T.C
ERCIYES UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
DEPARTMENT OF MECHATRONICS ENGINEERING

NOVEL DROWSINESS DETECTION MODELS BY HYBRID OPTIMIZATION ALGORITHMS AND MACHINE LEARNING TECHNIQUES

Prepared by
RADHWAN ALI ABDULGHANI SALEH

Supervisor
Asst. Prof. RÜŞTÜ AKAY

MS Thesis

July 2020
KAYSERI

T.C
ERCIYES UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
DEPARTMENT OF MECHATRONICS ENGINEERING

NOVEL DROWSINESS DETECTION MODELS BY HYBRID OPTIMIZATION ALGORITHMS AND MACHINE LEARNING TECHNIQUES

(MS Thesis)

Prepared by
RADHWAN ALI ABDULGHANI SALEH

Supervisor
Asst. Prof. RÜŞTÜ AKAY

July 2020
KAYSERI

SCIENTIFIC ETHICS SUITABILITY

I, hereby, declare that all information in this work was obtained in accordance with the academic and ethical rules. All results and materials that not been at the essence of this work are also transferred and expressed by giving reference as required by these rules and behaviour.

RADHWAN ALI ABDULGHANI SALEH



SUITABILITY FOR GUIDE

The MS thesis entitled **"NOVEL DROWSINESS DETECTION MODELS BY HYBRID OPTIMIZATION ALGORITHMS AND MACHINE LEARNING TECHNIQUES"** has been prepared in accordance with Erciyes University Graduate Education and Teaching Institute Thesis Preparation and Writing Guide.

Student

RADHWAN ALI ABDULGHANI SALEH

Supervisor

Asst. Prof. RÜŞTÜ AKAY

Head of Mechatronics Engineering Department

Prof. Dr. Şahin YILDIRIM

ACCEPTANCE AND APPROVAL PAGE

This study entitled **"NOVEL DROWSINESS DETECTION MODELS BY HYBRID OPTIMIZATION ALGORITHMS AND MACHINE LEARNING TECHNIQUES"** prepared by **"Radhwan Ali Abdulghani Saleh"** under the supervision of **Asst. Prof. RÜŞTÜ AKAY** was accepted by the jury as MSc. Thesis in Mechanical Engineering department.

…….. /…… / 2020

JURY:

Supervisor : Asst. Prof. RÜŞTÜ AKAY

Juror : Assoc. Prof. Muzaffer KANAAN

Juror : Asst. Prof. Mehmet BİLİM

APPROVAL

That the acceptance of this thesis has been approved by the decision of the Institute's Board of Directors with the ……...……. date and ……. numbered decision.

……. /……../ ……

Prof. Dr. Mehmet AKKURT

Director of the Institute

ACKNOWLEDGEMENTS

First and foremost, I would like to thank my supervisor Asst. Prof. RÜŞTÜ AKAY heartily for his unwavering support and helpful guidance throughout my master's degree. I deem myself fortunate to work under his supervision. I thank him sincerely for his advice, encouragement, and patience.

I would like also to thank my dear friend Shawqi FAREA, my wife Basma AL-ABSSEE and my brother Nagmy SALEH for their great support during the thesis work and preparation.

I would like to thank Türkiye Scholarships for giving me this opportunity to complete my master's degree in this great country: Turkey.

Last but not least, I would like to send my special thanks and regards to my great family, in particular my parents, for their endless support and help.

Radhwan Ali Abdulghani Saleh

KAYSERI, 2020

NOVEL DROWSINESS DETECTION MODELS BY HYBRID OPTIMIZATION ALGORITHMS AND MACHINE LEARNING TECHNIQUES

RADHWAN ALI ABDULGHANI SALEH
Erciyes University, Graduate School Of Natural And Applied Sciences
MS Thesis, July 2020

Supervisor: Asst. Prof. RÜŞTÜ AKAY

ABSTRACT

Drowsiness or tiredness is a significant cause of road accidents and has a huge effect on road safety. A lot of fatal accidents can be avoided by warning drowsy drivers on time. There are various methods for detecting drowsiness which tracks the driver's sleepiness while driving and warn the drivers if they do not concentrate on driving. In this study, Image-based approaches are developed using novel hybrid optimization algorithms and machine learning techniques. Indeed, to guarantee the success of these approaches, we have to focus on four things simultaneously: the generality of the dataset, the features extracted from the dataset, the complexity of the machine learning method, and the optimization algorithm used to optimize the parameters of this method. In this study, besides the novel features that are extracted from the images, novel optimization algorithms are developed then used to train the multi-layer neural network models for the purpose of yawning detection and eye state detection. Moreover, the fuzzy inference model is proposed for drowsiness level detection. As a result of creating novel optimization algorithm, more accurate models are developed in order to detect the drowsy of the driver automatically. Thus, the results of this study show that the proposed models are competitive models with high accuracies. And the obtained accuracy of the proposed model is 77.19%.

Keywords: Drowsiness level detection, Yawning detection, Eye state detection, MLP, Meta-heuristic algorithms, FIS

FARKLI HİBRİDİZASYON ALGORİTMALARI KULLANILARAK SÜRÜCÜ UYUŞUKLUK TESPİT SİSTEMİ GELİŞTİRİLMESİ

RADHWAN ALI ABDULGHANI SALEH
Erciyes Üniversitesi, Fen Bilimleri Enstitüsü
Yüksek Lisans Tezi, Temmuz 2020
Tez Danışmanı: Dr. Öğr. Üyesi RÜŞTÜ AKAY

ÖZET

Uyuklama veya yorgunluk, trafik kazalarının önemli bir nedenidir ve yol güvenliği üzerinde büyük bir etkiye sahiptir. Uykulu sürücüleri zamanında uyararak çok sayıda ölümcül kaza önlenebilir. Uyuklamayı tespit etmek için sürüş sırasında sürücünün uyuklama durumunu takip eden ve sürüşe konsantre olmadıkları takdirde sürücüleri uyaran çeşitli yöntemler vardır. Bu yöntemlerin başarısı veri kümesinin genelliği, veri kümesinden çıkarılan özellikler, makine öğrenme yönteminin karmaşıklığı ve kullanılan optimizasyon algoritmasına göre değişmektedir. Bu çalışmadasürücülerin uyuklama durumunu tespit etmek amacıyla hibrid optimizasyon algoritmaları ve makine öğrenme teknikleri kullanılarak görüntü tabanlı yaklaşımlar geliştirilmiştir. Geliştirilen modelde, esneme tespiti ve göz durumu tespiti amacıyla çok katmanlı sinir ağ modellerini eğitmek için yeni hibrid optimizasyon algoritmaları önerilmiş ve kullanılmıştır. Ayrıca, uyuklama seviyesi tespiti için bulanık çıkarım modelinden de faydalanılmıştır. Sonuçlar, önerilen bu yeni yaklaşımların %77.19 gibi yüksek bir doğrulukla rekabetçi bir model olduğunu göstermektedir.

Anahtar Kelimeler: Uyuklama tespiti, çok katmanlı algılayıcı, metasezgisel algoritmalar, bulanık çıkarım sistemi

TABLE OF CONTENTS

NOVEL DROWSINESS DETECTION MODELS BY HYBRID OPTIMIZATION ALGORITHMS AND MACHINE LEARNING TECHNIQUES

SCIENTIFIC ETHICS SUITABILITY ..... **i**

SUITABILITY FOR GUIDE ..... **ii**

ACCEPTANCE AND APPROVAL PAGE ..... **iii**

ACKNOWLEDGEMENTS ..... **iv**

ABSTRACT ..... **v**

ÖZET ..... **vi**

TABLE OF CONTENTS ..... **vii**

LIST OF ABBREVIATIONS ..... **ix**

LIST OF TABLES ..... **x**

LIST OF FIGURES ..... **xi**

INTRODUCTION **1**

CHAPTER 1

GENERAL REVIEW

1.1 LITERATURE REVIEW 3

1.2 OBJECTIVES AND AIMS 6

1.2.1 Overall Objective 6

1.2.2 Specific Aims 6

1.3 RESEARCH CONTRIBUTION 7

1.4 SCOPE OF STUDY 7

1.5 METHODOLOGY 7

1.6 OUTLINE 8

CHAPTER 2

BACKGROUND METHODS

2.1 VIOLA AND JONES METHOD 9

2.2 FEATURE EXTRACTION 10

2.3 OPTIMIZATION ALGORITHMS 11

2.3.1 Overview 11
2.3.2 The original ABC 13
2.3.3 The other versions of ABC 15
2.3.4 The original version of TLABC 17
2.3.5 The original version of ABC with Depth-first-Search (DFSABC_elite) .. 18
2.4 ARTIFICIAL NEURAL NETWORK 20
2.5 FUZZY INFERENCE SYSTEM 23

CHAPTER 3

PROPOSED METHODS

3.1 OVERVIEW 26
3.2 PROPOSED CLASSIFICATION MODELS 27
3.3 PROPOSED FUZZY MODEL 33

CHAPTER 4

EXPERIMENTS AND RESULTS

4.1 VALIDITY 37
4.2 DATASET DESCRIPTION 42
4.3 YAWNING DETECTION RESULTS 47
4.4 RIGHT EYE STATE DETECTION RESULTS 51
4.5 LEFT EYE STATE DETECTION RESULTS 54
4.6 DROWSINESS DETECTION RESULTS 57
4.7 STATISTICAL RESULTS 58

CHAPTER 5

CONCLUSION AND FUTURE STUDIES

5.1 SUMMARY AND DISCUSSION 61
5.2 FUTURE STUDY 62

REFERENCES 63

CURRICULUM VITAE 71

LIST OF ABBREVIATIONS

Acronym	Full Name
ABC	Artificial Bee Colony
ANFIS	Adaptive Neuro-Fuzzy Inference System
ANN	Artificial Neural Network
ECR	Eye Closure Rate
CERT	Computer Expression Recognition Toolbox
DDD	Deep Drowsiness Detection
DFSABC	ABC With Depth-First Search Framework
DL	Drowsiness Level
FIS	Fuzzy Inference System
GLCM	Gray Level Co-occurrence matrix
KSS	Karolinska Sleepiness Scale
MABC	Modified Artificial Bee Colony
MDFSABC	Modified Depth-First-Search Artificial Bee Colony
ML	Machine Learning
MLP	Multi-Layer Perceptron
MSE	Mean Square Error
MTLABC	Modified Teaching Learning Based Artificial Bee Colony
PERCLOS	Eye Closure Percentage
STD	Standard Deviation
SVD	Singular Value Decomposition
SVMs	Support Vector Machines
TLABC	Teaching-Learning-Based ABC
YD	Yawning Duration

