster center in $\mathbf{C}_x$ and thus select between an exploratory search and an exploitative action. The conditional application of G_{thresh} causes more particles to search near the current global best solution as the fittest clusters converge over time, indicating a good area fitness generally. This is achieved through monitoring the summed distance between the fittest cluster with the top 50% fittest clusters as shown in Figure 20 using Equation 5.3. Removing the top 50% fittest clusters

prevents recently explored areas of poor fitness from impacting the overall consensus on whether the algorithm has converged. In such cases there could be a strong indication of where the most promising areas of fitness is but this would not be strongly reflected in the Cluster Distance Metric. This would cause particles to be assigned an exploration search when in fact an exploitation type search would be more appropriate. The overall effect of removing the worst cluster centroids therefore makes the particle search selection more robust to noise whilst also reacting quicker to changes in the consensus of the most promising clusters, accelerating convergence

$$c_d = \sum_{k=0}^{k=\frac{K}{2}} \sqrt{\mathbf{C}_{x_k}^2 - \mathbf{C}_{x_0}^2} \tag{5.3}$$

Note that k is the current cluster and K is the total number of clusters. Therefore $\mathbf{C}_{x_k}$ is the k-th cluster centroid, $\mathbf{C}_{x_0}$ is the fittest cluster centroid, and c_d is the current cluster distance. At every iteration c_d is calculated and used to update the best and worst cluster distance (c_b and c_w respectively). c_d, c_b and c_w are further combined in Equation 5.4 producing the cluster distance improvement C_{imp}.

$$C_{\text{imp}} = -\frac{c_d - c_b}{c_b - c_w} \tag{5.4}$$

With C_{imp} being value between 0 and 1 which indicates how much the cluster distance has shrunk relative to the largest cluster distance seen during algorithm execution (See Figure 21). The cluster improvement measure forms the basis for G_{thresh} as shown in Equation 5.5.

$$G_{\text{thresh}} = S \times C_{\text{imp}} - 1 \tag{5.5}$$

where S is the number of particles in the swarm. With G_{thresh} established, the following subsection discusses the conditional assignment of the velocity equation to different particles, the process of cluster centroid velocity target selection and the subsequent effects therein.

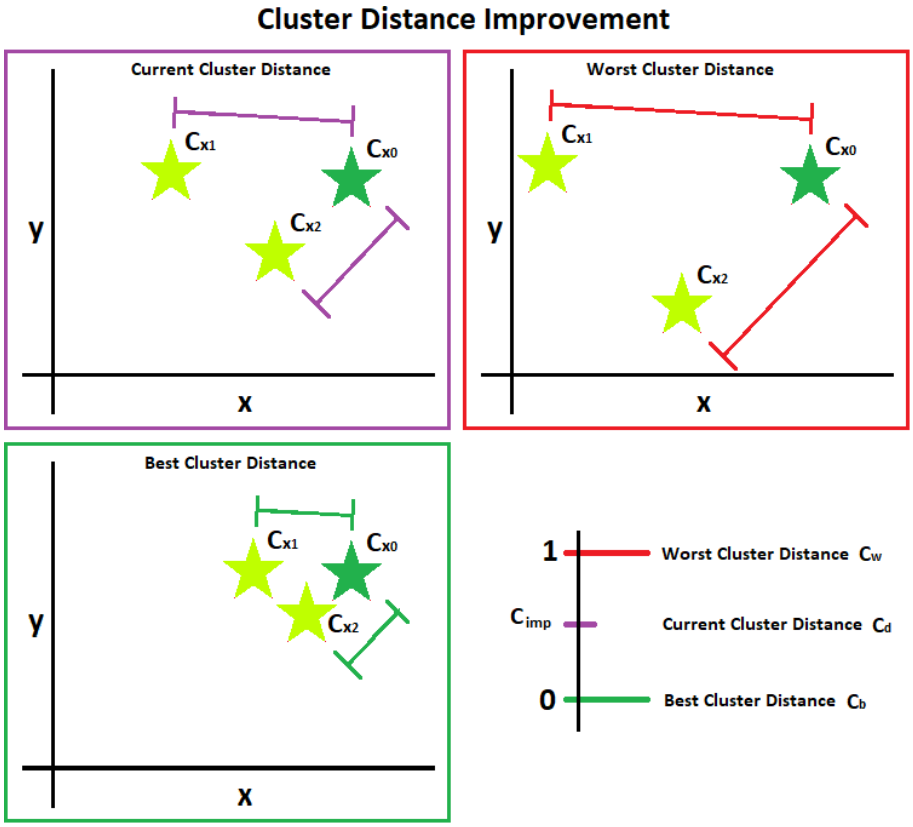


Figure 21: The cluster distance improvement C_{imp} is calculated using Equation 5.4. The current worst cluster distance c_w, the current best cluster distance c_b and the current cluster distance c_d are highlighted in the three graphs.

5.2.2 Particle Assignment

As mentioned previously, the type of velocity equation a particle will use is selected using the conditional statement in Equation 5.1. Equation 5.5 shows that the range of C_{imp} is expanded from $[0, 1]$ to $[-1, S-1]$ enabling G_{thresh} to conditionally assign a particle to either the $\vec{g}^t_{\text{best}}$ or a cluster-based search action based on the particle index. Note that the top of the expanded range $(S-1)$ ensures that one particle will always follow $\vec{g}^t_{\text{best}}$ leading to a constant exploitation of the most promising solution. As the algorithm progresses c_d will fluctuate causing the interval between the best and worst cluster distance values to widen and provide a strong measurement of convergence via C_{imp}. At points of low convergence (i.e. $C_{\text{imp}} \rightarrow 1$), most particles will search around the cluster centroids in decreasing order of fitness, leading to a varied set of exploration targets with a prioritisation on the most promising ones, excluding the true global best solution (see Figure 22). In the opposite case where cluster centers have converged (i.e. $C_{\text{imp}} \rightarrow 0$), the majority of particles are assigned the search guided by the global best solution.

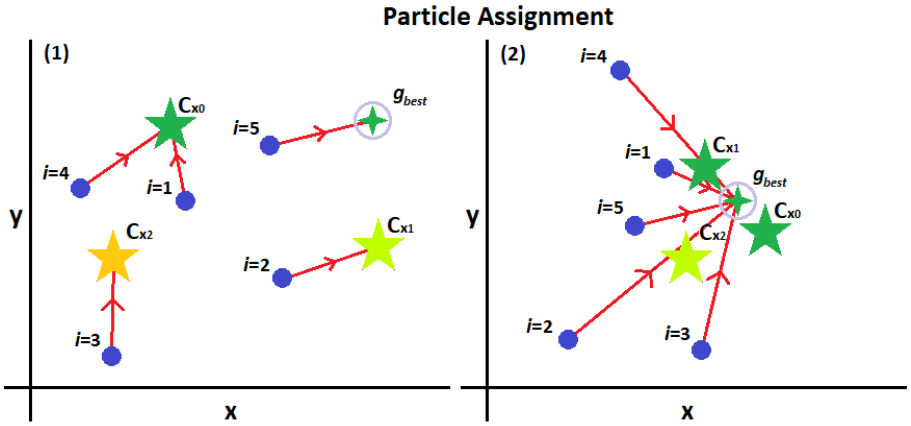


Figure 22: (1) shows the case where $C_{\text{imp}} \rightarrow 1$ such that one particle follows g_{best} and the rest are assigned to cluster centers in decreasing order of fitness. (2) shows the case where $C_{\text{imp}} \rightarrow 0$ and therefore all particles search toward the g_{best} position. In both cases there are three cluster centroids (5 pointed stars) and the global best position (4 pointed star). Fitness is indicated by $\mathbf{C}_x$ subscript numbers and coloured such that green is high and red is low. Velocity vectors for each particle i are extracted from the distance to the assigned target point (indicated by the red line).

5.3 Deep Learning Models

To conduct audio emotion classification, three optimisable block-based deep neural network models are introduced: 1DCNN, BiLSTM, and CNN-BiLSTM. Each of these architectures have an adjustable number of blocks n, with each block contain a standard selection of deep learning layers appropriate for classification tasks. Explicitly put, this means that when $n = 5$, 5 blocks will be stacked sequentially within a predetermined part of the deep learning architecture. Additionally each optimisable neural network has a parameter f which adjusts the number of hidden units or filters in the constituent deep learning layers. These three models are described in detail in the proceeding subsections.

5.3.1 1-Dimensional CNN

The 1DCNN uses melspectograms as a time sequence of frequency amplitudes. Such an input is fed into n convolutional blocks stacked in series as defined in Figure 23. Each block consists of two parallel convolutional layers, one of which has a dilation rate of 2 and the other a dilation rate of 1 (normal convolution) (see Figure 24), allowing for double the receptive field for the same memory and computational footprint. Preliminary investigation involved testing multiple rates of dilation summed in the same manner but seemed to yield no tangible benefit. The output of

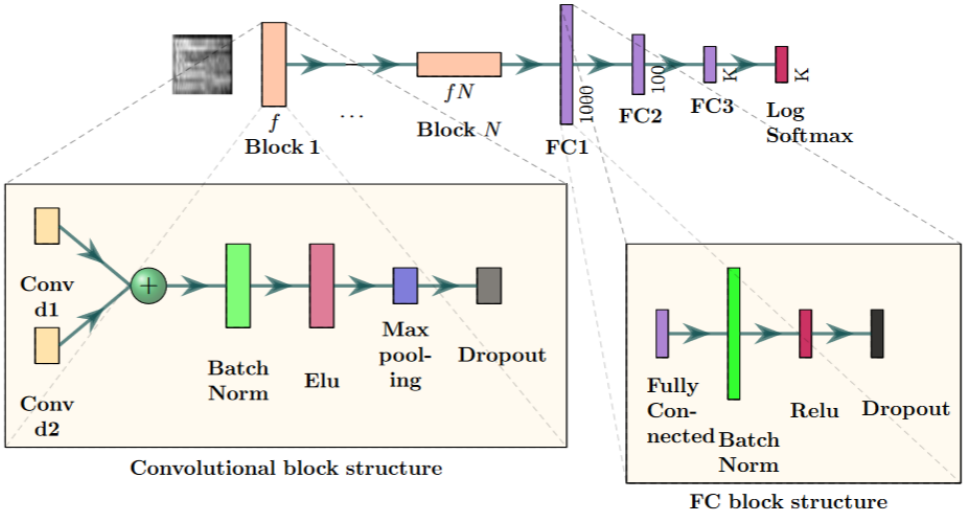
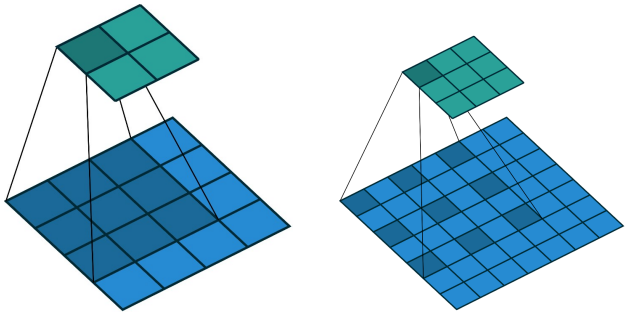


Figure 23: A visualisation of the 1DCNN architecture. d1 indicates no dilation i.e, a normal convolution. d2 indicates a dilation rate of 2 which spaces out the attention as according to Figure 24

these layers is summed then passed through a 1D batch normalisation layer, Elu activation layer, 1D Max pooling layer and Dropout layer. The output from the n-th block is then fed to three fully connected blocks, each of which having a fully connected, 1D Batch Normalisation, ReLU Activation, and Dropout layers. Each fully connected layer in the subsequent stacked blocks decreases unit size from 1000 to 100 and then K, where K is the number of emotion classes in the data sample's ground truth. The resultant output is passed through a log softmax function to produce the final predictions for the loss calculation. The optimisation targets for this network are the learning rate, the number of stacked blocks n, and the baseline number of filters within each block f. For each successive block, the number of filters is increased by a factor of $f \times n$.