LIST OF TABLES

Table 3.1 Details of inputs and output variables of FIS 33
Table 3.2 The fuzzy rules used in the proposed model. 35
Table 4.1 Comparison of average error in objective function value for 30 dimension of CEC 2014 test problems 38
Table 4.2 Comparison of the number of success of MABC against the other algorithms on the CEC 2014 benchmark 39
Table 4.3 Classification datasets and MLP structure 40
Table 4.4 Classification benchmarks results 41
Table 4.5 Summary of 22 features extracted from GLCM 45
Table 4.6 MSE and accuracy of yawning detection models 49
Table 4.7 Precision, Recall, and F1 Score of yawning detection models 50
Table 4.8 MSE and accuracy of right eye detection models 52
Table 4.9 Precision, Recall, and F1 Score of right eye detection models 53
Table 4.10 MSE and accuracy of left eye detection models 55
Table 4.11 Precision, Recall, and F1 Score of left eye detection models 56
Table 4.12 The accuracy of mouth and eyes detection 57
Table 4.13 Kruskal-Wallis Test Results of The Yawning, Left Eye, and Right Eye datasets 59
Table 4.14 Wilcoxon Sum Rank Test Results of The Three Dataset 60

LIST OF FIGURES

Figure 1.1 Block Diagram of the Proposed Model 8
Figure 2.1 Cascade mechanism [MATLAB Document] 10
Figure 2.2 The spatial connections between pixels [28] 11
Figure 2.3 Mechanism of Artificial Neural Network [72] 20
Figure 2.4 Multi-Layer Neural Network Architecture [72] 21
Figure 2.5 (a) Log-sigmoid function (b) Tan-sigmoid function 22
Figure 2.6 Structure of the Fuzzy Inference System 24
Figure 3.1 Multi-Layer Neural Network Training Process 28
Figure 3.2 The Proposed FIS Model 34
Figure 3.3 The Membership Functions of the three Inputs and the one Output 34
Figure 3.4 Rule Viewer of The Proposed Fuzzy Model 36
Figure 4.1 Examples of YawDD datase. 40
Figure 4.2 Convergence curves for yawning detection models. 46
Figure 4.3 Effect of threshold on accuracy of yawning detection models 48
Figure 4.4 Effect of threshold on F1 score of yawning detection models. 48
Figure 4.5 Convergence curves for right eye detection models 50
Figure 4.6 Effect of threshold on accuracy of right eye detection models 50
Figure 4.7 Effect of threshold on F1 score of right eye detection models 51
Figure 4.8 Convergence curves for left eye detection models 52
Figure 4.9 Effect of threshold on accuracy of left eye detection models 53
Figure 4.10 Effect of threshold on F1 score of left eye detection models 55

INTRODUCTION

Due to the increasing problem of traffic accidents around the world, many people deaths annually (at the rate of 1.25 million people) and between 20-50 million others exposures for non-fatal injuries and a lot of them with a disability as a result. However, one of the most important causes of these incidents is the dispersion or drowsiness of driver, which led us to the development of an intelligent system capable of detecting the drowsiness of the driver, which contributes to Decay the problem of traffic accidents at the lowest possible cost [1].

According to the US National Highway Traffic Safety Administration estimates, approximately 100000 traffic accidents and more than 1,500 deaths take place as a result of driver drowsiness and fatigue each year [2, 3]. In the U.S., at least 1500 persons die, and 40000 persons are injured in drowsy driver crashes each year, according to federal government accumulated data. These numbers are most likely an underestimation. Unless someone witnesses or survives the car accident and can testify to the driver's condition, it is challenging to identify driver drowsiness as it’s cause [3]. Thus, most deadly accidents can be avoided by integrating automatic fatigue warning systems into vehicles. Consequently, researchers have built different approaches for detecting driver drowsiness, and each one of them has its pros and cons.

Drowsiness detection approaches are typically divided into three major groups: Behavioural parameter-based techniques, Vehicular parameters-based techniques, and Physiological parameters-based techniques. Behavioural parameter-based techniques measure drivers’ fatigue through behavioral parameters of drivers such as yawning, facial expressions, facial expressions, eye closure ratio, head position, and eye blinking. Vehicular parameter-based techniques try to detect drowsiness based on vehicular features such as vehicle speed variability, frequent lane changing patterns, and steering wheel angle. Physiological parameters-based techniques detect drowsiness based on physical conditions such as heart rate, pulsation rate, and respiratory rate [4].

Techniques based on physiological parameters give accurate results, but their intrusive nature makes them unpractical techniques. On the other side, vehicular parameters-based techniques rely on weather and road geometry, which sometimes makes them unreliable. Although behavioral parameters-based techniques are influenced by lighting effects that decrease their accuracy compared with physiological parameters-based techniques, they are more reliable and easily implementable [5, 6]. Accordingly, in this study, we will focus on developing behavioral parameters-based techniques with better accuracies.

In literature, many approaches of drowsiness detection are based on behavioral parameters. But the Eye Closure Percentage (PERCLOS), which is the percentage of eyelid closure over the pupil over time, is among the most popular methods based on eye state observation. Moreover, this metric is often known as the key to driver fatigue [7, 8]. However, Caponecchia and Williamson did not find the effects of drowsiness in PERCLOS [9]. Moreover, the accuracy of these methods is negatively influenced by light, driver's glasses, and other factors. These results make PERCLOS not sufficient to build reliable and trustable drowsiness detection and implement it in vehicles.

Yawning-based detection systems examine changes in the geometric mouth shape of a drowsy person, such as broader mouth opening and lip location. Many researchers have identified yawning to be correlated with drowsiness [10]. But, these approaches can easily be misled if the driver is just talking instead of yawning [11].

Head position detection and gaze direction detection are other parameters that have been used in literature to detect fatigue, but they have poor reliability [11]. Therefore, the identification of drowsiness by continuous collection and analysis of the driver’s eyes and mouth features in real-time is an efficient process.

In this research, novel drowsiness detection models based on PERCLOS and yawning features are proposed. The contribution of this study is concluded in two things: creating a novel drowsiness detection model which is more reliable, and creating at least one novel optimization algorithm.

CHAPTER 1
GENERAL REVIEW

1.1 Literature review

As mentioned above, the behavioral parameters-based techniques are not offensive and do not make any disturbance to the driver; that is the reason these techniques are progressively better. In this section, we will discuss these techniques in detail.

As drivers start to suffer from drowsiness, physical changes occur in their faces. These are the behavioral changes that are to be observed for drowsiness detection. Yawning, head nodding, and eyelid state variations are the most common behavioral changes. To detect these behavioral changes, both image processing and machine learning techniques are integrated [4]. Various models have been introduced to identify the changes of either of these features.

In [12] the author proposed a model that used the head pose and eye state tracking to detect the driver's state. In this model, three features are used to identify whether the driver is drowsy or not. Although behavioral models are commonly used in many research, these models are generally affected by individual factors. In [13], the authors try to solve this problem by trying to use different thresholds according to the drivers' characteristics. The ambient lighting, head rotation, and the costs and performance of the cameras used in the acquisition are other issues related to behavioral approaches[14].

Some authors used in their models' meaningful features that are extracted from the driver’s eye over a fixed period. Then these features are used to detect the drowsiness state. Such features include PERCLOS, blink duration, and speed of opening and closing of the eyelid [4, 15]. Much work was done on the driver's eye status, and it is known to be one of the most accurate drowsiness detection models. Therefore, a lot of comparisons are made against it. Thus, in [16], a comparative study between ECG

(heartburn signal) and the closure of the eye found that the closure of the eye is more effective than ECG when assessing driver drowsiness. But it should be noted that when the driver is wearing lenses in front of illumination or changes head position, the accuracy of the monitoring of visual conditions degrades.

In some models, another behavioral approach was proposed. Such models detect if the driver's head is like a drowsy person's head [17]. In these models, a 3D camera or 3D software is required. However, if nodding is hugely likely to accompany drowsiness, this does not necessarily happen because drowsiness does not occur without nodding, or perhaps nodding only happens after slumber has been profoundly infused [18]. In both cases, it is obvious that the model efficiency will be reduced in terms of both accuracy and response time. Nevertheless, this issue was understood by scientists, and this kind of model generally is not used alone [12, 19].

Matsuo and Khiat, who work for NISSAN, isolated the driver's condition into four classifications: typical (alert), somewhat tired, strongly sluggish, and drowsy [20]. They utilized eye closure rate (ECR), head influence, and auxiliary practices to identify the driver's condition. IR based frameworks function admirably around evening time; however, don't function admirably during sunshine enlightenment since daylight reflections make it difficult to distinguish retinal impressions of infrared waves.

Strategies utilizing light enlightenment, for the most part, discover the area of the face and eyes by utilizing PC vision systems. Viola-Jones utilized the Haar-like features for face and eye localization. This strategy is utilized in numerous investigations [21]. Wu et al. utilizes AdaBoost classifier for face localization and utilizes the intensity of the picture to distinguish the pupils of the eyes. At that point, they take outspread symmetry change and use a support vector machine (SVM) to discover the area of eyes. After finding the area of eyes, utilizing local binary patterns (LBP) and SVM classifiers, they detect the eye state[22].