5.3.2 BiLSTM

The optimisable BiLSTM model also uses melspectogram inputs with the frequency amplitude bins used as features and the time dimension defining the sequence. This BiLSTM model follows a similar optimisation and classification structure to the 1DCNN, however instead of optimising n to select the number of convolutional blocks, n selects the number of stacked bidirectionalLSTM layers. In this case f is also adopted from the 1DCNN model but instead decides the number of hidden units within the bidirectionalLSTM layers. The last bidirectionalLSTM layer ouput is passed to the same fully connected block structure as defined for the 1DCNN model. While both



(a) (b)

Figure 24: E

xamples of two convolution types [129]: (a) Normal Convolution (dilation rate 1) and, (b) Atrous convolution (dilation rate 2). As can be seen the receptive field is almost doubled when using a dilation rate of 2. Since the atrous and normal convolutions yield different filter sizes when applied directly, padding is applied to ensure the output and input sizes are the same, please note the filter sizes used in this illustration are for demonstration purposes only. The intention is to highlight the spacing between the receptive field of the two kernels

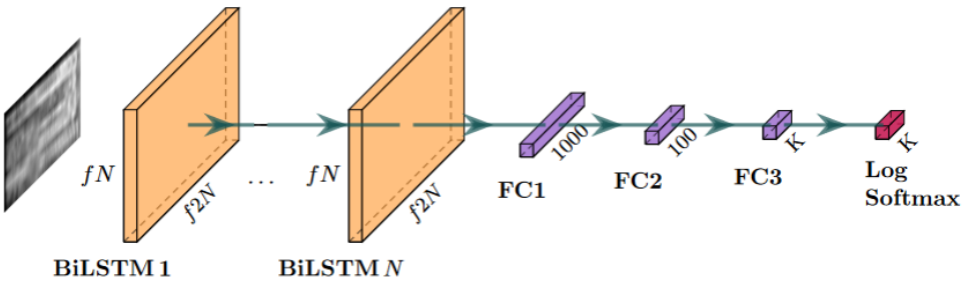


Figure 25: The optimisable BiLSTM model structure.

the 1DCNN and BiLSTM based models work directly with sequential features to extract temporal features, 1DCNNs typically are poor at capturing global features where as the BiLSTM model is typically better at dealing with longer signals present in the input sample. However, this capability comes with higher memory consumption and computation costs in comparison to those of 1DCNN models. The BiLSTM structure is visulaised in Figure 29.

5.3.3 CNN-BiLSTM

Following the optimisable n-block structure construed by the 1DCNN model, the CNN-BiLSTM also employs n to select a number of stacked 2D convolutions blocks and applies f to select the number of baseline filter. As with the 1DCNN, the number of filters in each convolutional layer of a block increases by a factor of $n \times f$ for each subsequently stacked block and the output is passed through the same fully connected block structure. The first key deviation from the

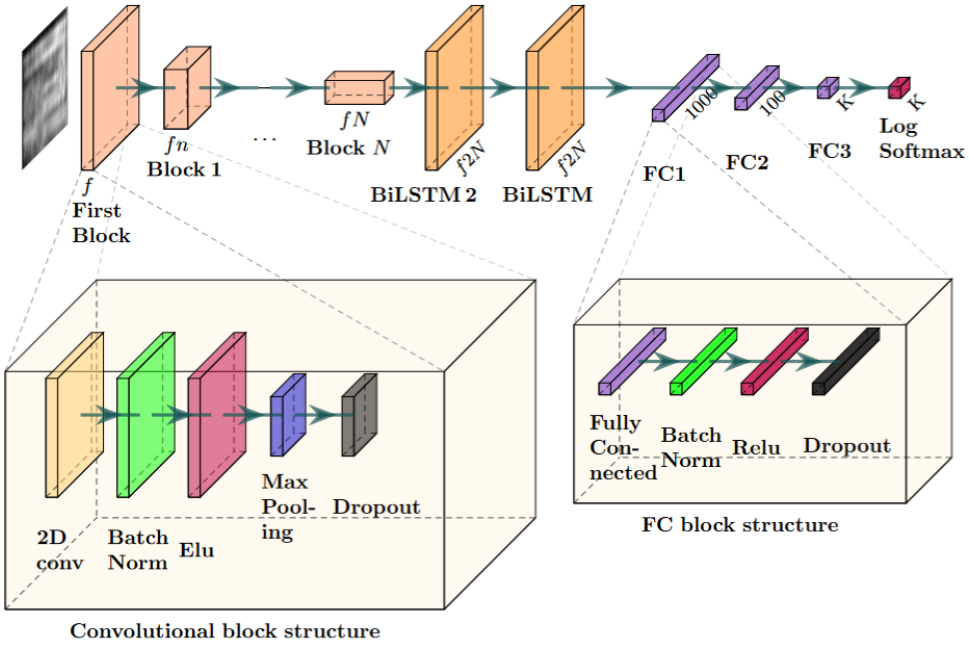


Figure 26: The optimisable ConvolutionalLSTM model structure.

1DCNN structure is an additional convolutional block placed at the beginning of the network to expand the number of channel filters to f, enabling 2D convolutions to work effectively. The second key difference is the addition of two bidirectional convolutional layers appended to the n-block stack of 2d convolutions. The combination of these two elements allows for the 2D convolutions to extract spatial features from the melspectrogram then pass them to the BiLSTM layers to extract sequentially relevant information from these spatial features. The resultant spatio-temporal features are then passed through the fully connected block structure and a log softmax activation layer producing the final predictions. This structure is highlighted in Figure 27

5.4 Experimental Studies

We evaluate each CSO optimised n block model on three well know data sets. CSO is validated on a number of benchmark test functions along with the Wilcoxon Rank Sum Test to validate the statistical significance of the results.

5.4.1 Data sets

To prepare the data for evaluation, the raw audio is first converted to a sample rate of 16kHz then converted into melspectrograms via applying Fast Fourier Transforms to overlapping time

windows producing a spectrogram that is adjusted by the mel-scale. We perform this operation using the *melspectrogram* function from the python package *librosa* with the following settings: *n_fft=4096*, *hop_length=2048*, *n_mels=256*, *pad_mode="reflect"* and, *norm="slaney"*.

Opting for melspectrograms strikes a balance between the temporal and spectral aspects of audio data. melspectrograms offer a comprehensive representation that adeptly preserves time-related nuances while harnessing spectral analysis advantages. They retain vital temporal information, capturing speech's sequential dynamics, including prosody, intonation, and emotional state-related changes. This spectral-temporal insight proves pivotal in discerning emotional cues in speech. melspectrograms further stand out due to their utilization of the mel scale, a perceptually-grounded scale mirroring human pitch perception. This logarithmic scale aligns the feature extraction process with human auditory perception, bolstering the model's capacity to apprehend and interpret emotional cues akin to human perception.

Notably, employing melspectrograms as input for a deep learning model yields substantial benefits. Deep learning models, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs), excel in disentangling intricate patterns within structured data like spectrograms. This capability empowers them to discern subtle emotional cues that may prove elusive when directly extracting information from raw audio. Additionally, melspectrograms streamline dimensionality, fortify resilience against noise, and enhance computational efficiency, collectively enhancing the accuracy of emotion classification.

This approach furnishes a robust audio data representation aligned with the subtleties of emotional expression in speech, capitalizing on the perceptually-grounded mel scale to closely mirror human auditory perception.

Brief descriptions of the evaluation datasets are provided in the following subsections.

Berlin Database of Emotional Speech (Emo-DB)

This data set contains 535 German language utterances from 5 male speakers and 5 female speakers which are stored as wave files with a sample rate of 16kHz. Each utterance belongs to one of seven emotion categories, i.e. boredom, anxiety, anger, sadness, happiness, disgust and neutral.

Surrey Audio-Visual Expressed Emotion (SAVEE)

SAVEE contains 500 English language utterances spoken by four male actors and stored as video files. For this work, only the audio is extracted which is then downsampled from 44.1kHz to 16kHz. Each utterance falls into one of the seven categories: anger, disgust, fear, happiness, sadness, surprise, and neutral. The data train/test split is 360/120 with all speakers being isolated in each split.

Downsampling from 44.1 kHz to 16 kHz was done to align this dataset with the Emo-DB and SAVEE datasets, facilitating more direct comparison. This uniformity at 16 kHz not only streamlines computational efficiency, reduces memory requirements, and mitigates noise but also simplifies data preprocessing, and maintains consistent evaluation metrics. Given that the range of human hearing is approximately between 20 to 20,000Hz, downsampling to 16000Hz entails a loss of high-frequency details, which may impact classification accuracy, particularly if crucial emotional cues reside in those frequencies. It is worth noting however that this frequency range still covers the common audible range of human speech, which typically falls has frequency ranges between 85 Hz 3400 Hz; fundamental frequency (pitch): 85 Hz to 255 Hz and formant frequencies (resonant frequencies): 300 Hz to 3400 Hz. Since emotional cues in speech are often conveyed through variations in pitch, intonation, and spectral characteristics within these frequency ranges, downsampling to 16 kHz is unlikely to significantly affect emotion recognition. The re-sampling operation is handled using the *resample* from the *Librosa* Python package

Toronto Emotional Speech Set (TESS)

The TESS data set consists of two female speakers each speaking 200 words in the context of the phrase ”Say the word <target>”. Each of these phrases are acted out and labeled in accordance with one of the following emotional categories: fear, disgust, surprise, happiness, anger, sadness, pleasant, and neutral. The data set has a total number of 2800 samples with a sample rate of 16kHz. Since there are only two speakers in the data set, the train/test split is 1400/1400 samples to ensure speaker independence.

5.4.2 Emotion Classification with n-Block Deep Learning Models

To evaluate CSO optimisation of deep learning models we test against the following audio emotion data sets: Berlin Database of Emotional Speech (Emo-DB), Toronto Emotional Speech Set (TESS), and the Surrey Audio-Visual Expressed Emotion (SAVEE). Each deep learning model is trained using the ADAM optimiser with a weight decay of 0.005 and batch size of 64. A Cosine Warm-up learning rate scheduler is employed to slowly increase the learning rate up to the selected setting during initial 10% of epochs, and sinusoidally decrease the learning rate for the remaining epochs, until it is near zero at the last epoch. Initially the models are optimised using one of the three optimisation algorithms (i.e., PSO, FA, and CSO) using training and validation data to extract a set of hyper parameters. These hyper parameters are then used as settings for training the model from scratch. First we pre-train each deep learning network on the CREAM-D data set for 10 epochs. Each model is then trained on the target data set for a further 100 epochs. We repeat the experiment 5 times for each deep learning model and data set permutation and present the results in Tables 24. The data sets are split in order to maintain speaker-independence of the test and training sets.

Table 24: The mean classification accuracy over 5 runs for the CSO-optimised deep learning models across three data sets

Data set	Search	CNN-BiLSTM (%)	1DCNN (%)	BiLSTM (%)
Emo-DB	Default	69.4	60.3	59.3
	FA	74.4	68.2	65.4
	PSO	73.8	70.1	72.0
	CSO	**80.8**	73.1	77.0
SAVEE	Default	47.1	51.4	52.2
	FA	52.5	55.0	57.2
	PSO	57.7	57.5	61.8
	CSO	**68.3**	60.3	64.8
TESS	Default	54.5	48.2	45.2
	FA	59.4	51.8	47.5
	PSO	73.5	53.2	58.2
	CSO	**79.1**	56.4	61.6

From the results presented in Table 24, we can see the results for the Emo-DB data set indicate that the CSO-optimised CNN-BiLSTM provides the strongest classification accuracy across all of the models with optimal settings identified by other search methods. Moreover we observe

that the CNN-BiLSTM outperforms all of the other methods for a given optimisation algorithm. This would suggest the advantages gained from both the spatial and temporal feature extraction capabilities embedded in the CNN-BiLSTM architecture with respect to boosting classification performance. In regards to hyper parameter optimisation, it seems that CSO produces the highest classification accuracy across the three search methods. This further implies that CSO is selecting well balanced parameters for the learning rate, number of blocks and filters. This implication will be further analysed in the next subsection. A similar pattern can be seen across the SAVEE and TESS data sets with the CSO-optimised CNN-BiLSTM yielding the highest classification accuracy for both data sets. For the SAVEE data set the BiLSTM model seems to perform reasonably well regardless of optimisation algorithm, suggesting that there may be more temporally relevant underlying patterns present in the data set. The Tess data set shows similar finding to the Emo-DB data set, in this regard showing strong dominance of the CNN-BiLSTM algorithm in comparison to all other models using any of the three optimisation methods. Relative comparison of the baseline results shows that optimisation for these models on these data sets produces improvements consistently. To further inspect some of these implications, we conduct in-depth analysis of the specific optimized hyper parameters in reference of their performances in the following subsection.

5.4.3 Hyper parameters Analysis

In Tables 25, 26, and 27 the devised average learning rates, number of blocks and number of filters/hidden units over a set of 5 runs are presented for the CNN-BiLSTM, 1DCNN and BiLSTM models, respectively.

By observing the hyper parameter graphs shown in Figure 27, those selected by CSO for the CNN-BiLSTM model are generally tightly clustered with high accuracy indicating a strong positive consensus on the stability and performance of the CSO algorithm. In contrast, PSO shows occasionally shows a tight grouping of selected hyper parameters but with lower accuracy suggesting it is becoming trapped in local optima. FA trained CNN-BiLSTM models show the worst classification accuracies with the seemingly random spread of hyper parameter groupings which is likely caused by poor initialisation combined with oscillatory movements induced by competition between high intensity regions. CSO consistently avoids such poor configurations identified by

Table 25: Average hyper parameters identified over 5 runs for optimisation on the CNN-BiLSTM model

Data set	Search	Lr	n	f	acc (%)
Emo-DB	CSO	0.000406	3.6	37.0	**80.4**
	PSO	0.000648	3.2	50.0	74.2
	FA	0.000586	2.6	65.6	74.4
SAVEE	CSO	0.000323	3.2	48.8	**68.3**
	PSO	0.000288	4.2	59.8	57.7
	FA	0.000691	2.8	30.0	50.2
TESS	CSO	0.000306	3.4	40.4	**79.1**
	PSO	0.000462	4.0	74.2	73.5
	FA	0.000406	3.6	41.4	57.3

FA that have extremely low Learning Rate (0.00001) or number of filters (1). Since these configurations have such a large effect they are classed as outliers and therefore not used for fitting the polynomial lines for other hyper parameter graphs on the same data set. These fitted polynomial lines help emphasise the rough regions of good or poor optimally for each hyper parameter. Across each data set, they highlight that the optimal regions are moderately low learning rates, moderate to low number of filters, and a moderate number of blocks. More specifically these regions seem to center around 0.00035, 3.5, and 35 for the learning rate, blocks and filters respectively which is supported by the average values present in Table 25. These average results also confirm the more general observation that the optimal configuration is moderate/low learning rate moderate/low number of filters with a moderate number of blocks. The only exception to this are the average hyper parameters results for the FA optimised CNN-BiLSTM on the TESS data set, which are similar to those determined by CSO but have much lower accuracy rates. Looking more closely at Figure 27 it is evident that FA's wider and more random selection of hyper parameters skew the average results in Table 25 such that this specific result can be considered an anomaly. Also of note is the fact only one run selected the highest possible number of blocks ($n = 6$) which suggests the benefit of depth is limited by the vanishing gradient problem present in networks that do not employ skip connections [130]. The tendency toward lower filters indicates that width is less beneficial to the models learned representation than depth. Wider networks facilitate learning more distinct independent patterns simultaneously however, there is no mechanism which forces the filters to capture useful information. This can lead to redundant filters in the network that contribute nothing of value. The moderately low learning rate would allow for this configura-

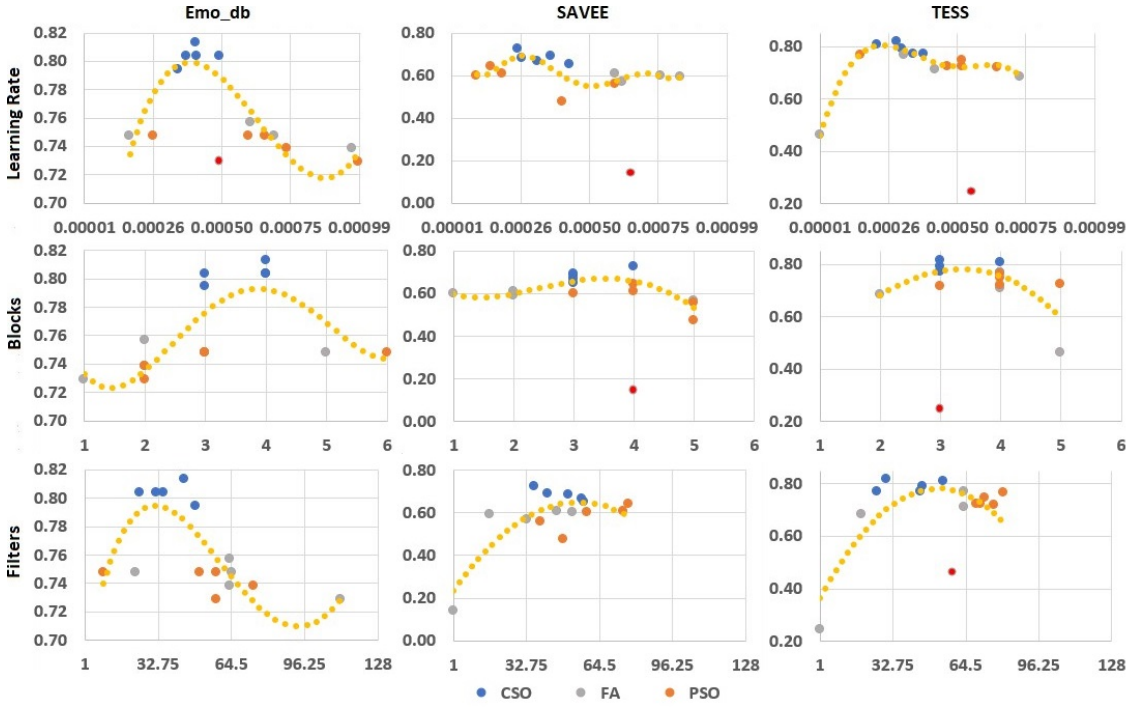


Figure 27: A grid of scatter plots displays the results of every run of each optimized CNN-BiLSTM model. Each column of graphs represents the runs associated with a specific data set, and each row shows the results in relation to a particular hyperparameter: learning rate, number of blocks, and filters. The data from all runs of every optimization method is plotted as a fitted polynomial line in yellow. Some outliers are excluded from the line of best fit and are marked in red.

tion to have sufficiently sized updates to the weights to tune the internal representation without over-fitting or getting trapped in less optimal configurations.

The optimal configurations for the 1DCNN, as shown in Figure 28, exhibit similar trends to the CNN-BiLSTM, with optimal learning rates and numbers of filters being moderate to low across all data sets. Some deviation is observed for the number of blocks, which has a region of optimality at a moderately high level. This trend may be influenced by the lower representational capacity of the 1DCNN compared to the CNN-BiLSTM. The latter model has three additional layers, including an initial two-dimensional CNN block and two BiLSTM layers at the end, which increase the baseline network depth and exacerbate the vanishing gradient problem. The lower representational capacity of the 1DCNN due to its lower dimensionality may also require a deeper network to enable the model to learn a good generalized representation of the data. The optimal regions of high numbers of blocks with moderate to low learning rates and numbers of filters are confirmed by the average results displayed in Table 26.

Table 26: Average hyper parameters identified over 5 runs for optimisation on the 1DCNN model

Data set	Search	Lr	n	f	acc (%)
Emo-DB	CSO	0.000454	4.4	51.0	**73.1**
	PSO	0.000474	3.4	78.6	70.1
	FA	0.000351	2.2	63.0	68.2
SAVEE	CSO	0.000435	4.4	51.6	**60.3**
	PSO	0.000458	3.0	66.2	57.5
	FA	0.000961	2.2	52.8	55.0
TESS	CSO	0.000338	4.4	55.8	**56.4**
	PSO	0.000343	4.0	73.6	53.1
	FA	0.000126	3.8	54.0	51.8

Table 27: Average hyper parameters identified over 5 runs for optimisation on the BiLSTM model

Data set	Search	Lr	n	f	acc (%)
Emo-DB	CSO	0.000400	3.4	44.8	**77.0**
	FA	0.000395	2.6	50.0	63.2
	PSO	0.000449	4.0	84.0	72.0
SAVEE	CSO	0.000473	4.0	61.6	**64.8**
	FA	0.000397	4.0	89.4	57.2
	PSO	0.000625	3.4	60.0	61.8
TESS	CSO	0.000398	3.4	57.8	**60.2**
	FA	0.000633	2.6	48.8	47.5
	PSO	0.000303	3.0	96.8	58.2

Examining Figure 29 and Table 27 reveals that the BiLSTM model exhibits similar trends, with optimal configurations consisting of hyperparameters with a moderately low learning rate, a moderately low number of hidden units, and a moderate number of blocks. As with the previous networks, depth is preferred over width. Width in this case refers to the number of hidden units, and having too many can result in capturing too much irrelevant information from the signal, while having too few can reduce the ability to extract important signals from the data. Despite the significant differences in approach between BiLSTM networks and CNNs, the overall configurations of moderately low learning rate, higher depth, and moderately low width appear to be beneficial across all three models.

Out of the three models, the 1DCNN typically produces the worst results, followed by the BiLSTM, and then the CNN-BiLSTM. Since melspectrograms are essentially a visual representation of sequential frequency data, it is unsurprising that the BiLSTM performs well, as it is specifi-

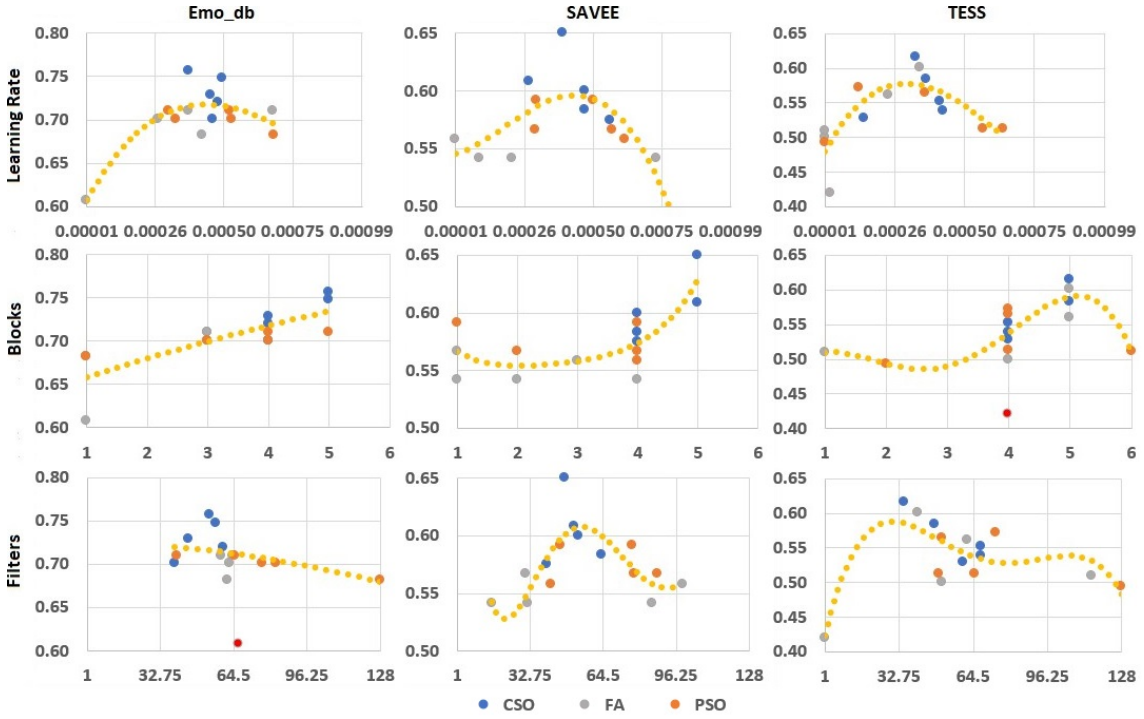


Figure 28: A grid of scatter plots displays the results of every run of each optimized 1DCNN model. Each column of graphs represents the runs associated with a specific data set, and each row shows the results in relation to a particular hyperparameter: learning rate, number of blocks, and filters. The data from all runs of every optimization method is plotted as a fitted polynomial line in yellow. Some outliers are excluded from the line of best fit and are marked in red.

cally designed to incorporate 1-dimensional temporal information. The additional accuracy improvements achieved by the CNN-BiLSTM can be attributed to its ability to capture both the frequency information embedded in the image spatially and the larger sequential aspects of the input data.

Across all models and data sets, the top performing classification accuracies come from models trained with CSO optimised hyper parameters. Due to the clustering dynamic allocation of particles that search the global best position CSO has an advantage over both PSO and FA where since particles can focus directly on exploration or exploitation.