In [23], they accepted that eye district is recognized effectively, and afterward, they utilize the eye picture to distinguish whether the eye is completely open, halfway open, or completely shut. They utilize singular value decomposition (SVD) and the term called Eigen-eyes and choose which group the eye picture has a place with. In addition to eye closure, yawning information is utilized as a marker of drowsiness in [24]. They

merged mouth and eye state information to distinguish drowsiness. Notwithstanding, Vural et al. state that is yawning, which is thought to be prescient of tiredness, is a negative indicator of the 60-second window preceding accident [25]. Drivers yawn less at times before nodding off. They utilize facial developments for distinguishing tiredness. Facial developments are separated by the assistance of a tool stash called Computer Expression Recognition Toolbox (CERT), which is a wholly programmed instrument for outward appearance acknowledgment [26]. They utilize facial action coding systems and discover which action units have positive or negative connections with tiredness, and afterward, by utilizing this activity unit information, they choose whether the driver is drowsy or alert [27, 28].

In [29] a system developed to detect driver drowsiness which uses a camera to monitor the movements of the driver carefully and accurately; To detect the driver's eyes from the image extracted from the video input and then make the processing on the image by detecting the driver's flashing patterns, when the driver's eyes are closed for a specified period, the system moves to detect the front of the driver's head. Finally, after making sure that the driver is in a drowsy state, the system will alert the driver. All this was done using a cascade arm detector from the OpenCV library.

One of the basic indicators of sleep is yawning, so in [30], a system has been developed to detect drowsiness in real-time to check if the driver wants to sleep or not. To achieve their objective, they need first to detect the face by revealing the color of the skin and then dividing the area into the upper and lower parts. The system will focus on the lower part to monitor the mouth. Finally, the properties extracted for yawning are the histogram values taken from the vertical projection of the lower part of the face.

In [31] the author proposed and embedded an automated framework that uses computer vision with a smart camera platform to detect yawning. In this model, the Viola-Jones algorithm is modified and used for face and mouth detections; thereafter, a back-projection theory is used in order to measure the rate and the number of mouth changes to detect yawning.

In 2017 Park S., Pan F., Kang S., and Yoo C.D proposed a deep architecture referred to as deep drowsiness detection (DDD) network for learning effective features and detecting drowsiness given an RGB input video of a driver. The DDD network

comprises of three deep networks in order to achieve global robustness to background and environmental variations and learning local facial movements and head gestures important for reliable detection. The results of the three networks are integrated and fed to a softmax classifier for drowsiness detection. DDD achieves 73.06% detection accuracy on the NTHU-drowsy driver detection benchmark dataset, as shown by experimental results [32].

A cheap, non-intrusive, facial, and eye monitoring model is proposed in [33] for driver drowsiness detection. This work ended up with the fact that although the proposed models in literature have high accuracies, they cannot be applied in real-world cases. Thus, in this work, a reliable model is proposed based on the eye pupil detection and various models of machine learning.

In [34] a convolution neural network is used to build a drowsiness detection model that can deal with complex situations like head deflection. In this model, the fatigue characteristics are recognized from the acquired eye and mouth images. Afterward, the multi-task cascaded convolutional neural network is used to detect the fatigue of the driver. Although this model has high accuracy and is easy to implement, it suffers from low detection speed.

1.2 Objectıves and Aıms

1.2.1 Overall Objective

According to the formulated problem, the purpose of this research is to build novel machine learning-based models that are capable of identifying and detecting driver drowsiness.

1.2.2 Specific Aims

To achieve the overall objective, some specific aims have to be considered, such as:

1. Develop novel optimization algorithms.
2. Pre-process the images and extract the valuable features of them.
3. Hybridize the proposed optimization algorithms in 1 with candidates of machine learning algorithms, such as a multi-layer neural network for better detection accuracy.

1.3 Research Contribution

The research contribution is concluded in three things: creating a novel drowsiness detection model which is more reliable, creating at least one novel optimization algorithm, and creating a machine learning model that can provide an efficient and accurate way to describe the driver drowsiness state.

1.4 Scope of Study

The scope of this research is determined as the following:

1. Swarm intelligence algorithms.
2. Artificial neural network and fuzzy inference systems.
3. Image preprocessing and feature extraction.

1.5 Methodology

This study went through four steps. The first step focused on developing novel optimization algorithms by modifying the artificial bee colony algorithm (ABC) and two of its last versions. The second step focused on the drawbacks of the available models and tried to build a novel model to solve these drawbacks. Viola and Jone's method was used to detect the drivers' faces, eyes, and mouth. Then, texture features were extracted from the regions of the eyes and mouth each separately. After that, three classifiers were trained separately (one for each eye region and one for the mouth region). The classification stage was implemented using Multi-layer Artificial Neural Network (ANN), which trained using the developed optimization algorithms. Finally, the fuzzy inference system was implemented to take the final decision of the driver's drowsiness state. The methodology of this study is summarized in Figure 1.1.

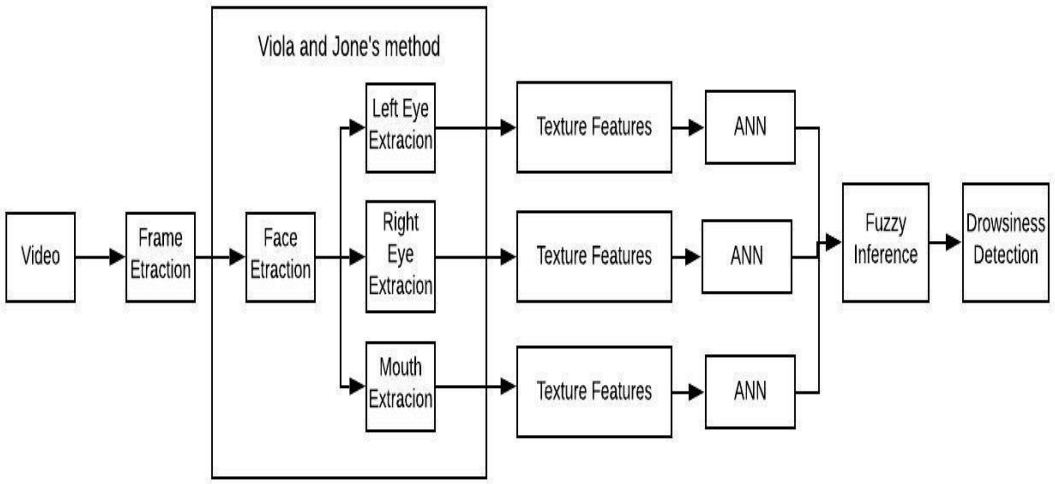


Figure 1-1 Block diagram of the proposed model

1.6 Outline

The rest of this thesis is as follows:

Chapter 2 gives background information on some image processing methods, feature extraction, swarm optimization algorithms, and neural networks.

Chapter 3 describes the method we propose to detect drowsiness and analyses all of the steps of the solution.

Chapter 4 presents the test data, ground truth data, experiments, and the results of the experiments. Comments and comparisons are made on the results.

Chapter 5 includes the conclusion and ideas or suggestions for future studies.

CHAPTER 2
BACKGROUND METHODS

2.1 Viola and Jones Method

Viola and Jones proposed a method for object detection, which consists of weak cascaded classifiers, but their general output is a strong classifier. In Viola and Jone's method, four principles have to be taken into consideration: haar-like features, integral image, AdaBoost, and cascading.

In each stage of the cascade, there is one weak classifier called decision stumps. Each weak classifier is trained using a technique called boosting to reject some portion of the non-target object. Boosting provides the opportunity to train an extremely accurate classifier by taking the weighted average of the decisions made by the weak classifiers. Although it takes a long time for training, it rapidly classifies images.

In order to extract facial features, the classifiers use haar-like features. A target size window is passed along the entire image. And if the object is not found in the current region, the detector slides the window to the next location. If the object is found in the current region, the classification process of this region is finished by this weak classifier, and the detector transfers the window to the next stage classifier. The detector identifies an object located at the current window location if the object region was found in the final stage classifier [21]. Figure 2.1 summarizes this mechanism.

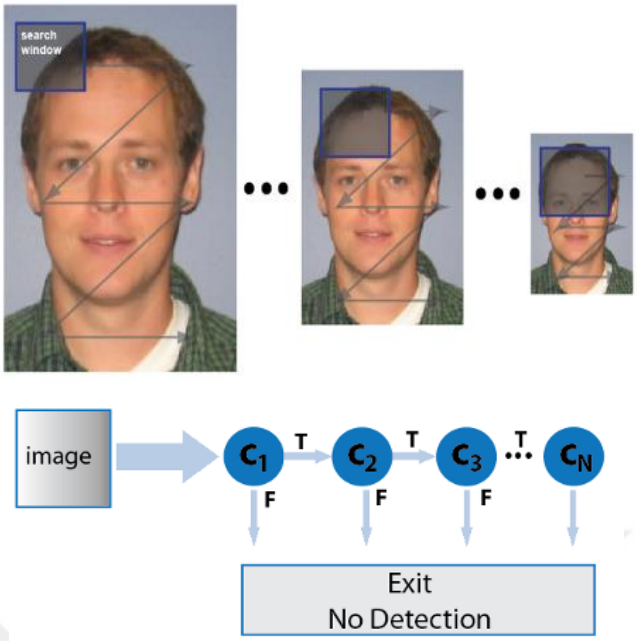


Figure 2-1 Cascade mechanism [MATLAB Document]

2.2 Feature Extraction

Computer vision still has a challenging problem with extracting optimal features, which can represent the intrinsic contents of the images as thoroughly as possible. The images can consist of redundant, irrelevant, and appropriate characteristics. These characteristics lead to high complexity and low learning difficulties in real applications, such as drowsiness detection. The features extracted from images are divided into three kinds: color features, texture features, and shape features. Color features are the first kind of features used to distinguish between images. But most of its methods are sensitive to noise so they cannot be used for driver's drowsiness detection [35]. Besides, shape features cannot be used for the driver's drowsiness detection since they are computationally complex and not easily implementable [36]. In contrast, texture features are easily implementable and not computationally complex. Texture features can be, in general, divided into spectral texture methods and spatial texture methods. Spectral texture methods are more robust and less computation than spatial texture methods [35].