The results of all models and data sets demonstrate that CSO outperforms PSO, which in turn outperforms FA. The superior performance of PSO over FA is often attributed to the incorporation of time-based information in the particles' previous best positions and the global best position, whereas FA only utilizes the current state of the fireflies and their intensities. CSO builds upon

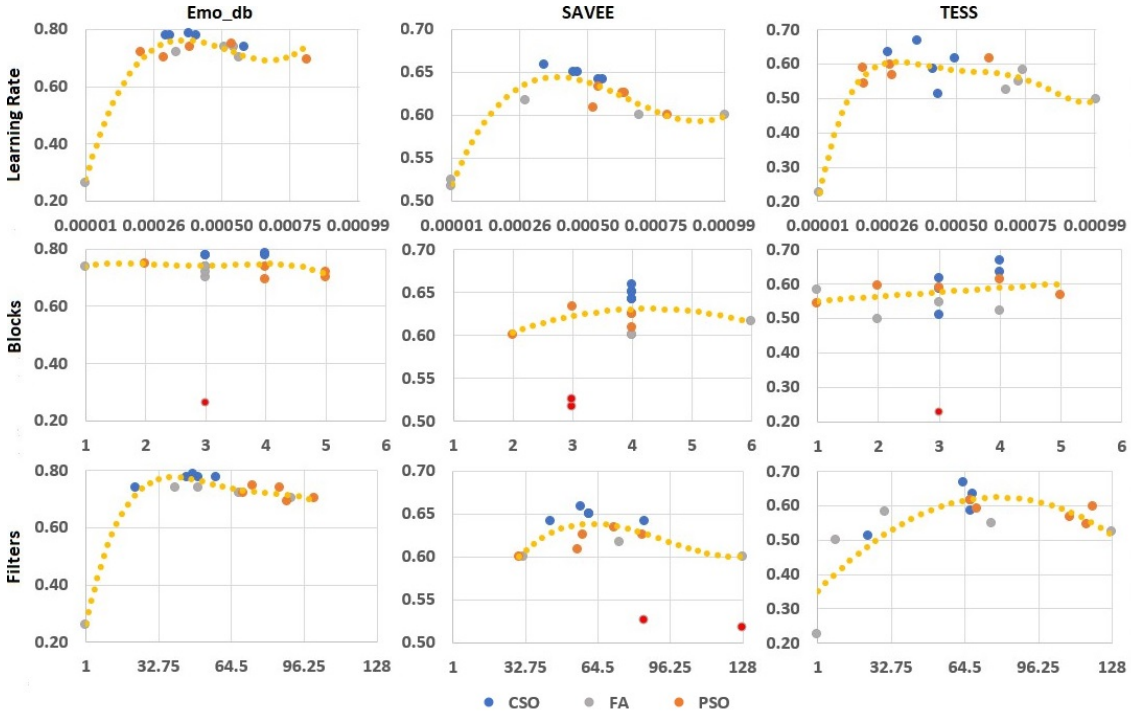


Figure 29: A grid of scatter plots displays the results of every run of each optimized BiLSTM model. Each column of graphs represents the runs associated with a specific data set, and each row shows the results in relation to a particular hyperparameter: learning rate, number of blocks, and hidden units (labeled filters). The data from all runs of every optimization method is plotted as a fitted polynomial line in yellow. Some outliers are excluded from the line of best fit and are marked in red.

this idea by tracking previous particle positions and fitnesses through clustering. This allows multiple potential sites for global solutions to be identified and used to inform the particles. The use of cluster centers also provides a more general direction for velocity updates during exploration, helping particles avoid local optima traps by not searching directly near their personal best locations, as they would in PSO. The Global Search Particle Selection Threshold in CSO enables mutually exclusive searches between exploration and exploitation, allowing the desired search behavior to be tailored to a specific goal. The necessity of each search is then determined based on the current state of convergence compared to the worst and best states, encouraging exploitation when cluster centroids are in consensus and exploration when clusters are more dispersed. Additionally, since one particle always searches the global best position regardless of algorithm convergence, a thoroughly investigated local optimum will always be found, even when overall convergence is poor. These advantages are reflected in the typically close groupings of CSO-selected hyperpa-

rameters in Figures 27, 28, and 29. In comparison, the groupings of PSO hyperparameters tend to be slightly less consistent, and the groupings of FA hyperparameters often appear spread out and random.

Table 28: Comparison with other Reported Results

Method	Acc (%)
Emo-DB	
CSO CNN-BiLSTM	**80.4**
DGA [91]	77.5
eGeMAPs and emobase [93]	76.9
NNPM [90]	55.9
CNN-LSTM (Parry et. al.) [89]	69.7
LSTM (Parry et. al.) [89]	59.7
CNN (Parry et. al.) [89]	58.9
SAVEE	
CSO CNN-BiLSTM	68.3
DGA[91]	69.9
LSTM (Parry et. al.) [89]	60.6
CNN (Parry et. al.) [89]	70.6
CNN-LSTM (Parry et. al.) [89]	**72.7**
$\text{CNN}_{\text{Wavegram-Logmel}}$ [92]	65.5
DNN [131]	59.7
eGeMAPs + emobase [93]	42.4
TESS	
CSO CNN-BiLSTM	**79.1**
2D CNN with global avg. pool [132]	62.0
LSTM (Parry et. al.) [89]	45.1
CNN (Parry et. al.) [89]	48.9
CNN-LSTM (Parry et. al.) [89]	49.5

To place these result into a wider context we compare them with other relevant audio emotion classification research as shown in Table 28. The proposed CSO optimised CNN-BiLSTM outperforms all results on the Emo-DB with 80.4% classification accuracy.The closest result is from the Kanwal et. al. [91] which used DGA to perform feature selection of OPENSMILE which are classified by an SVM. The reduced set of features remove mostly redundant information allowing for the simple SVM type classifier to focus on learning clear distinct patterns from the data. In comparison the CSO optimised CNN-BiLSTM model learns from mel adjusted frequency-time data allowing for more subtle patterns in the data to learned via the more complex representation facilitated by the compounded layers of CNN and BiLSTM layers. In comparison to a similar CNN-BiLSTM network by Parry et. al. [89] we see a 10.7% performance improvement. Though this work was focused on cross corpus training, it is worth noting only four classes from the datasets were used for their single corpus training which we report here. Additionally, they in-

dicate that the lack of model optimisation is likely to be a cause for the low performance in this instance. Conversely, the model from this work employs CSO optimisation yielding extra performance from the robust parameter selection enabled by the cluster convergence monitoring and resultant particle assignment between the cluster center guided search and the global best guided search.

The CNN-BiLSTM results on the SAVEE data set are not as competitive as those for the EMO-DB data set yielding only 68.3%. Comparing against Parry et. al. [89] CNN-BiLSTM model we see a significant performance gap of 4.4%. This is in part due to the lower number of classes used in training in Parry's CNN-BiLSTM model but also due to a discrepancy in train test split. In their work the use a favourable 80%, 10%, and 10% for the train/validation/test splits. In this work a 75%, 25% train/test split is used for the final evaluation with 20% of the train data being used as validation for determining the hyper parameters used to train the final networks with optimised parameters. Due to the fact the dataset only has four actors this is the only way to ensure speaker independence, meaning that none of the actors are found in both the train set or the test set. Both the reduced classes and the lack of speaker independence of the in the data for the other models are high likely contributing to the higher classification accuracies. However, it is worth noting that the SAVEE dataset undergoes a significant downsampling from 48kHz to 16kHz in this work, which could also be affecting the result.

On the TESS dataset we see a classification performance of 79.1% by the CSO optimised CNN-BiLSTM. This result is higher than second most performant model, The 2D CNN with global average pooling model by Venkataramanan et. al. [132] yielding 62.0% and thus a performance improvement of 17.1%. The main focus of this work was on separating gender as part of the emotion classification producing 14 classes from the 7 emotion categories and gender. Despite this, they do report classification accuracy on the normal 7-class emotion problem which we report in Table 28. They discuss that for their 2D-CNN that four CNN layers seems to be optimal for their network however they do not apply optimisation to obtain this result nor do they adjust the number of filters. Since we optimise learning rate number of filters and CNN blocks, the CNN-BiLSTM model can better adapt the architecture by reducing the width and increasing the depth of the network affording increases in classification accuracy.

5.4.4 Benchmark Functions for CSO Evaluation

To further evaluate the CSO algorithm, it was tested on 10 well-established benchmark functions against 12 other search methods, including Adaptive Random Search (ARS) [126], Arithmetic Optimisation Algorithm (AOA) [125], Autonomous Particle Groups PSO (AGPSO) [127], Cuckoo Search (CS) [124], FA [117], Flower Pollination Algorithm (FPA) [123], Genetic Algorithm (GA) [122], Gravitational Search Algorithm (GSA) [121], Jaya [120], Memetic Algorithm (MA) [119], PSO, and Random Search (RA) [118]. Each search algorithm was evaluated on each benchmark 30 times with 50 search agents, 500 iterations, and using 30 dimensions for the test function. The benchmark functions presented unique challenges for search algorithms, with multiple local optima and distinct search spaces. Results for the Ackley, Dixon Price, Griewank, Powell, Rastrigin, Rosenbrock, Rotated Hyper-Ellipsoid, Sphere, Sum Squares, and Zakharov functions are shown in Table 29.

Table 29: Evaluation results for the benchmark functions with dimension=30

		RA	MA	Jaya	GSA	GA	FPA	CS	AOA	ARS	PSO	FA	AGPSO	CSO
Ackley	mean	1.82E+01	1.73E+01	1.91E+01	1.61E+01	1.91E+01	1.59E+01	1.88E+01	1.96E+01	1.78E+01	1.81E+01	1.96E+01	1.54E+01	**1.42E+01**
	min	1.77E+01	1.58E+01	1.78E+01	1.49E+01	1.85E+01	1.37E+01	1.75E+01	1.83E+01	1.66E+01	1.76E+01	1.86E+01	1.42E+01	**1.19E+01**
	max	1.85E+01	1.83E+01	1.97E+01	1.67E+01	1.97E+01	1.74E+01	1.92E+01	2.01E+01	1.82E+01	1.84E+01	1.99E+01	1.69E+01	**1.57E+01**
	std	2.09E-01	5.77E-01	3.71E-01	4.32E-01	3.27E-01	9.35E-01	3.88E-01	4.03E-01	3.69E-01	**1.75E-01**	2.60E-01	5.05E-01	8.53E-01
DixonPrice	mean	6.26E+05	3.13E+04	1.42E+06	1.69E+06	9.34E+04	6.28E+04	8.50E+05	1.62E+06	5.56E+05	4.13E+03	1.64E+06	5.26E+02	**8.79E-01**
	min	3.93E+05	1.84E+04	6.87E+05	5.11E+05	1.07E+03	3.51E+04	3.72E+05	9.30E+05	3.64E+05	2.35E+03	7.65E+05	2.80E+02	**6.67E-01**
	max	8.93E+05	4.40E+04	2.23E+06	2.28E+06	7.11E+05	1.11E+05	1.20E+06	2.32E+06	7.32E+05	5.67E+03	2.22E+06	1.05E+03	**2.83E+00**
	std	1.25E+05	6.10E+03	3.46E+05	3.82E+05	1.62E+05	2.17E+04	1.98E+05	3.28E+05	9.89E+04	7.65E+02	3.34E+05	1.54E+02	**5.23E-01**
Griewank	mean	3.63E+02	1.66E+02	5.24E+02	8.73E+01	4.78E+02	9.87E+01	4.08E+02	5.73E+02	3.43E+02	3.36E+01	5.90E+02	2.03E+01	**3.69E+00**
	min	2.90E+02	5.56E+01	3.92E+02	6.40E+01	3.70E+02	5.94E+01	2.97E+02	4.07E+02	2.82E+02	2.66E+01	5.06E+02	1.49E+01	**1.26E-02**
	max	4.24E+02	2.65E+02	6.27E+02	1.09E+02	5.58E+02	1.30E+02	4.90E+02	6.87E+02	3.91E+02	3.97E+01	7.09E+02	2.49E+01	**1.03E+01**
	std	3.06E+01	4.97E+01	5.53E+01	1.12E+01	4.33E+01	1.65E+01	3.87E+01	6.60E+01	3.09E+01	3.16E+00	4.94E+01	**2.69E+00**	**2.77E+00***
Powell	mean	7.78E+08	8.78E+08	1.44E+12	1.60E+13	1.60E-02	1.41E+06	3.04E+10	3.18E+13	5.36E+08	1.24E+03	2.97E+12	1.99E+01	**2.24E-03**
	min	2.06E+07	2.20E+07	6.15E+08	2.43E+09	2.33E-03	1.88E+03	1.42E+08	1.22E+10	7.25E+06	9.72E+01	6.03E+09	9.10E+00	**4.84E-34**
	max	3.62E+09	4.17E+09	1.07E+13	1.22E+14	**5.18E-02**	1.78E+07	1.28E+11	2.82E+14	2.43E+09	5.85E+03	1.84E+13	4.24E+01	**6.72E-02***
	std	8.07E+08	8.90E+08	2.02E+12	2.92E+13	**1.06E-02**	3.19E+06	3.39E+10	6.63E+13	5.71E+08	1.13E+03	4.44E+12	7.95E+00	**1.21E-02***
Rastrigin	mean	3.37E+02	3.40E+02	4.01E+02	2.10E+02	3.80E+02	2.93E+02	3.61E+02	4.29E+02	3.31E+02	1.73E+02	4.22E+02	1.11E+02	**1.38E+01**
	min	3.01E+02	2.91E+02	3.33E+02	1.65E+02	3.20E+02	2.59E+02	3.19E+02	3.85E+02	3.00E+02	1.38E+02	3.75E+02	1.59E+01	**3.98E+00**
	max	3.70E+02	3.63E+02	4.50E+02	2.40E+02	4.12E+02	3.19E+02	3.98E+02	4.62E+02	3.47E+02	1.94E+02	4.60E+02	1.79E+02	**3.18E+01**
	std	1.54E+01	1.73E+01	2.48E+01	1.73E+01	1.87E+01	1.51E+01	1.97E+01	2.17E+01	1.27E+01	1.39E+01	1.88E+01	7.00E+01	**6.92E+00**
Rosenbrock	mean	4.29E+05	8.25E+04	1.02E+06	1.43E+06	5.29E+04	5.50E+04	7.70E+05	1.32E+06	3.85E+05	1.66E+05	1.45E+06	9.22E+04	**5.27E+01**
	min	2.19E+05	5.15E+04	2.97E+05	4.34E+05	4.19E+03	2.67E+04	2.21E+05	5.20E+05	2.10E+05	1.36E+05	8.78E+05	2.65E+04	**1.83E-01**
	max	6.10E+05	1.11E+05	1.76E+06	2.08E+06	1.86E+05	9.02E+04	1.08E+06	1.93E+06	4.88E+05	1.97E+05	2.06E+06	1.21E+05	**3.09E+02**
	std	1.09E+05	1.26E+04	3.35E+05	3.60E+05	4.88E+04	1.77E+04	2.14E+05	3.50E+05	6.18E+04	1.25E+04	3.24E+05	2.04E+04	**6.15E+01**
RotatedHyperEllipsoid	mean	2.23E+05	6.39E+04	3.47E+05	3.98E+05	2.66E+05	5.91E+04	2.54E+05	3.97E+05	2.12E+05	2.04E+04	3.89E+05	7.85E+03	**1.11E-02**
	min	1.79E+05	4.39E+03	2.49E+05	2.76E+05	2.04E+05	3.64E+04	1.58E+05	2.97E+05	1.51E+05	1.44E+04	3.05E+05	4.11E+03	**2.50E-08**
	max	2.70E+05	1.09E+05	4.38E+05	4.67E+05	3.13E+05	9.17E+04	3.18E+05	5.03E+05	2.49E+05	2.48E+04	4.84E+05	1.49E+04	**2.98E-01**
	std	2.51E+04	2.43E+04	4.61E+04	4.19E+04	2.85E+04	1.46E+04	3.22E+04	4.67E+04	1.92E+04	2.86E+03	3.55E+04	2.65E+03	**5.36E-02**
Sphere	mean	1.03E+02	1.05E+02	1.51E+02	7.69E+00	1.40E+02	2.60E+01	1.19E+02	1.73E+02	1.01E+02	9.20E+00	1.71E+02	5.65E+00	**1.62E+00**
	min	7.41E+01	8.19E+01	1.20E+02	4.16E+00	1.09E+02	1.65E+01	7.39E+01	1.34E+02	7.22E+01	7.07E+00	1.43E+02	4.45E+00	**1.13E-04**
	max	1.22E+02	1.21E+02	1.83E+02	1.23E+01	1.59E+02	3.89E+01	1.44E+02	2.04E+02	1.17E+02	1.11E+01	2.01E+02	6.79E+00	**5.59E+00**
	std	1.09E+01	1.06E+01	1.53E+01	2.20E+00	1.25E+01	5.65E+00	1.59E+01	1.65E+01	8.68E+00	9.81E-01	1.52E+01	**5.36E-01**	1.31E+00
SumSquares	mean	1.43E+03	1.30E+03	2.23E+03	7.88E+01	1.56E+03	3.59E+02	1.59E+03	2.43E+03	1.35E+03	1.34E+02	2.40E+03	7.60E+01	**1.60E-05**
	min	1.17E+03	1.08E+03	1.75E+03	3.14E+01	9.39E+02	2.25E+02	1.18E+03	1.49E+03	7.81E+02	1.04E+02	1.84E+03	5.39E+01	**1.05E-10**
	max	1.70E+03	1.50E+03	2.76E+03	1.35E+02	2.10E+03	5.51E+02	1.97E+03	3.01E+03	1.56E+03	1.70E+02	2.90E+03	9.20E+01	**1.39E-04**
	std	1.34E+02	1.17E+02	2.37E+02	2.04E+01	2.30E+02	7.15E+01	2.17E+02	3.12E+02	1.54E+02	1.76E+01	2.80E+02	9.57E+00	**3.47E-05**
Zakharov	mean	4.12E+02	2.44E+02	7.18E+02	8.35E+08	3.59E+02	4.24E+02	5.39E+02	4.91E+08	3.71E+02	2.30E+08	5.11E+07	2.20E+02	**3.16E+01**
	min	2.61E+02	1.74E+02	4.08E+02	1.48E+03	2.13E+02	3.25E+02	2.13E+02	6.17E+02	2.63E+02	5.92E+02	6.14E+02	2.91E+01	**1.22E+01**
	max	5.38E+02	3.47E+02	1.02E+03	1.01E+10	6.72E+02	5.28E+02	7.40E+02	4.26E+09	4.52E+02	3.97E+09	1.00E+09	6.95E+02	**5.55E+01**
	std	5.93E+01	4.57E+01	1.45E+02	1.85E+09	8.86E+01	6.26E+01	1.25E+02	8.72E+08	4.91E+01	7.47E+08	1.93E+08	1.68E+02	**9.73E+00**

The results indicate that CSO consistently outperforms all other search algorithms in terms of the mean and minimum results across all test functions. In terms of the maximum value found on any given test function, CSO produces the lowest maximum values, except on the Powell func-

tion where it achieves the second-best results. Similarly, for the standard deviation metric, CSO achieves the lowest result except on the Griewank, Powell, and Sphere functions, with the latter two yielding the second-best results. Overall, CSO performs very well across all test functions. These results are statistically verified using the Wilcoxon Rank Sum Test, as shown in Table 30, which indicates that all results have a p-value lower than 0.05. The statistical analysis using the Wilcoxon Rank Sum Test allows us to test the null hypothesis, which is the assumption that the data samples between any two search algorithms are from continuous distributions with equal medians. The results of the test show that the null hypothesis can be rejected for all of the results presented in Table 29, indicating that the differences in performance between the search algorithms are statistically significant and not just random fluctuations. This suggests that the differences in performance between the different search algorithms are genuine and not just random deviations.

Table 30: The Wilcoxon rank sum test results for the benchmark functions over 30 runs

	Ackley	DixonPrice	Griewank	Powell	Rastrigin	Rosenbrock	RotHyp	Sphere	SumSquares	Zakharov
RA	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
MA	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
Jaya	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
GSA	1.46E-10	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	8.15E-11	3.02E-11	3.02E-11
GA	3.02E-11	3.02E-11	3.02E-11	5.57E-10	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
FPA	6.01E-08	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
CS	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
AOA	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
ARS	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
PSO	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
FA	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
AGPSO	3.52E-07	3.02E-11	3.02E-11	3.02E-11	1.31E-08	3.02E-11	3.02E-11	2.15E-10	3.02E-11	1.07E-09

5.5 Conclusion

In conclusion, this research focused on the application of deep learning models for emotion classification, particularly the CNN-BiLSTM architecture, across three distinct emotion datasets. We introduced the Cluster-based Symbiotic Organisms (CSO) optimization algorithm and compared its performance with Particle Swarm Optimization (PSO) and Firefly Algorithm (FA) for hyperparameter tuning.

The findings indicate that CSO consistently outperforms PSO and FA, producing optimal hyperparameter settings characterized by moderate to low learning rates, filter numbers, and block or hidden unit counts. These optimal configurations, when applied to deep learning models, lead to

improved classification accuracy.

Moreover, the CSO-optimized CNN-BiLSTM model exhibited impressive results, surpassing existing state-of-the-art approaches on certain emotion datasets, showcasing its effectiveness in capturing nuanced emotional patterns in audio data.

Additionally, evaluation was performed on benchmark functions, where CSO demonstrated competitive performance compared to a significant number of other optimization algorithms.

Overall, this research underscores the significance of hyperparameter optimization and the efficacy of the CSO algorithm in enhancing deep learning model performance for emotion classification.

5.6 Limitations of the Work

While this research has provided valuable insights into the application of deep learning models and the CSO algorithm for emotion classification, it is essential to acknowledge the inherent limitations in the study. Recognizing these limitations is crucial for understanding the scope and potential constraints of the findings. In the following list, we outline several key limitations that encompass various aspects of the research, including dataset selection, preprocessing choices, algorithmic considerations, and more. These limitations should be taken into account when interpreting the results and may guide future research directions for further refinement and exploration.

1. Limited Dataset Variety: The study focused on three specific emotion datasets, which may not fully represent the diversity of emotional expressions found in real-world scenarios. Generalizability to other emotional contexts could be limited.

2. Downsampling: The downsampling of audio from 48kHz to 16kHz in the SAVEE dataset may result in the loss of high-frequency information, potentially affecting classification accuracy. However, the information lost is not likely to be significant as much of the import speech audio is in much lower frequency ranges, even so there could be very subtle signals in the higher frequency ranges that are useful for classification.

3. Speaker Independence: While speaker independence was maintained in the test/train splits, the impact of variations in speakers' emotional expressions within the same emotion cate-

gory was not extensively studied.

4. Ethnic and Gender Bias: The datasets' demographics (e.g., German and English speakers) may introduce biases related to ethnicity and gender in emotion classification, which were not explicitly considered.

Addressing these limitations or considering them in future research can enhance the robustness and applicability of the findings.

Chapter 6

Conclusions

6.1 EnvPSO Optimised Multi-stream CNN for Human Action Recognition

The research in Chapter 3 was conducted to address the following research questions:

- Can greater classification performance be achieved on still image human action recognition tasks by employing CNN and Swarm Optimisation techniques?
- What methods can be employed to improve upon the PSO algorithm to reduce it's tendency to fall into local optima?
- Can the number of re-trainable layers in the transfer learning process be encoded such that it can be searched by optimisation algorithm?

To address the first question, a multi-stream CNN ensemble model for undertaking human action recognition was proposed. Additionally, A new PSO variant, denoted as EnvPSO, has been designed to perform automatic optimal hyper parameter selection. To address the issues highlighted in question two, the proposed PSO variant incorporates a Gaussian fitness surface estimation method and exponential adaptive coefficients to search for global optimality. Specifically, the time-varying exponential coefficients optimally calibrate the contribution of both social and cognitive components during each iteration, while gradient information yielded by the Gaussian fitness estimation surface is used to guide the search agents towards promising search regions. The third

question is addressed by the proposed Layer Strip-back optimisation parameter that determines the number of re-trainable layers of a stream CNN model at the fine-tuning stage.

This multi-stream ensemble model integrates the three EnvPSO optimised CNN streams that are subsequently ensembled for action classification. The ensemble diversity is not only enhanced by diverse learned representations of differing CNN networks with optimised distinctive transfer learning configurations, but also enriched by various input channels using raw images and Mask R-CNN segmented salient features. The empirical results indicate that EnvPSO yields significant efficiency in hyper parameter selection for optimising each CNN stream in the ensemble. Evaluated using two still image human action data sets, i.e. BU-101 and Willow7, the proposed multi-stream CNN ensemble model with EnvPSO hyper parameter optimisation shows improved performance and outperforms the counterparts devised by PSO and other state-of-the-art existing studies. Considering the results, the proposed search strategies, which include Gaussian fitness surface estimation and exponential coefficients, seem to account for the better efficiency of our devised ensemble model with better generalised internal representations of diverse action classes. The proposed PSO variant shows statistically significant improvements against a number of classical and advanced search methods for solving diverse unimodal and multimodal benchmark functions, as confirmed by Wilcoxon rank sum tests.

6.2 Proposed Neural Inference Search for Multi-loss Segmentation Models

Work presented in Chapter 4 was conducted to address the following research questions:

- Which loss functions perform well on CNN based segmentation tasks and what benefit if any, can be achieved through compounding them?
- Noting the flaws inherent in PSO, Is it possible to enhance or replace the personal and global best search mechanisms that typically lead to local optima traps?

This research has proposed a combination of Cross Entropy, Dice, and Focal Loss to train CNN models for undertaking semantic segmentation tasks. A novel NIS algorithm has been devised to optimise multi-loss and learning parameters. Three new search behaviours have been formulated, i.e. LSTM-CNN based Maximised Standard Deviation and Local Best Velocity Prediction, as well

as n-dimensional angle rotation, to improve search diversity. A Staged Discrete Adaptive Wave function has also been exploited for implementing a stage-based behaviour scheduling to precisely balance the contribution of each behaviour pertaining to intensification and diversification during the course of hyper parameter search.