The spectral texture methods transform an image from the original image domain into frequency domain and then calculates feature from the frequency domain image. Gray Level Co-occurrence Matrix (GLCM) is one of the common methods of spectral texture

features [37]. By calculating how often pairs of pixels occur in an image with specific values and a specified spatial relationship, the GLCM characterize the texture of an image [38]. This calculation is used to create the GLCM matrix, and then statistical metrics are derived from this matrix.

Given an image P, of size m×m, the co-occurrence matrix G can be defined as it is shown in Eq. 2.1:

$$G(i,j) = \sum_{x=1}^{m} \sum_{y=1}^{m} \begin{cases} 1, & \text{if } P(x,y) = i \text{ and } P(x+d, y+d) = j \\ 0, & \text{Othewise} \end{cases} \quad (2.1)$$

Where the distance (d) and the angle (θ) between the pixels are the two control parameters used to create different matrixes of GLCM, Figure 2.2 shows the spatial connections between pixels in terms of distance and angle.

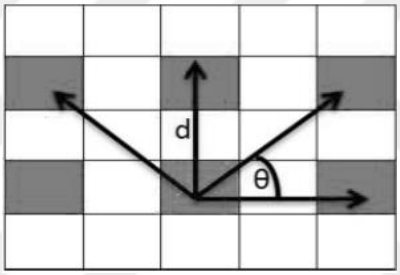


Figure 2.2 The spatial connections between pixels [28]

2.3 Optimization Algorithms

2.3.1 Overview

Most real-world applications have many optimization problems, many of them often being non-convex, non-differential, or even implicit equations. For conventional gradient-based optimization approaches, solving these problems is rather difficult[39]. Population-based metaheuristic (also known as P-metaheuristic or stochastic) algorithms are powerful optimization algorithms that are often used as an alternative, efficient methods to the traditional optimization techniques in order to overcome their limitations [40]. Indeed, P-metaheuristic algorithms achieve better results than traditional optimization algorithms [41].

Meta-heuristics have shown superior performance in solving many of engineering problems as an effective solution and are therefore increasingly being taken care of by

researchers for the following reasons: they are simple; thus, scientists from different fields can easily learn and apply them to solve their problems. Besides, these methods are flexible; therefore, they do not need any specific modification in their structure to use them in any application. Furthermore, they are derivative-free mechanisms; consequently, they are highly optimized for real problems with costly or unknown derivative data. Finally, because of their stochastic nature, they are local optima avoidable, which makes them a good option for real problems [42]. These characteristics motivated the scientists to simulate various natural concepts, introduce new metaheuristics, hybridize two or more, or enhance the current meta-heuristics.

In literature, there are many P-metaheuristic algorithms and which one of them is the best depends on the application itself; that is, one algorithm might show good results in one specific application but show bad results in other applications [40]. In fact, each algorithm has its limitations; some have good exploitation ability while the others have good exploration ability [41]. Hybrid optimization algorithms are proposed in the literature to overcome the drawbacks of the individual algorithms and mix their strengths. The power of the hybridization leads researchers to test various global and local hybridization strategies to improve the exploitation and exploration of the algorithms[43].

ABC algorithm, originally developed in 2005 by D. Karaboga, is one of the best algorithms of population-based meta-heuristics which its simple structure, low number of parameters and ease of implementation attracted many researchers to release new better versions of it. Although the convergence rate of ABC in solving complex optimization problems is competitive compared with other population-based optimizer algorithms, which is due to its good exploration, it is not good enough because of its poor exploitation [44]. Each researcher tried to explain the reason for this drawback in his point of view and modified the original ABC in a way to manage and decrease this reason as much as possible.

In both employed and onlooker phases of ABC, only one of the parent solution parameters of the randomly generated new solution is modified. This form of updating makes the update during the search along the current axis and ignores the other axes, which leads to weak exploitation [45]. This fact leads the authors in the literature to

change the updating equations of ABC, such as ABC/best and GABC [46, 47]. Besides, in the scout phase, the abandoned food source is replaced by a new random source, which confirms the fact that the ABC search equation and structure focus more on exploration while the effective meta-heuristic depends heavily on a good compromise between exploration and exploitation. The main challenge is, therefore, to improve the performance of the ABC algorithm in balancing exploration and exploitation of the solution search equation.

2.3.2 The original ABC

The ABC algorithm is a kind of optimization algorithms inspired by the wobble dance and brilliant behavior of honey bees. In ABC, the solution imitates the food source for the bee, and the fitness of the solution imitates the quantity of nectar in each source of food. The ABC algorithm splits the bee foraging activities into three stages: The first half of the colony is made up of employed bees that are responsible for randomly looking for a portion of better food in the vicinity of the respective parent food source then transmit food source information to onlooker bees. The second part of the colony is made up of onlooker bees that use the information coming from onlooker bees to check for better sources of food. If a predetermined number of evaluations (limited) does not improve the quality of a food source, this source of food is abandoned by its employed bee, which then becomes a scout bee and starts to look for a new random source of food. In ABC, there are three control parameters used: The number of employed equals the number of onlooker bees, which equals the number of food sources, limit, and the maximum number of cycles.

2.3.2.1 Initialization

The first step of any meta-heuristic algorithms is the initialization step. In ABC Eq. 2.2 is used to generate a random population of food sources (SN). Where x^{j}_{max} is the upper bound and x^{--j}_{min} is the lower bound of the jth food source (solution).

$$x_i^j = x_{min}^j + rand(0,1)(x_{max}^j - x_{min}^j) \qquad (2.2)$$

2.3.2.2 Employed Bee Task

Eq. 2.3 is used to change the existing solution in this phase where k ∈ [2, SN] and j ∈[2, D] are randomly chosen indices, where SN is the number of food sources, D is the dimension of the solution and k is not equal to i. The current solution is updated and replaced by employed bees based on the fitness value of a new solution determined by Eq.2.4.

$$v_i^j = x_i^j + rand(0,1)(x_i^j - x_k^j) \quad (2.3)$$

$$Fit(i) = \begin{cases} \frac{1}{1+f(i)}, & f(i) \geq 0 \\ 1 + |f(i)|, & f(i) < 0 \end{cases} \quad (2.4)$$

2.3.2.3 Onlooker Bee Task

The task of the onlooker bees starts with the end of the employed bee task. In this phase, every onlooker bee will fly to a food source according to the quality information calculated from the food resources provided by the employee bees. The probability of the quality of those sources is calculated by Eq. 2.5. The higher the fitness value is, the greater the chance of selection is.

2.3.2.4 Scout Bee Task

If that source of food is not changed for a limited number of iterations, it is presumed that a food source is exhausted. After that, the employed bee becomes a scout, and a randomly new food source in the search area is generated using equation 2.2. The main steps of ABC are given in Algorithm 1 below [48]

$$P_i = \frac{Fit(i)}{\sum_{i=1}^{N} Fit(i)} \quad (2.5)$$

Algorithm 1: Artificial Bee Colony (ABC) [48]

Step 1: Initialization.

Step 2: Employed bee phase.

Step 3: Onlooker bee phase.

Step 4: Scout bee phase.

Step 5: Memorizing the best solution.

2.3.3 The other versions of ABC

As the performance of the original ABC in solving difficult optimization problems is poor, several ABC models were developed to boost their performance. Inspired by PSO, a gbest-guided ABC (GABC) was introduced, which merged knowledge from the best global solution to improve the speed of convergence in the search equation [47]. In [49], the interactive ABC was exploited by the inclusion in the search solution equation of the principle of universal gravitation. In [46] two search equations were designed in order to maintain an excellent balance between the exploration and exploitation of ABC. In this work, a control parameter p was developed in which two search equations, focusing respectively on exploitation and exploration, are combined with balancing exploitation with exploration (called MABC). In [50], a parameter P was introduced to take account of the balance between the search equations and introduced an incoming search strategy. [51] has developed another way of making use of the knowledge of the best global in the onlooker phase. The results of the experiments show that, in some cases, the approach described exceeds the comparable algorithm. [52] Suggested a new search equation like that found in the Genetic Algorithm crossover procedure (named CABG). Besides, the orthogonal learning method is used with three different versions of ABC to find out more valuable information during the searching process. The simulation results show that CABG is competitive and effective. PS-MEABC inspired the search equation, that uses the knowledge of the best solution and other good solutions, from PSO [53]. The Gaussian search equation has been proposed to provide solutions for the onlooker phase. This equation takes advantage of exploiting the hidden information found in the best solution to improve the exploitation [54]. Then they developed two new equations to act as search equations for both onlooker and employee phases and used Powell's method to enhance the exploitation [55]. A Gaussian ABC was provided, using a parameter p to control gaussian and uniform distributions frequency [56]. A quick ABC (qABC) was proposed by Karaboga, where a novel search equation for the onlooker phase has been introduced to exploit the neighbors of the best solution [57]. The directional information was appended to ABC to create a new search strategy that can construct a candidate solution based on the previous directional information (called dABC) [58].

There are also several other strategies used for improving ABC efficiency. For example, Rosenbrock ABC, referred to as RABC, was proposed by Kang et al. wherein ABC do the exploration phase, and the rotational path approach completes the exploitation phase. In [45], two control parameters are presented and added to ABC to control the magnitude and frequency of perturbation. In [59], the chaotic mapping technique is used, in the initialization and Scout phase, to increase ABC's performance and quality (called ERABC). In [60] Hooke–Jeeves pattern search is combined with ABC, where ABC complete the exploration, and the exploitation is satisfied by pattern search. Its results showed that it is promising with regard to convergence speed, the accuracy of the solution, and efficiency. ABC algorithm with multi-strategy ensemble, called MEABC, is proposed for the purpose of solving optimization problems with various features. It includes a pool of different resolution search strategies that competes with each other during the search process to produce offspring [61]. Also, five multiple update strategies were combined with ABC to enhance its performance [62]. A self-adaptive local search strategy is implemented with ABC to exploit the searching area near the best solution, which leads to accelerating ABC convergence [63]. A depth-first search (DFS) framework, which assigns more computing resources to higher quality food sources, and two search equations, which used to exploit the elite solutions, added to enhance the performance of ABC (called DFSABC_elite). The experimental results prove that the suggested framework can speed up the convergence rate, and DFSABC_elite has quality and robustness better than the compared algorithms [64].