In regards to question 1 the results indicate that using multiple loss functions leads in this case led to models with superior performance in comparison with those using single loss functions and that the Dice score typically yields the best performance comparatively when only using single loss functions. The method takes the advantage of complementary information embedded within each loss function. It's crucial to emphasize that while this observation holds true for Segmentation Datasets, its applicability to other machine learning models or problem domains may vary significantly and is inherently problem-specific. Additionally, it's essential to acknowledge that diverse problem domains necessitate distinct loss functions. For instance, regression-based problems often benefit from MSE-like loss functions. When combining multiple loss functions, it becomes imperative to ensure that each aligns with the specific problem domain and offers unique, complementary signals. To validate these assumptions, further evaluation across different models and problem domains would be required.

Addressing question 2, the results comparing NIS against the original PSO and FA on the CamVid, Freiburg Forest and MESSIDOR datasets reveal the effectiveness of the proposed novel behaviours and new operation scheduling regimen introduced in NIS on these specific datasets. This has been further confirmed by the notable performance of NIS against 12 well-known and modern search methods on several multimodal and unimodal benchmark functions indicating improved ability to avoid local optima traps. Thus, both multiple loss functions and NIS optimised loss coefficients and learning configurations have shown great benefits on the given tasks, working in tandem to further enhance the performance. It's possible that applying these techniques to other domains and model architectures could yield similar benefits.

6.3 Proposed Cluster Search Optimisation of Deep Learning Models for Audio Emotion Classification

The research presented in Chapter 5 aims to answer the following questions:

- Can the structure and hyperparameters of a deep learning network be optimized to improve performance on audio emotion classification problems?
- What types of deep learning architectures are suitable for the audio emotion recognition task, and how can their structure be encoded for optimization?
- What kinds of search mechanisms can effectively adapt between global and local searches during the optimization process, avoiding traps in local optima and premature convergence?"

To address these questions, we proposed CSO, a novel search optimization algorithm for optimizing the hyperparameters and architecture of n-block deep learning models including 1DCNN, BiLSTM, and CNN-BiLSTM. CSO uses OPTICS clustering of particle positions and fitness values to introduce two key mechanisms: the Cluster Distance Improvement metric and a new cluster-based velocity target for exploratory search behavior. The Cluster Distance metric guides particles to perform global or local searches depending on the convergence of the clusters, helping to prevent premature convergence around the current global best position. The use of cluster centers as velocity targets ensures that particles performing local searches are much less likely to get stuck in local optima while exploring. This is suggested in the hyperparameter analysis, where CSO consistently produces hyperparameter values a higher accuracy compared to FA and PSO. These results were further strengthened by statistically significant evaluations on benchmark test functions, where CSO consistently outperformed 12 other search algorithms. CSO also demonstrated competitive performance against several relevant comparative works in the field.

The improved performance of the optimized 1DCNN, BiLSTM, and CNN-BiLSTM models directly addresses research question 1, demonstrating that optimization of both structure and hyperparameters leads to increased classification accuracy for the audio emotion classification task on the Emo_db, SAVEE and TESS datasets (as shown in Table 24). This suggest improvements could be gained when applied to other audio emotion classification tasks as well, but further evaluations would be needed to confirm this. For research question 2, The similar architectures to the (1DCNN, BiLSTM, and CNN-BiLSTM) used for this study are commonly used for sequence classification. The inclusion of n-block model encoding and dynamic filter/hidden unit scaling provided sufficient flexibility for the search algorithms to select useful architectural configurations that improved the classification accuracy on the target datasets. In particular, the tendency towards

deeper, thinner networks showed increased performance over the default and poorly optimized models (as shown in Tables 25, 27, and 26). For research question 3, the classification accuracy improvements shown in these tables suggest that the proposed Cluster Distance Improvement metric and cluster center velocity targets address the issues with local optima traps caused by personal best and global best guided searches in PSO.

6.4 Future Work

The success of using swarm algorithms to optimize deep learning models across various modalities and machine learning tasks suggests that this approach is promising. However, there is still room for further research and development in this area. Some potential avenues for future work include:

1. Enhancing the mechanisms used in EnvPSO with different interpolation methods for surface estimation. While the Gaussian function currently used works well enough to predict the surface, it is not necessarily the best option. It is however challenging to extend such functions into a generalized n-dimensional formula, which would be necessary for high-dimensional search spaces (e.g., optimizing n hyperparameters). In particular a study focused on comparing the performance of Linear, Polynomial, Traingular, and Wavelet functions in comparison to the Gaussian could reveal better interpolation methods for surface prediction.

2. One way to improve upon the integer-based Layer Strip-back parameter involves exploring the concept of optimizing a vector of real-values that assigned to each individual layer. These optimised fading factors would determine the degree to which each layer's weights are updated during training, offering several benefits. Fine-grained control over the learning process, enabling smooth transitions between frozen and fine-tuned layers, adapting to the intricacies of HAR tasks. Additionally, dynamic adaptation of fading factors during training could enhance convergence and enhance accuracy. These factors act as natural regularizers, curbing overfitting and improving generalization. Efficient resource utilization could be achieved by freezing layers with fading factors set to 0, removing the need for gradient calculations.

3. Creating a more flexible loss function that can produce a wide range of loss curves, allowing for customized punishment and reward for different machine learning problems. In particular the adaptation of the Cubic Bezier Curve function into a loss function would provide a range of curves through four configurable points that can cover all of the traditional classification loss function shapes and even regression loss curves such as the Mean Squared Error loss. By optimising these four points unique and optimal loss curves could be identified for various machine learning tasks including Semantic Segmentation

4. The use of the VPM in the NIS algorithm provides a way to embed the shape of the search space into a neural network. While this is beneficial to the search it only learns from the current search space. Ideally pretraining the model on various search spaces before hand would likely speed up the execution time of the algorithm and also provide much more robust predictions.

5. The VPM in NIS is used to predicts velocities within a more complex algorithm, it may be beneficial to investigate ways to directly predict the position of a particle. From initial investigations not recorded in this work suggest that this is not a trivial problem and would depend highly on how the prediction target of the position is defined.

6. One promising avenue for enhancing the CSO algorithm is to leverage the information derived from the reachability plot generated during OPTICS clustering to refine the algorithm's initialization process, random position reset or search targets. By utilizing the reachability plot, CSO can intelligently identify positions for particles to search by through areas of low reachability, which typically correspond to regions of low cluster density or near cluster boundaries. This approach would force the swarm toward areas of the solution space that require exploration. Future work can focus on developing robust techniques for this cluster-aware initialization, potentially leading to faster convergence and more efficient exploration of complex optimization landscapes.

7. A possibility for enhancing the CSO algorithm involves the incorporation of reinforcement learning techniques, such as Q-learning, to enable adaptive selection between diverse exploration behaviours beyond the cluster-based search. By leveraging Q-learning, the CSO algorithm can autonomously learn and adapt its decision-making process, dynamically de-

termining when to employ various exploration strategies based on the evolving optimization landscape. This approach extends the algorithm's adaptability to scenarios where alternative exploration mechanisms, distinct from clustering, may be beneficial.

8. Expanding beyond the current focus on block stacking and optimization centered on filters and hidden units, a promising avenue for future research lies in optimizing the actual block structure itself. The development of a more flexible model encoding structure, capable of dynamically selecting from diverse layer types (e.g., LSTM, CNN, pooling, fully connected, activation), represents a crucial step toward crafting adaptable deep learning models. This heightened flexibility empowers models to autonomously choose the most suitable layers and architectures for specific data patterns and complexities. By incorporating this capability within a block optimization strategy, models can systematically explore and refine the arrangement and composition of layers within blocks or segments, fine-tuning architecture for optimal performance. Such an approach holds great potential, particularly in tasks like emotion recognition, for augmenting the adaptability and effectiveness of deep learning models.

6.5 Research Application in the Proceeding 3-5 years

The culmination of this work has resulted in the development of three novel swarm optimization algorithms for optimizing deep learning methods across a variety of domains. These algorithms were designed in conjunction with novel deep learning approaches inspired by the existing literature in the respective fields of human action recognition, semantic segmentation, and audio emotion classification. These contributions represent significant advancements in the field by successfully automating hyperparameter and neural architecture searches with and without transfer learning, and incorporating multiple loss functions. These findings provide valuable insights and potential solutions for similar optimization problems in the future, particularly those with high levels of dimensionality and search space variability.

Through the course of this research, we have demonstrated the effectiveness of using swarm optimization algorithms to optimize deep learning models in various domains. We have also shown that these techniques can be tailored to specific problems and incorporate domain-specific consid-

erations, such as transfer learning and multiple loss functions. These contributions can contribute to a better understanding of how to optimize deep learning models for a variety of tasks and domains, and pave the way for further research in this area.

The development of novel swarm optimization algorithms for optimizing deep learning models across multiple domains has the potential to have significant impacts in both academia and industry. In academia, these contributions can advance the state-of-the-art in optimization techniques and deep learning methods, and stimulate further research in related areas. They can also provide new tools and techniques for researchers and practitioners to optimize and improve the performance of deep learning models for various tasks and applications.

In industry, the use of these optimization algorithms can facilitate the development of more effective and efficient deep learning systems for a wide range of applications, including image and video analysis, speech and language processing, medical imaging, robotics, and more. By automating the optimization of hyperparameters and neural architectures, these algorithms can reduce the time and resources required to design and train deep learning models, and potentially improve their performance and robustness. They can also enable the development of more adaptive and flexible deep learning systems that can adapt to changing requirements or data distributions.

Overall, the optimization of deep learning models using swarm algorithms has the potential to facilitate the widespread adoption and deployment of deep learning systems in various sectors and industries, and to enhance their impact and usefulness in solving real-world problems.

Acronyms

1DCNN	1 Dimensional Convolutional Neural Network vii, viii, 97, 103–106, 109, 110, 112–114, 125
ABC	Artificial Bee Colony 18
ADAM	Adaptive Moment Estimation 8, 109
AFS	Active Feature Selection 37
AGMOPSO	Adaptive Gradient Multi-Objective Particle Swarm Optimisation 22
ANN	Artificial Neural Network 20
BiLSTM	Bidirectional Long Short-Term Memory ii, v, vii, viii, 97, 103–106, 109, 110, 112, 113, 115, 116, 125
BPNN	Backpropagation Neural Network 20
CIFAR	Canadian Institute for Advanced Research 18, 20
CNN	Convolutional Neural Network i–vii, 1–3, 5–13, 15, 16, 18–20, 25, 27–32, 35, 37, 39–41, 47–58, 60–62, 68, 70, 72, 83, 85, 87, 88, 97, 103, 112, 113, 116, 117, 122, 123
CNN-BiLSTM	Convolutional Neural Network with Bidirectional Long Short Term Memory v, vii, viii, 97, 103, 105, 109–114, 116, 117, 125
CSO	Cluster Search Optimisation i–iii, v, viii, 3, 15, 97–99, 106, 109–111, 113–119, 125, 127
CSO_2	Competitive Swarm Optimiser 21

DB-SCAN Density-Based Spatial Clustering of Applications with Noise 36

DE Differential Evolution 18, 19, 63

DGA DB-SCAN for Genetic Algorithms 36

DMS-PSO Dynamic Multi-Swarm Particle Swarm Optimisation 21

DPC Density Peaks Clustering 22, 23

Dsal Discriminative spatial Saliency 27

EnvPSO Environmental Particle Swarm Optimisation i, iii–v, vii, 2, 12, 15, 39–42, 44, 47, 48, 50–64, 122, 123, 126

EPM Expanded Parts Model 26, 27, 59

EPUS-PSO Efficient Population Utilisation Strategy Particle Swarm Optimisation 21

FA Firefly Algorithm 7, 84–93, 109–111, 113, 114, 116, 118, 119, 124, 125

FCM Fuzzy C-Means Clustering 18, 19

FST-PSO Fuzzy Self-Tuning Particle Swarm Optimisation 18

G-FRNet Gated Feedback Refinement Network 32

GAN Generative Adversarial Network 1, 24

GIST Global Image Feature 4

GPT Generative Pre-trained Transformer 1

Grad-CAM Gradient-weighted Class Activation Mapping 28

GT Ground Truth 13, 32, 68, 71–73, 78–81, 85, 86, 88–90

HAR Human Action Recognition i, vii, 2–5, 12, 26–30, 39–41, 48, 51, 56, 59, 60, 63

HAT Human Attributes 26

HII Human Interaction Image 28

HOG Histogram of Oriented Gradients 4, 5

HSSO Hierarchical Sorting Swarm Optimiser 21

IGD Inverted Generational Distance 22

IoU Intersection over Union 32

JHS-SPN Joint learning Hierarchical Spatial Sum Product Network 27

LLC Locality-constrained Linear Coding 27, 29

LSTM Long Short-Term Memory vii, 13, 35, 68, 70, 72, 104, 106, 116

MAE Minimum Annotation Effort 26, 59

MAP Mean Average Precision vii, 26–28, 51–55, 57–61

mIoU Mean Intersection Over Union viii, 13, 33, 34, 82–87, 92

MLCC Multilevel Cooperative Coevolution 21

MLR Multiple Linear Regression 29

MNIST Differential Evolution 19

MPSO Multi-level Particle Swarm Optimisation 19

NCNN Non-sequential Convolutional Neural Network 30

NIS Neural Inference Search i–iii, v, viii, 2, 3, 13, 15, 68, 69, 75, 76, 78, 81–94, 123, 124

NNPM Neural Network with Pseudo Multilabel 36

NSGA-II Non-dominated Sorting Genetic Algorithm II 22

PCA Principal Component Analysis 34

PDPC Particle Swarm Optimisation with Density Peaks Clustering 22, 23

PSO Particle Swarm Optimisation vii, 6–8, 10, 11, 15–19, 22–25, 39, 41–43, 47, 51, 52, 54–58, 61, 63–65, 84–94, 97–99, 109–111, 113–116, 118, 119, 122–126

R-CNN Recurrent Convolutional Neural Network 12, 39, 40, 49, 54, 55, 59, 60, 62, 123

RGB Red, Blue, Green 3, 31, 81

RGB-D Red, Blue, Green, Depth 3

RMP Regularised Max Pooling 27

RMSProp Root Mean Square Propagation 8

SA Simulated Annealing 19

SDWPSO Sine Mapped Dynamic Weighted particle Swarm Optimisation 20

SGD Stochastic Gradient Decent 8, 82, 85

SIFT Scale-invariant Feature Transform 4, 5

SL-PSO Social Learning Particle Swarm Optimisation 21

SODBAE Swarm Optimised Dense Block Architecture Ensemble 18

SPEA2 Strength Pareto Evolutionary Algorithm 2 22

SPN Sum Product Network 27, 59

STCNN Spatio-Temporal Convolutional Neural Network 29

SVM Support Vector Machine 20, 26, 27, 36, 37, 116

VGG16 Visual Geometry Group 16 model 26, 29–31, 33, 59, 60

VGG19 Visual Geometry Group 19 model vi, 12, 30, 39, 40, 47–50, 54–56, 58–60, 62

VPM Velocity Prediction Model 69–74, 127

ZTD Zero-shot Tensor Decomposition 29

References

[1] G. Guo and A. Lai, "A survey on still image based human action recognition," *Pattern Recognition*, vol. 47, no. 10, pp. 3343–3361, 2014.

[2] Y. Zheng, Y.-J. Zhang, X. Li, and B.-D. Liu, "Action recognition in still images using a combination of human pose and context information," in *2012 19th IEEE International Conference on Image Processing*. IEEE, 2012, pp. 785–788.

[3] C. Thurau and V. Hlavác, "Pose primitive based human action recognition in videos or still images," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[4] C. Desai, D. Ramanan, and C. Fowlkes, "Discriminative models for static human-object interactions," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*. IEEE, 2010, pp. 9–16.

[5] N. Shapovalova, W. Gong, M. Pedersoli, F. X. Roca, and J. Gonzalez, "On importance of interactions and context in human action recognition," in *Iberian conference on pattern recognition and image analysis*. Springer, 2011, pp. 58–66.

[6] L.-J. Li and L. Fei-Fei, "What, where and who? classifying events by scene and object recognition," in *2007 IEEE 11th international conference on computer vision*. IEEE, 2007, pp. 1–8.

[7] A. Gupta, A. Kembhavi, and L. S. Davis, "Observing human-object interactions: Using spatial and functional compatibility for recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 10, pp. 1775–1789, 2009.

[8] S. Maji, L. Bourdev, and J. Malik, "Action recognition from a distributed representation of pose and appearance," in *CVPR 2011*. IEEE, 2011, pp. 3177–3184.

[9] C. Desai and D. Ramanan, "Detecting actions, poses, and objects with relational phraselets," in *European conference on computer vision*. Springer, 2012, pp. 158–172.

[10] V. Delaitre, J. Sivic, and I. Laptev, "Learning person-object interactions for action recognition in still images," *Advances in neural information processing systems*, vol. 24, pp. 1503–1511, 2011.

[11] F. Sener, C. Bas, and N. Ikizler-Cinbis, "On recognizing actions in still images via multiple features," in *European Conference on Computer Vision*. Springer, 2012, pp. 263–272.

[12] B. Yao and L. Fei-Fei, "Recognizing human-object interactions in still images by modeling the mutual context of objects and human poses," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 9, pp. 1691–1703, 2012.

[13] B. Yao, A. Khosla, and L. Fei-Fei, "Classifying actions and measuring action similarity by modeling the mutual context of objects and human poses," *a) A*, vol. 1, no. D2, p. D3, 2011.

[14] P. Li and J. Ma, "What is happening in a still picture?" in *The First Asian Conference on Pattern Recognition*. IEEE, 2011, pp. 32–36.

[15] D. T. Le, R. Bernardi, and J. Uijlings, "Exploiting language models to recognize unseen actions," in *Proceedings of the 3rd ACM conference on International conference on multimedia retrieval*, 2013, pp. 231–238.

[16] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. Guibas, and L. Fei-Fei, "Human action recognition by learning bases of action attributes and parts," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 1331–1338.

[17] V. Delaitre, I. Laptev, and J. Sivic, "Recognizing human actions in still images: a study of bag-of-features and part-based representations," in *BMVC 2010-21st British Machine Vision Conference*, 2010, pp. 1–11.

[18] H. A. Qazi, U. Jahangir, B. M. Yousuf, and A. Noor, "Human action recognition using sift and hog method," in *2017 International Conference on Information and Communication Technologies (ICICT)*. IEEE, 2017, pp. 6–10.

[19] G. Sharma, F. Jurie, and C. Schmid, "Expanded parts model for human attribute and action recognition in still images," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 652–659.

[20] P. M. Dhulavvagol and N. C. Kundur, "Human action detection and recognition using sift and svm," in *International Conference on Cognitive Computing and Information Processing*. Springer, 2017, pp. 475–491.

[21] B. Li, R. Xiao, Z. Li, R. Cai, B.-L. Lu, and L. Zhang, "Rank-sift: Learning to rank repeatable local interest points," in *CVPR 2011*. IEEE, 2011, pp. 1737–1744.

[22] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[23] P. Kreowsky and B. Stabernack, "A full-featured fpga-based pipelined architecture for sift extraction," *IEEE Access*, vol. 9, pp. 128 564–128 573, 2021.

[24] M. F. Aslan, A. Durdu, K. Sabanci, and M. A. Mutluer, "Cnn and hog based comparison study for complete occlusion handling in human tracking," *Measurement*, vol. 158, p. 107704, 2020.

[25] X. Wang, T. X. Han, and S. Yan, "An hog-lbp human detector with partial occlusion handling," in *2009 IEEE 12th international conference on computer vision*. IEEE, 2009, pp. 32–39.

[26] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent advances and trends in visual tracking: A review," *Neurocomputing*, vol. 74, no. 18, pp. 3823–3831, 2011.

[27] A. Oliva and A. Torralba, "Building the gist of a scene: The role of global image features in recognition," *Progress in brain research*, vol. 155, pp. 23–36, 2006.

[28] C. Siagian and L. Itti, "Rapid biologically-inspired scene classification using features shared with visual attention," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 2, pp. 300–312, 2007.

[29] B. Xie, J. Qin, X. Xiang, H. Li, and L. Pan, "An image retrieval algorithm based on gist and sift features." *Int. J. Netw. Secur.*, vol. 20, no. 4, pp. 609–616, 2018.

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

[31] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *arXiv preprint arXiv:1406.2199*, 2014.

[32] M. Al-Faris, J. Chiverton, D. Ndzi, and A. I. Ahmed, "A review on computer vision-based methods for human action recognition," *Journal of Imaging*, vol. 6, no. 6, p. 46, 2020.

[33] B. Degardin and H. Proença, "Human behavior analysis: A survey on action recognition," *Applied Sciences*, vol. 11, no. 18, p. 8324, 2021.

[34] H.-B. Zhang, Y.-X. Zhang, B. Zhong, Q. Lei, L. Yang, J.-X. Du, and D.-S. Chen, "A comprehensive survey of vision-based human action recognition methods," *Sensors*, vol. 19, no. 5, p. 1005, 2019.

[35] G. Yao, T. Lei, and J. Zhong, "A review of convolutional-neural-network-based action recognition," *Pattern Recognition Letters*, vol. 118, pp. 14–22, 2019.

[36] Y. Kong and Y. Fu, "Human action recognition and prediction: A survey," *arXiv preprint arXiv:1806.11230*, 2018.

[37] L. Zhang, K. Mistry, S. C. Neoh, and C. P. Lim, "Intelligent facial emotion recognition using moth-firefly optimization," *Knowledge-Based Systems*, vol. 111, pp. 248–267, 2016.

[38] T. Lawrence, L. Zhang, K. Rogage, and C. P. Lim, "Evolving deep architecture generation with residual connections for image classification using particle swarm optimization," *Sensors*, vol. 21, no. 23, p. 7936, 2021.

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

[40] T. Y. Tan, L. Zhang, C. P. Lim, B. Fielding, Y. Yu, and E. Anderson, "Evolving ensemble models for image segmentation using enhanced particle swarm optimization," *IEEE access*, vol. 7, pp. 34 004–34 019, 2019.

[41] T. Fan, G. Wang, Y. Li, and H. Wang, "Ma-net: A multi-scale attention network for liver and tumor segmentation," *IEEE Access*, vol. 8, pp. 179 656–179 665, 2020.

[42] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[43] Q. Jin, C. Li, S. Chen, and H. Wu, "Speech emotion recognition with acoustic and lexical features," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 4749–4753.

[44] C. S. Ooi, K. P. Seng, L.-M. Ang, and L. W. Chew, "A new approach of audio emotion recognition," *Expert systems with applications*, vol. 41, no. 13, pp. 5858–5869, 2014.

[45] A. Huq, J. P. Bello, and R. Rowe, "Automated music emotion recognition: A systematic evaluation," *Journal of New Music Research*, vol. 39, no. 3, pp. 227–244, 2010.

[46] B. C. Ko, "A brief review of facial emotion recognition based on visual information," *sensors*, vol. 18, no. 2, p. 401, 2018.

[47] S. Ebrahimi Kahou, V. Michalski, K. Konda, R. Memisevic, and C. Pal, "Recurrent neural networks for emotion recognition in video," in *Proceedings of the 2015 ACM on international conference on multimodal interaction*, 2015, pp. 467–474.

[48] D. Bertero, F. B. Siddique, C.-S. Wu, Y. Wan, R. H. Y. Chan, and P. Fung, "Real-time speech emotion and sentiment recognition for interactive dialogue systems," in *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016, pp. 1042–1047.

[49] S. Slade, L. Zhang, Y. Yu, and C. P. Lim, "An evolving ensemble model of multi-stream convolutional neural networks for human action recognition in still images," *Neural Computing and Applications*, vol. 34, no. 11, pp. 9205–9231, 2022.

[50] S. Slade, L. Zhang, H. Huang, H. Asadi, C. P. Lim, Y. Yu, D. Zhao, H. Lin, and R. Gao, "Neural inference search for multiloss segmentation models," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[51] Y. Wang, H. Zhang, and G. Zhang, "cpso-cnn: An efficient pso-based algorithm for fine-

tuning hyper-parameters of convolutional neural networks," *Swarm and Evolutionary Computation*, vol. 49, pp. 114–123, 2019.

[52] G. L. F. da Silva, T. L. A. Valente, A. C. Silva, A. C. de Paiva, and M. Gattass, "Convolutional neural network-based pso for lung nodule false positive reduction on ct images," *Computer methods and programs in biomedicine*, vol. 162, pp. 109–118, 2018.

[53] F. C. Soon, H. Y. Khaw, J. H. Chuah, and J. Kanesan, "Hyper-parameters optimisation of deep cnn architecture for vehicle logo recognition," *IET Intelligent Transport Systems*, vol. 12, no. 8, pp. 939–946, 2018.

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

[55] B. Fielding and L. Zhang, "Evolving image classification architectures with enhanced particle swarm optimisation," *IEEE Access*, vol. 6, pp. 68 560–68 575, 2018.

[56] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.

[57] K. Mistry, L. Zhang, S. C. Neoh, C. P. Lim, and B. Fielding, "A micro-ga embedded pso feature selection approach to intelligent facial emotion recognition," *IEEE transactions on cybernetics*, vol. 47, no. 6, pp. 1496–1509, 2016.