A hybrid optimization algorithm was also developed, based on ABC and other methods. For instance, Differential Evolution strategy is hybridized with ABC, and the performance of this hybridization, called DE-ABC, is better than the performance of both DE and ABC [65]. Hybridization between ABC and PSO is produced in order to share useful information between them. Two information-sharing processes are added between PSO and ABC, and the proposed hybridized method shows competitive results [66]. To obtain the parallel value of GA calculation and the speed convergence of ABC hybridization between GA and ABC is produced by exchanging knowledge between the bee colony population and the GA population [67]. Levy flight and opposition-based learning methods are used to improve the performance of ABC by enhancing its exploitation in different works [68-70]. Another hybridization of ABC with teaching-

learning based optimization (TLBO) is produced, called TLABC, where ABC exploration combines with TLBO exploitation and enhances the performance of both ABC and TLBO [71].

In this study, novel modification is proposed and implemented on ABC, DFSABC_elite, and TLABC algorithms.

2.3.4 The original version of TLABC

TLABC is a hybridization between TLBO and ABC algorithm, which combines the advantages of both (the exploration of ABC and the exploitation of TLBO). It effectively employs three hybrid search phases as follows [71]:

2.3.4.1 Teaching based employed bee phase

Here each employed bee uses a hybrid of TLBO and mutation operator of differential evolution to search a new food source, which can develop the variety of search tendencies extraordinarily and upgrade the searchability of TLABC. Eq. 2.6 is the search equation used in this phase.

$$u_{i,d} = \begin{cases} x_{i,d}^{old} + rand_2\left(x_{teacher,d} - T_F \,.\, x_{mean,d}\right), & if\ rand_1 < 0.5 \\ x_{r1,d} + F \,.\, \left(x_{r2,d} - x_{r3,d}\right), & otherwise \end{cases} \quad (2.6)$$

Where i is the current solution and $d \in [1, D]$; $r_1, r_2, and\ r_3 \in [1, SN]$; $rand_1$ and $rand_2$ are normaly distriuted random numbers; *TF=round(1+rand * 1);* and *F* is a scale factor in [0,1].

2.3.4.2 Learning-Based Onlooker Bee Phase

In this stage, an onlooker bee chooses a food source to search out as indicated by the selection probability, which is determined to utilize Eq. 2.5. After that, the onlooker bee finds out new food sources using the TLBO's learning strategy expressed in Eq. 2.7.

$$u_s = \begin{cases} x_s + rand\,.\left(x_s - x_j\right), & if\ f(x_s) \le f(x_j) \\ x_s + rand\,.\left(x_j - x_s\right), & if\ f(x_j) > f(x_s) \end{cases} \quad (2.7)$$

Where $j \in [1, SN]$ and j≠s.

2.3.4.3 Generalized Oppositional Scout Bee Phase

In this stage, if a nourishment source cannot be improved further for a specific period time, it is viewed as depleted and would be relinquished. At that point, an arbitrary candidate solution and the generalized oppositional solution of it are created. The best solution of them is utilized rather than the old depleted nourishment source. Eq. 2.2 and Eq. 2.8 are the equations used in this phase, respectively. The TLABC framework is shown in Algorithm 2.

$$x_{ind}^{GO} = k\,.\left(a_j + b_j\right) - x_{ind} \qquad (2.8)$$

Where k is a random number in [0, 1], $a_j = \max(X)$, and $a_j = \min(X)$.

Algorithm 2: Teaching Learning Based Artificial Bee Colony (TLABC) [71]

Step 1: Initialization.
Step 2: Teaching based employed bee phase.
Step 3: Learning-based onlooker bee phase.
Step 4: Generalized oppositional scout bee phase.
Step 5: Memorizing the best solution.

2.3.5 The original version of ABC with Depth-first-Search (DFSABC_elite)

It is common knowledge that ABC is one of the most common swarm algorithms due to its high exploration. However, sadly, its search equation (Eq. 2.3) makes it has slow convergence. Consequently, in [64], a good balance between exploration and exploitation in addition to the depth-first-search framework were proposed. There are two modifications to ABC introduced in this version, which are:

1. Two different new search equations replace the search equation used in the employed and onlooker bee phase of ABC. Consequently, the exploitation ability has increased.

2. In the onlooker bee phase, just the best solutions (elite solutions) are used among the overall population. Since more focus is paid on elite solutions regions, the more improving in exploitation will be noted.

These two modifications guarantee the improvement of ABC exploitation and its overall efficiency. Since DFSABC_elite is another version of ABC, it has the same phases with some changes in employed and onlooker phases. Consequently, in this section, we will discuss both the employed and onlooker phases in detail.

2.3.5.1 Employed Bee Phase

At the beginning of this phase, the elite solutions (P food sources usually 10% or 20% of the population) are chosen, and a flag parameter flag1 is set to 1. Where flag1 is equal to 1, for each employed bee a random source of food X_r is selected and a new solution X_{new} is generated in the neighbourhood of that employed bee by Eq. 2.9.

$$X_j^i = X_j^b + rand(0,1)\left(X_j^b - X_j^k\right) \qquad (2.9)$$

where j ∈ [2, D] and X_j^i is the j^{th} dimension of the i^{th} new solution in each iteration, X_j^b is the j^{th} dimension of b^{th} solution, and the b^{th} solution is a randomly chosen solution from the elite solutions, X_j^k is the j^{th} dimension of k^{th} solution, and the k^{th} solution is chosen randomly from the whole current population where k, b, and r are different from each other. The old food source is replaced by the new one if and only if its objective value is better than the old one and the failure value of the i^{th} food source will be reset to 0, and flag1 resets to 0 (This means that the next employed bee in the same iteration will also search in the same food source as flag1 is not 1). Otherwise, the old food source will also be maintained, failure value of the i^{th} food source will be increased by 1, and flag1 resets to 1 (this means that the next employed bee is looking for another random source of food because Flag1 is 1).

2.3.5.2 Onlooker Bee Phase

At the beginning of this phase, a flag parameter flag2 is set to 1. Where flag1 is equal to1, for each onlooker bee a random source of food X_b is selected from the elite

solutions and a new solution X_{new} is generated in the neighbourhood of that onlooker bee by Eq. 2.10.

$$X_j^i = \frac{1}{2}(X_j^b + X_j^{best}) + rand(0,1)\left(X_j^b - X_j^k\right) \qquad (2.10)$$

Where X_j^{best} is the best solution among all the population and is different from X_j^b. The old food source is replaced by the new one if and only if its objective value is better than the old one and the failure value of the i^{th} food source will be reset to 0, and flag2 resets to 0 (This means that the next onlooker bee in the same iteration will also search in the same food source as flag2 is not 1). Otherwise, the old food source will also be maintained, failure value of the i^{th} food source will be increased by 1, and flag2 resets to1 (this means that the next employed bee is looking for another random source of food because Flag2 is 1). The main steps of DFSABC_elite are the same as ABC and given in Algorithm 1.

2.4 Artificial Neural Network

The ANN is simply a collection of artificial neurons that are close together and internally connected. These elements are a simulation of biological nervous systems. Naturally, the connections between elements largely determine the network function. And training the neural network to perform a particular function can be done by adjusting the values of the connections (weights) between elements. So the general idea of artificial neural networks is that such parameters can be adjusted so that the network exhibits some desired or interesting behaviour. Figure 2.3 summarizes the mechanism of artificial neural networks.

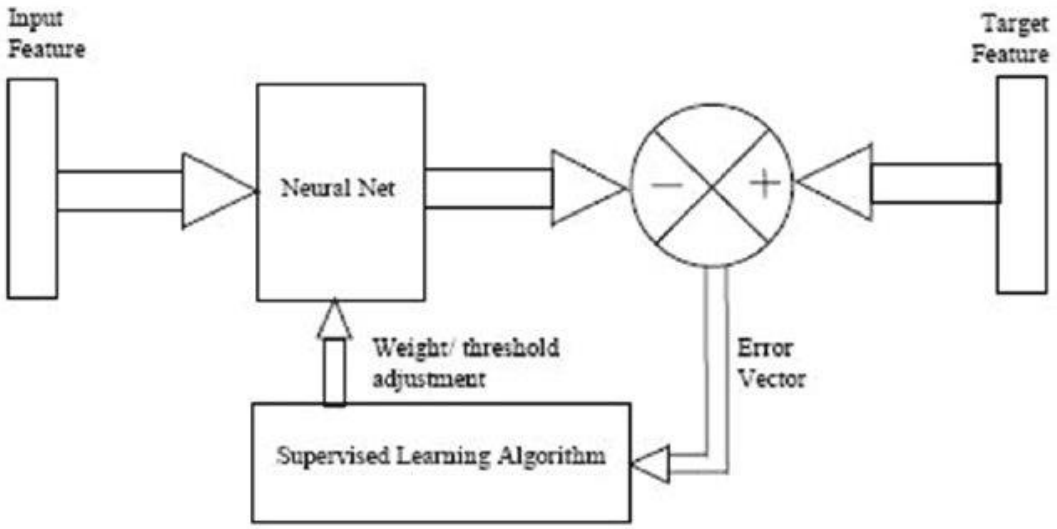


Figure 2.3 Mechanism of Artificial Neural Network [72]

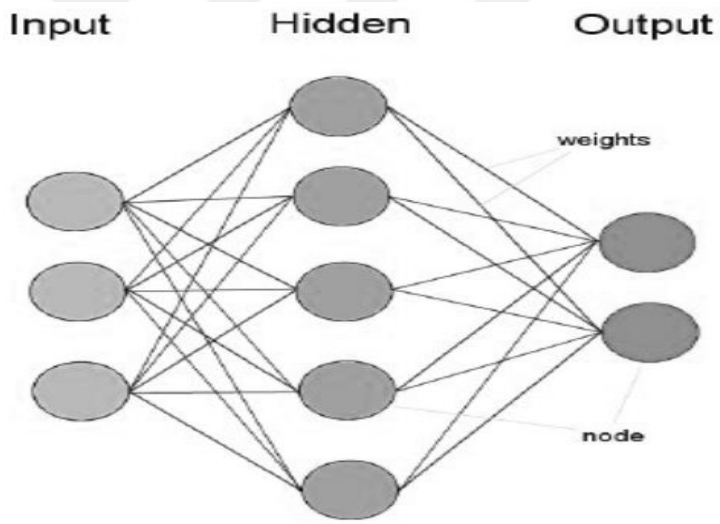


Figure 2.4 Multi-Layer Neural Network Architecture [72]

A Feedforward neural network with enough neurons in just one hidden layer can solve any finite mapping issue [73]. The Multi-layer neural network is a kind of artificial neural network that consists of three layers (input layer, hidden layer, and output layer). The architecture of the multi-layer neural network is shown in Figure 2.4, where each layer gives the input to the next, and there are a number of neurons in each layer. Each neuron is connected with neurons in the next layer by weights values. Thus, we can train the network to do a particular job or to achieve some desired end by adjusting these weights.

The input layer represents the features of data that are fed into the network. This is the layer of the network in which its weights values never change. Then the output of every input neuron of this layer is spread along all neurons in the hidden layer and multiplied with their weights. The hidden layer neurons receive data from the input layer and processed them. The output of the hidden layer is then sent to the output layer, but it will also be changed by using weights connected between the hidden and the output layers. The output layer of neurons process data obtained from the hidden layer and produce an output. Each neuron has an activation function. This activation function defines the output of this neuron. It might also be noted; only non-linear activation functions allow nontrivial problems to be calculated via a small number of neurons [74].

The sigmoid function, whose graph is "S"-shaped, is the most commonly used activation function. It has been defined as a strictly increasing function showing a graceful balance of linear and non-linear behaviour. There are two types of sigmoid function, the first one called log-sigmoid and the second one called tan-sigmoid. Mathematically the two types are equivalent, but the log-sigmoid differs in that it runs faster than tan-sigmoid [74]. The log-sigmoid and tan-sigmoid can be represented as Eq. 2.11. and Eq. 2.12. respectively where the limitation of the log-sigmoid output in the period [0, 1] and the limitation of the tan -sigmoid output in the period [-1, 1]. Figure 2.5 shows both shapes.

$$y_k = \frac{1}{1+e^{-v_k}} \qquad (2.11)$$

$$y_k = \frac{2}{1+e^{-2v_k}} - 1 \qquad (2.12)$$

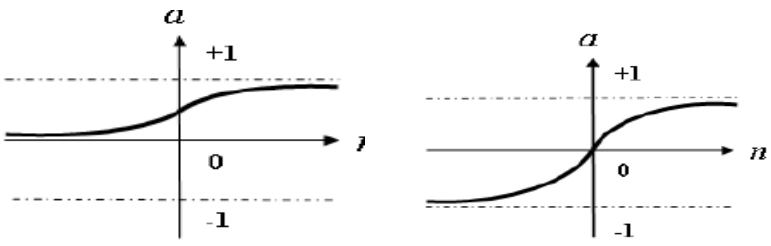


Figure 2.5 (a) Log-sigmoid function (b) Tan-sigmoid function

Artificial Neural Network has been used to solve various types of classification issues. Nevertheless, the convergence is gradual and not always guaranteed for ordinary training algorithms, such as Backpropagation (BP). This issue requires a successful optimization strategy to achieve rapid convergence and higher levels of accuracy [75].

However, swarm intelligence algorithms have been used in many studies for Neural Network training, for example, Ant Colony Optimization (ACO) [76, 77], PSO [78], DE [79], Evolutionary Algorithms (EA) [80], GA [81], Artificial Bee Colony (ABC) [82], Hybrid Ant Bee Colony (HABC) [83], GABC [84], Global Hybrid Ant Bee Colony (GHABC) [76], Bat Algorithm (BA) and Cuckoo Search (CS) [85, 86] and so on.

In this study, the proposed novel swarm optimization algorithms are used to train the multi-layer neural network.

2.5 Fuzzy Inference System

Fuzzy inference system is defined as a field of soft-computing which works on the vagueness discipline and gives interpretable results. The fuzzy system uses the theory of fuzzy set, fuzzy reasoning, and inference mechanism so that systems like this can be used in different applications. The advantage of fuzzy set theory over the classical set theory is that the object can belong to more than one set in a certain degree of membership to each of them. As a result, fuzzy systems are preferred to be used if knowledge must be based on verbal, ambiguous, or imprecise information [87].

Any fuzzy inference system (FIS) goes through four steps: Fuzzification, Rule Base, Inference Mechanism, and De-fuzzification. The process of fuzzification is defined as the process by which a crisp set is transformed into a fuzzy set (linguistic variables). While the process of de-fuzzification is defined as the process by which a fuzzy set is transformed into a crisp set. The rule base is the unit of fuzzy inference system, which consists of If-Then rules connected in parallel. Finally, the inference mechanism is the decision-making unit, which uses the base rule unit to formulate the mapping from a given input to an output. The structure of the fuzzy inference system is shown in Figure 2.6.

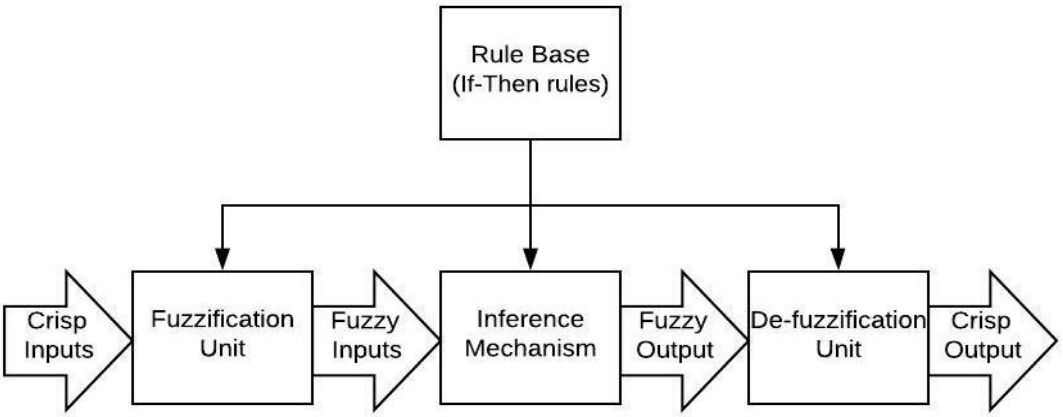


Figure 2.6 Structure of the Fuzzy Inference System

In FIS, the crisp inputs and outputs are encoded into fuzzy sets, which are overlapping in vague boundaries. All fuzzy sets are defined in terms of their membership functions, which the human experts provide, and then the fuzzy system developers set them up (optimize them). Sigmoid, Gaussian, and Pi are common functions used as membership functions. These functions nevertheless increase the calculation time. Consequently, most systems also use linear fit functions instead. The fuzzy set can contain other sets called the subset.

When crisp inputs are encoded into fuzzy sets, fuzzy rules are applied to them by the inference mechanism to map the inputs into output. The concept of "fuzzy rule" itself is quite misleading because the majority of applications use this rule to specify a mapping of the input and output [88]. Fuzzy inference mechanism uses the inputs and the available fuzzy rules as a method to conclude the output of the inference mechanism [88]. It should be noted that the fuzzy IF-THEN rules use fuzzy logic while the classical IF-THEN rules use binary logic. All rules in FIS are fired to a partial degree. If the antecedent is applicable to a certain degree of membership, the consequent is valid to the same degree as well.

Mamdani and Sugeno are the two common types of inference mechanisms. However, the fuzzy inference method of Mamdani is the most popular fuzzy methodology. Linguistic variables are the antecedent and consequence of the rules in the Mamdani-type. Accordingly, the output of Mamdani needs a defuzzification technique to convert the linguistic output into crisp value. Sugeno-type is similar to the Mamdani except that

the rule consequent of Sugeno is not a fuzzy set; it is a mathematical function of the input variables. Thus, the overall output of Sugeno-type is achieved with the weighted average operation instead of the weighted sum operation used in Mamdani. Indeed, this process increases the calculation speed [87].

Tiredness is a type of body state which is preferred to be expressed in levels instead of binary expressions (found or not). Accordingly, using fuzzy systems to detect the state of driver drowsiness is more logical and practical than crisp detection systems.

In [90] fuzzy system based on image processing has been used to detect the drowsiness of the driver. They conclude that the fuzzy system is more accurate than the other systems, which only depend on only the time of closure of drivers' eyes. But they just depended on eyes blinking frequency and eyelid closure duration to build their system, which makes it not reliable and trustable as mentioned above. Moreover, it does not work well in a low lighting environment [90].

CHAPTER 3
PROPOSED METHODS

3.1 Overview

This thesis uses eye and mouth features to propose a reliable drowsiness detection model based on the fuzzy system and multi-layer neural networks. The proposed model starts by extracting the frames from the videos. Then, the faces are detected from these frames by using Viola and Jone's method. After that, the left and right eyes and the mouth regions are detected by using Viola and Jone's method too and cropped into three separate images. Now we have three images, and each one of them will go through the feature extraction and detection stage separately. In the feature extraction stage, the texture features are extracted by creating the GLCM matrix and extracting 22 statistical features from it. Then, the 22 texture features are inserted into the trained multi-layer neural network. The multi-layer neural network is trained using the proposed hybridization algorithms. We used neural networks to classify the eye and the mouth state. More clearly, we used it to determine if an eye is open, half-closed, or closed. Moreover, we used it to determine if the mouth is open or closed in the system we propose. Finally, the fuzzy inference system is used to detect the drowsiness level of the driver by using the output of the trained multi-layer neural networks of eyes and mouth as inputs. In this chapter, we will discuss each step in detail as the following.