[58] B. Fielding and L. Zhang, "Evolving deep denseblock architecture ensembles for image classification," *Electronics*, vol. 9, no. 11, p. 1880, 2020.

[59] M. S. Nobile, P. Cazzaniga, D. Besozzi, R. Colombo, G. Mauri, and G. Pasi, "Fuzzy self-tuning pso: A settings-free algorithm for global optimization," *Swarm and evolutionary computation*, vol. 39, pp. 70–85, 2018.

[60] P. Singh, S. Chaudhury, and B. K. Panigrahi, "Hybrid mpso-cnn: Multi-level particle swarm optimized hyperparameters of convolutional neural network," *Swarm and Evolutionary Computation*, vol. 63, p. 100863, 2021.

[61] B. Bai, J. Zhang, X. Wu, G. wei Zhu, and X. Li, "Reliability prediction-based improved

dynamic weight particle swarm optimization and back propagation neural network in engineering systems," *Expert Systems with Applications*, vol. 177, p. 114952, 2021.

[62] R. Lan, L. Zhang, Z. Tang, Z. Liu, and X. Luo, "A hierarchical sorting swarm optimizer for large-scale optimization," *IEEE Access*, vol. 7, pp. 40 625–40 635, 2019.

[63] H. Han, W. Lu, L. Zhang, and J. Qiao, "Adaptive gradient multiobjective particle swarm optimization," *IEEE transactions on cybernetics*, vol. 48, no. 11, pp. 3067–3079, 2017.

[64] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary computation*, vol. 8, no. 2, pp. 173–195, 2000.

[65] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary multiobjective optimization*. London: Springer, 2005, pp. 105–145.

[66] J. Cai, H. Wei, H. Yang, and X. Zhao, "A novel clustering algorithm based on dpc and pso," *IEEE Access*, vol. 8, pp. 88 200–88 214, 2020.

[67] P. Hu, J.-S. Pan, S.-C. Chu, and C. Sun, "Multi-surrogate assisted binary particle swarm optimization algorithm and its application for feature selection," *Applied Soft Computing*, vol. 121, p. 108736, 2022.

[68] X. Zhang, S. Xia, X. Li, and T. Zhang, "Multi-objective particle swarm optimization with multi-mode collaboration based on reinforcement learning for path planning of unmanned air vehicles," *Knowledge-Based Systems*, p. 109075, 2022.

[69] R. Vaze, N. Deshmukh, R. Kumar, and A. Saxena, "Development and application of quantum entanglement inspired particle swarm optimization," *Knowledge-Based Systems*, vol. 219, p. 106859, 2021.

[70] L. Zhang and L. Zhao, "High-quality face image generation using particle swarm optimization-based generative adversarial networks," *Future Generation Computer Systems*, vol. 122, pp. 98–104, 2021.

[71] G. Sharma, F. Jurie, and C. Schmid, "Expanded parts model for semantic description of humans in still images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 1, pp. 87–101, 2016.

[72] Y. Zhang, L. Cheng, J. Wu, J. Cai, M. N. Do, and J. Lu, "Action recognition in still images with minimum annotation efforts," *IEEE Transactions on Image Processing*, vol. 25, no. 11, pp. 5479–5490, 2016.

[73] J. Wang and G. Wang, "Hierarchical spatial sum–product networks for action recognition in still images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 1, pp. 90–100, 2016.

[74] J. Li, Y. Wong, Q. Zhao, and M. S. Kankanhalli, "Attention transfer from web images for video recognition," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 1–9.

[75] M. Safaei, "Action recognition in still images: Confluence of multilinear methods and deep learning methods and deep learning," Ph.D. dissertation, University of Central Florida, 2020.

[76] X. Yu, Z. Zhang, L. Wu, W. Pang, H. Chen, Z. Yu, and B. Li, "Deep ensemble learning for human action recognition in still images," *Complexity*, vol. 2020, pp. 1–23, 2020, article ID 9428612.

[77] L. Liu, R. T. Tan, and S. You, "Loss guided activation for action recognition in still images," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 152–167.

[78] S. Yan, J. S. Smith, W. Lu, and B. Zhang, "Multibranch attention networks for action recognition in still images," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 4, pp. 1116–1125, 2017.

[79] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[80] Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2018, pp. 3–11.

[81] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang,

V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.

[82] Y. Zhang, O. Morel, M. Blanchon, R. Seulin, M. Rastgoo, and D. Sidibé, "Exploration of deep learning-based multimodal fusion for semantic road scene segmentation." in *VISIGRAPP (5: VISAPP)*, 2019, pp. 336–343.

[83] D. Saire and A. R. Rivera, "Empirical study of multi-task hourglass model for semantic segmentation task," *IEEE Access*, vol. 9, pp. 80 654–80 670, 2021.

[84] M. Amirul Islam, M. Rochan, N. D. Bruce, and Y. Wang, "Gated feedback refinement network for dense image labeling," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3751–3759.

[85] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2017, pp. 11–19.

[86] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[87] M. Abdullah, M. M. Fraz, and S. A. Barman, "Localization and segmentation of optic disc in retinal images using circular hough transform and grow-cut algorithm," *PeerJ*, vol. 4, p. e2003, 2016.

[88] S. Morales, V. Naranjo, J. Angulo, and M. Alcañiz, "Automatic detection of optic disc based on pca and mathematical morphology," *IEEE transactions on medical imaging*, vol. 32, no. 4, pp. 786–796, 2013.

[89] J. Parry, D. Palaz, G. Clarke, P. Lecomte, R. Mead, M. Berger, and G. Hofer, "Analysis of deep learning architectures for cross-corpus speech emotion recognition." in *Interspeech*, 2019, pp. 1656–1660.

[90] J. Li, N. Yan, and L. Wang, "Unsupervised cross-lingual speech emotion recognition using pseudo multilabel," in *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2021, pp. 366–373.

[91] S. Kanwal and S. Asghar, "Speech emotion recognition using clustering based ga-optimized feature set," *IEEE Access*, vol. 9, pp. 125 830–125 842, 2021.

[92] T. Hussain, W. Wang, N. Bouaynaya, H. Fathallah-Shayk, and L. Mihaylova, "Deep learning for audio visual emotion recognition," in *Proceedings of the 25th International Conference on Information Fusion (Fusion 2022)*. Institute of Electrical and Electronics Engineers, 2022.

[93] F. Haider, S. Pollak, P. Albert, and S. Luz, "Emotion recognition in low-resource settings: An evaluation of automatic feature selection methods," *Computer Speech & Language*, vol. 65, p. 101119, 2021.

[94] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[95] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[96] S. Ma, S. A. Bargal, J. Zhang, L. Sigal, and S. Sclaroff, "Do less and achieve more: Training cnns for action recognition utilizing action images from the web," *Pattern Recognition*, vol. 68, pp. 334–345, 2017.

[97] M. Safaei and H. Foroosh, "Still image action recognition by predicting spatial-temporal pixel evolution," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 111–120.

[98] G. Sharma, F. Jurie, and C. Schmid, "Discriminative spatial saliency for image classification," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3506–3513.

[99] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 2169–2178.

[100] M. Safaei, P. Balouchian, and H. Foroosh, "Ucf-star: A large scale still image dataset for understanding human actions," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 03, 2020, pp. 2677–2684.

[101] A. S. A. Alraimi, "Development of new models for vision-based human activity recognition," Ph.D. dissertation, Universitat Rovira i Virgili, 2019.

[102] D. R. Nayak, R. Dash, and B. Majhi, "Discrete ripplet-ii transform and modified pso based improved evolutionary extreme learning machine for pathological brain detection," *Neurocomputing*, vol. 282, pp. 232–247, 2018.

[103] A. R. Jordehi, "Enhanced leader pso (elpso): A new pso variant for solving global optimisation problems," *Applied Soft Computing*, vol. 26, pp. 401–417, 2015.

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

[105] Q. Chen, Y. Chen, and W. Jiang, "Genetic particle swarm optimization–based feature selection for very-high-resolution remotely sensed imagery object change detection," *Sensors*, vol. 16, no. 8, p. 1204, 2016.

[106] S. Mirjalili, "Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Computing and Applications*, vol. 27, no. 4, pp. 1053–1073, 2016.

[107] S. Mirjalili, "The ant lion optimizer," *Advances in engineering software*, vol. 83, pp. 80–98, 2015.

[108] A. Chaurasia and E. Culurciello, "Linknet: Exploiting encoder representations for efficient semantic segmentation," in *2017 IEEE Visual Communications and Image Processing (VCIP)*. IEEE, 2017, pp. 1–4.

[109] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2881–2890.

[110] J. H. Kumar, A. K. Pediredla, and C. S. Seelamantula, "Active discs for automated optic disc segmentation," in *2015 IEEE global conference on signal and information processing (GlobalSIP)*. IEEE, 2015, pp. 225–229.

[111] Z. U. Rehman, S. S. Naqvi, T. M. Khan, M. Arsalan, M. A. Khan, and M. Khalil, "Multi-

parametric optic disc segmentation using superpixel based feature classification," *Expert Systems with Applications*, vol. 120, pp. 461–473, 2019.

[112] M. N. Zahoor and M. M. Fraz, "A correction to the article "fast optic disc segmentation in retina using polar transform"," *IEEE Access*, vol. 6, pp. 4845–4849, 2018.

[113] Z. Fan, Y. Rong, X. Cai, J. Lu, W. Li, H. Lin, and X. Chen, "Optic disk detection in fundus image based on structured learning," *IEEE journal of biomedical and health informatics*, vol. 22, no. 1, pp. 224–234, 2017.

[114] A. Kundu, V. Vineet, and V. Koltun, "Feature space optimization for semantic video segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3168–3175.

[115] M. A. Islam, S. Naha, M. Rochan, N. Bruce, and Y. Wang, "Label refinement network for coarse-to-fine semantic segmentation," *arXiv preprint arXiv:1703.00551*, 2017.

[116] G. L. Oliveira, W. Burgard, and T. Brox, "Dpdb-net: exploiting dense connections for convolutional encoders," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4525–4531.

[117] X.-S. Yang and X. He, "Firefly algorithm: recent advances and applications," *arXiv preprint arXiv:1308.3898*, 2013.

[118] Z. B. Zabinsky *et al.*, "Random search algorithms," *Department of Industrial and Systems Engineering, University of Washington, USA*, 2009.

[119] F. Neri and C. Cotta, "Memetic algorithms and memetic computing optimization: A literature review," *Swarm and Evolutionary Computation*, vol. 2, pp. 1–14, 2012.

[120] E. H. Houssein, A. G. Gad, and Y. M. Wazery, "Jaya algorithm and applications: A comprehensive review," *Metaheuristics and Optimization in Computer and Electrical Engineering*, pp. 3–24, 2021.

[121] M. Khajehzadeh and M. Eslami, "Gravitational search algorithm for optimization of retaining structures," *Indian Journal of Science and Technology*, vol. 5, no. 1, pp. 1821–1827, 2012.

[122] S. Sivanandam and S. Deepa, "Genetic algorithm optimization problems," in *Introduction to genetic algorithms*. Springer, 2008, pp. 165–209.

[123] X.-S. Yang, "Flower pollination algorithm for global optimization," in *International conference on unconventional computing and natural computation*. Springer, 2012, pp. 240–249.

[124] M. Mareli and B. Twala, "An adaptive cuckoo search algorithm for optimisation," *Applied computing and informatics*, vol. 14, no. 2, pp. 107–115, 2018.

[125] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The arithmetic optimization algorithm," *Computer methods in applied mechanics and engineering*, vol. 376, p. 113609, 2021.

[126] S. Andradóttir and A. A. Prudius, "Adaptive random search for continuous simulation optimization," *Naval Research Logistics (NRL)*, vol. 57, no. 6, pp. 583–604, 2010.

[127] S. Mirjalili, A. Lewis, and A. S. Sadiq, "Autonomous particles groups for particle swarm optimization," *Arabian Journal for Science and Engineering*, vol. 39, no. 6, pp. 4683–4697, 2014.

[128] E. Schubert and M. Gertz, "Improving the cluster structure extracted from optics plots," in *LWDA*, 2018.

[129] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *ArXiv e-prints*, mar 2016.

[130] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[131] H. M. Fayek, M. Lech, and L. Cavedon, "Towards real-time speech emotion recognition using deep neural networks," in *2015 9th international conference on signal processing and communication systems (ICSPCS)*. IEEE, 2015, pp. 1–5.

[132] K. Venkataramanan and H. R. Rajamohan, "Emotion recognition from speech," *arXiv preprint arXiv:1912.10458*, 2019.