The systems that depend on one parameter like yawning, eye state, PERCLOS, or blinking frequency to detect the driver’s state may fail to detect the driver’s true state for one reason or another, as we mentioned in the problem formulation of this study. Therefore, it cannot be denied that when the model depends on multiple visual parameters, the model will be more effective and have more accuracy than the model that depends on just one parameter. As a result, our proposed model combines multiple visual parameters like eye state and yawning analysis to detect the drowsiness level of the driver. Parameter selection is a quid pro quo between accuracy and speed since the more features of the driver we observe, the better accuracy we can achieve, but this

might affect the response of a real-time system in terms of speed. Hence it is important to carefully choose these parameters in a way that the speed of the system is not compromised for the sake of accuracy.

3.2 Proposed Classification Models

In this study, we proposed three classification models by proposing three versions of ABC and use them for training the multi-layer neural network. In the scout phase of the original ABC and its other versions, the abandoned food source is replaced by a new random source or by its oppositional, which leads to increase exploration. But ABC and its versions already have high exploration. So in this work, we focused on using the scout phase in exploitation instead. Accordingly, to achieve good exploitation, two steps have to be taken under consideration: first, we have to know where the fertile area to be exploited is. Then we can exploit this area by using a good local search equation. The first step is completed by introducing indicator parameters for each solution, which increases in each evaluation if there is an improvement in that solution. The idea here is that in the fertile area, the solution will continue to improve, unlike the useless area. The second step is completed by exploiting the neighbour of the solution with the maximum indicator. Here we focus on replacing the scout phase from exploration phase to exploitation phase by replacing the abandoned food source with a new solution in the fertile area. To achieve this scenario, the gaussian search equation is used. This structure is incorporated in the original ABC and other well-known versions of ABC such as DFSABC_elite and TLABC. The respective amended ABCs are therefore referred to as MABC, MDFSABC, and MTLABC respectively.

The most critical factor in training an MLP using meta-heuristics is the problem representation. The training process of MLP focuses on finding an optimal set of weights and biases values that provides the highest accuracy. These values are the variables of the meta-heuristics. As stated earlier, in training an MLP, the goal is to achieve the highest accuracy of classification for training and testing datasets. The Mean Square Error (MSE) is a standard metric used for the MLP evaluation. A set of training samples is applied to the MLP, and the MSE equation calculates how much the desired output is differing from the value derived from the MLP. Then, the mean of MSE over all the training samples is taken to evaluate the MLP performance. Therefore,

the objective of the MLP training process using meta-heuristics algorithms is to minimize the mean of MSE, as shown in Eq. 3.1.

$$Minimize : \overline{MSE} = \frac{\sum_{k=1}^{n} \left(O^k - D^k\right)^2}{n} \qquad (3.1)$$

Where n is the number of training samples, O^k is the actual output, and D^k is the desired output of the kth sample. The overall MLP training cycle using meta-heuristics algorithms is shown in Figure 3.1. The meta-heuristics algorithms change the weights and biases iteratively to reduce the average MSE for all training samples.

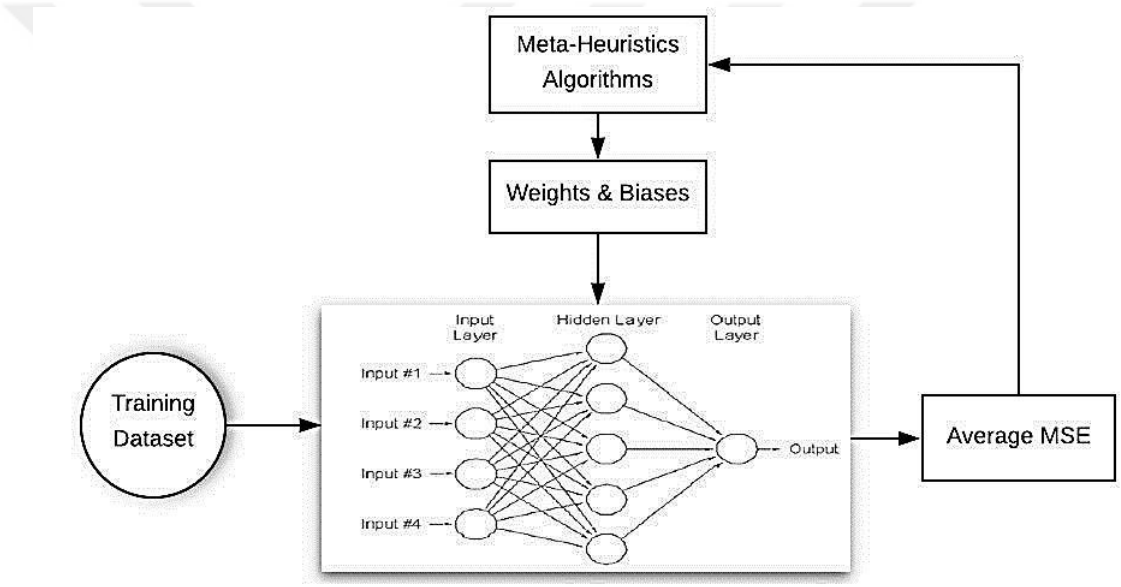


Figure 3.1 Multi-Layer Neural Network Training Process

The proposed Modification of ABC (MABC) TLABC (MTLABC) and DFSABC_ elite (MDFSABC) algorithms have the advantages of increasing the exploitation of the original versions since it applies the gaussian search equation to exploit the fertile area. As stated in the literature, ABC is not efficient enough in the exploitation, so if the abandoned food source is replaced by a new source from the fertile area in the scout phase, the exploitation would absolutely increase.

In fact, the good exploration ability of the ABC algorithm attracts a considerable number of researchers to hybridize it with many other swarm algorithms, as mentioned previously. Some of them keep both the search equations and the ABC framework, while others keep just the ABC framework. In the TLABC hybridization and DFSABC_

elite, only the framework was maintained; therefore, in this study, we have also modified the search equation used in the scout phase. This modification enhances the performance of exploitation searching of TLABC and DFSABC_ elite in the generalized oppositional scout bee phase, where the solution created randomly using Eq. 2.2 was replaced by a solution created using the gaussian equation using Eq. 3.2 around the fertile area of the search rather than the old depleted nourishment source. In fact, the scout phase of the original ABC, TLABC, and DFSABC_ elite generates solutions randomly to improve the exploration more.

$$x_{ind} = N(m, sd) \tag{3.2}$$

Where x_{ind} is the new solution generated in the fertile area using the normal distribution which its mean "m" is the average of the global optimal solution and the solution with maximum indicator value. N is the abbreviation of normal distribution and "sd" is the abbreviation of standard deviation.

It is known that the exploration (i.e., the ability of the algorithm to universally search for potential regions in the search space) of ABC is already high, and we here make some tradeoffs between the exploration and exploitation by generating solutions near the best solution instead of randomly. By doing so, we improve the exploitation of the algorithm and achieve some type of balance between exploration and exploitation. Besides, we add an indicator of the fertile area, which is the other modification we added. The fertile area could be discovered by using this indicator for each solution, which will increase whenever the solution increases. Thus the solution which its indicator is the highest is indeed in the fertile area

The MABC pseudo-code for the purpose of classification using Multi-Layer Perceptron is shown in Algorithm 3, the MTLABC pseudo-code for the purpose of classification using Multi-Layer Perceptron is shown in Algorithm 4, and the MDFSABC pseudo-code for the purpose of classification using Multi-Layer Perceptron is shown in Algorithm 5.

Algorithm 3: Modified Artificial Bee Colony (MABC)

Initialize the population (the weights and biases) to be in the range [-10, 10] by Eq. 2.2.
Select number of Input, Output and Hidden neurons for the NN according to the dataset.
For (i=1:Maximum number of iterations)
  for (i=1:SN)
    Generate a new solution X_{new} using Eq. 2.3 and pass them to the MLP.
   if $fitness\ (X_{new}) \geq fitness(X_i)$ %Fitness of the MSE of the trained MLP
      X_i is replaced by X_{new}
      Set failure(i) = 0, indicator(i) = indicator(i) + 1
    else
      Set failure(i) = failure(i) + 1
    end if
  end for
Determine the nectar amounts of food sources by Eq. 2.5.
  for (i=1:SN)
    Using the roulette method, some onlooker bees will be selected to move onto the food sources according to the selection probability of Eq. 2.5.
    Generate a new solution X_{new} using Eq. 2.3 and pass them to the MLP.
   if $fitness\ (X_{new}) \geq fitness(X_i)$ %Fitness of the MSE of the trained MLP
      X_i is replaced by X_{new}
      Set failure(i) = 0, indicator(i) = indicator(i) + 1
    else
      Set failure(i) = failure(i) + 1
    end if
  end for
Find the solution with maximum failure value.
   IF (max(failure) > limit):
      Find the food source with maximum improvement indicator (absolutely it will be in the fertile area of searching)
      Update its value using Eq. 3.2.
      Create its generalized oppositional solution using Eq. 2.8.
      The best solution of them is utilized rather than the old depleted nourishment source.
  End IF
End For

Employed Bee Phase

Onlooker Bee Phase

Scout Bee Phase

Algorithm 4: Modified Teaching Learning Based Artificial Bee Colony (MTLABC)

Initialize the population (the weights and biases) to be in the range [-10, 10] by Eq. 2.2.
Select number of Input, Output and Hidden neurons for the NN according to the dataset.
For (i=1:Maximum number of iterations)
  for (i=1:SN)
    Generate a new solution X_{new} using Eq. 2.6 and pass them to the MLP.
    if $fitness\ (X_{new}) \geq fitness(X_i)$ %Fitness of the MSE of the trained MLP
      X_i is replaced by X_{new}
      Set failure(i) = 0, indicator(i) = indicator(i) + 1
    else
      Set failure(i) = failure(i) + 1
    end if
  end for

Teaching Based Employed Bee Phase

Determine the nectar amounts of food sources by Eq. 2.5.
  for (i=1:SN)
    Using the roulette method, some onlooker bees will be selected to move onto the food sources according to the selection probability of Eq. 2.5.
    Generate a new solution X_{new} using Eq. 2.7 and pass them to the MLP.
    if $fitness\ (X_{new}) \geq fitness(X_i)$ %Fitness of the MSE of the trained MLP
      X_i is replaced by X_{new}
      Set failure(i) = 0, indicator(i) = indicator(i) + 1
    else
      Set failure(i) = failure(i) + 1
    end if
  end for

Learning Based Onlooker Bee Phase

Find the solution with maximum failure value.
  IF (max(failure) > limit):
    Find the food source with maximum improvement indicator (absolutely it will be in the fertile area of searching)
    Update its value using Eq. 3.2.
    Create its generalized oppositional solution using Eq. 2.8.
    The best solution of them is utilized rather than the old depleted nourishment source.
  End IF

Oppositional Scout Bee Phase

End For

Algorithm 5: Modified Depth-First-Search Artificial Bee Colony (MDFSABC)

Initialize the population (the weights and biases) to be in the range [-10, 10] by Eq. 2.2.
Select number of Input, Output and Hidden neurons for the NN according to the dataset.
For (i=1:Maximum number of iterations)

Employed Bee Phase

set flag1=1, set flag2=1
for (i=1:SN)
if flag1 == 1
Choose a random food source X_r from the population
end if
Generate a new solution X_{new} using Eq. 2.9 and pass them to the MLP.
if f(X_{new}) ≤ f(X_r) %MSE of trained MLP
X_r is replaced by X_{new}
Set failure(r) = 0, flag1 =0, indicator(r) = indicator(r) + 1
else
Set failure(r) = failure(r) + 1, flag1 =1
end if
end for

Onlooker Bee Phase

for (i=1:SN)
if flag2 == 1
Choose a random food source X_b from the elite solutions
end if
Generate a new solution X_{new} using Eq. 2.10 and pass them to the MLP.
if $f(X_{new}) \leq f(X_r)$ %MSE of trained MLP
X_b is replaced by X_{new}
Set failure(r) = 0, flag2 =0, indicator(r) = indicator(r) + 1
else
Set failure(r) = failure(r) + 1, flag2 =1
end if
end for

Scout Bee Phase

Find the solution with maximum failure value.
IF (max(failure) > limit):
Find the food source with maximum improvement indicator
Update its value using Eq. 3.2.
Create its generalized oppositional solution using Eq. 2.8.
The best solution of them is utilized rather than the old depleted nourishment source.
End IF
End For

3.3 Proposed Fuzzy Model

The first step in building the fuzzy inference system is to determine the number and nature of the linguistic variables based on expert experience. So the duration of yawning (YD), and PERCLOS are used as crisp input variables while drowsiness level (DL) is used as an output variable. We selected to use this combination of variables as inputs for the FIS due to the limitation of using just one of them, as we mentioned in the introduction section. The input and output variables details and the membership functions of the FIS are described in Table 3.1. And Figure 3.2 shows the proposed FIS model, and Figure 3.3 shows the membership functions of inputs and output.

Table 3.1 Details of inputs and output variables of FIS

Variable	**Type**	**Range**	**Labels**	**Linguistic terms**	**Membership functions**
PERCLOS	Input	[0, 1]	3	Low, medium, high	Triangular, Trapezoidal
YD	Input	[0, 1]	3	Small, medium, long	Triangular, Trapezoidal
DL	Output	[0, 10]	3	Low, medium, high	Triangular, Trapezoidal

We have got a rule base with 18 rules for the proposed FIS that can easily be interpreted. If the driver is tired, the yawning will appear, and its duration will depend on the degree of his tiredness; if it continued for a long time, then the degree of tiredness could be high. At the same, the PERCLOS value increases with the degree of tiredness of the driver. Therefore, depending on these two features, the rule base of the fuzzy logic is specified. Since the PERCLOS feature could be extracted from the left and right eyes, nine rules are built for each one of them. Table 3.2 shows the fuzzy rules used in the proposed model. And Figure 3.4 shows the rule viewer of the proposed fuzzy model.

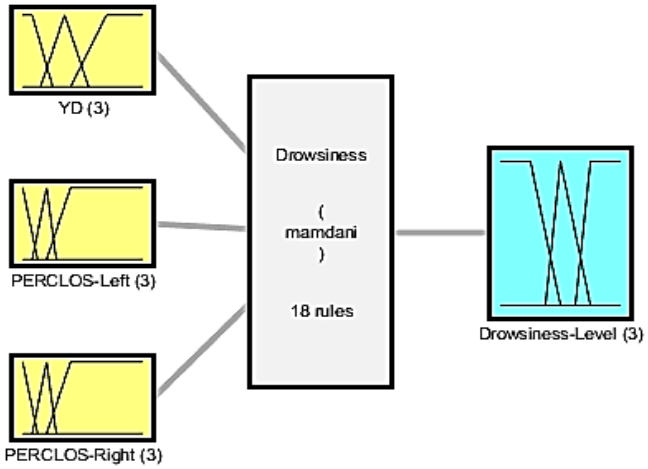


Figure 3.2 The Proposed FIS Model

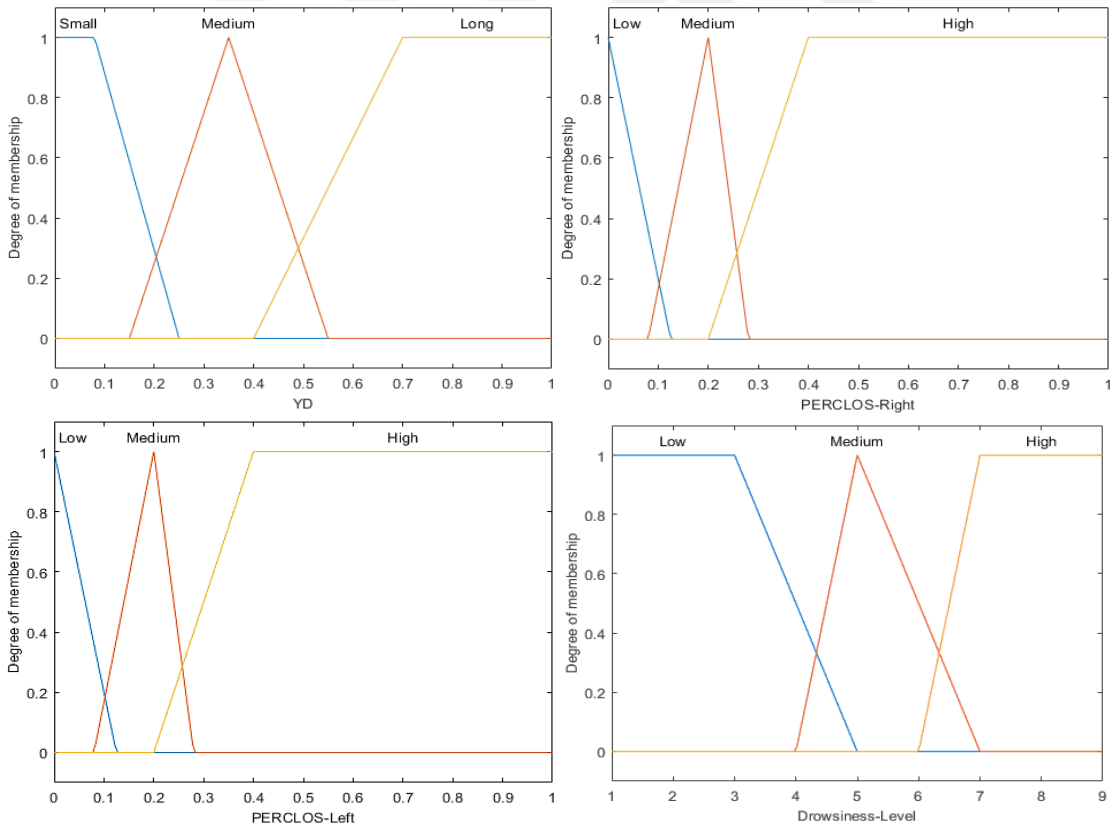


Figure 3.3 The Membership Functions of the three Inputs and the one Output

Table 3.2 The fuzzy rules used in the proposed model.

No.	Rule
1	**If** YD is small & PERCLOS_Left is low **then** Drowsiness-Level is low
2	**If** YD is small & PERCLOS_Left is Medium **then** Drowsiness-Level is low
3	**If** YD is small & PERCLOS_Left is High **then** Drowsiness-Level is Medium
4	**If** YD is Medium & PERCLOS_Left is low **then** Drowsiness-Level is low
5	**If** YD is Medium & PERCLOS_Left is Medium **then** Drowsiness-Level is Medium
6	**If** YD is Medium & PERCLOS_Left is High **then** Drowsiness-Level is High
7	**If** YD is Long & PERCLOS_Left is low **then** Drowsiness-Level is Medium
8	**If** YD is Long & PERCLOS_Left is Medium **then** Drowsiness-Level is High
9	**If** YD is Long & PERCLOS_Left is High **then** Drowsiness-Level is High
10	**If** YD is small & PERCLOS_Right is low **then** Drowsiness-Level is low
11	**If** YD is small & PERCLOS_ Right is Medium **then** Drowsiness-Level is low
12	**If** YD is small & PERCLOS_ Right is High **then** Drowsiness-Level is Medium
13	**If** YD is Medium & PERCLOS_ Right is low **then** Drowsiness-Level is low
14	**If** YD is Medium & PERCLOS_ Right is Medium **then** Drowsiness-Level is Medium
15	**If** YD is Medium & PERCLOS_ Right is High **then** Drowsiness-Level is High
16	**If** YD is Long & PERCLOS_ Right is low **then** Drowsiness-Level is Medium
17	**If** YD is Long & PERCLOS_ Right is Medium **then** Drowsiness-Level is High
18	**If** YD is Long & PERCLOS_ Right is High **then** Drowsiness-Level is High

As mentioned above, any FIS goes through four steps: Fuzzification, Rule Base, Inference Mechanism, and De-fuzzification. The crisp inputs are converted by using the membership functions in the fuzzification stage into linguistic variables. Thus, in the fuzzification stage, we focus on choosing the most suitable membership function. Here we tried to use trapezoidal, triangular, and gaussian MFs, and we found that the trapezoidal and triangular MFs were the most appropriate for this problem.

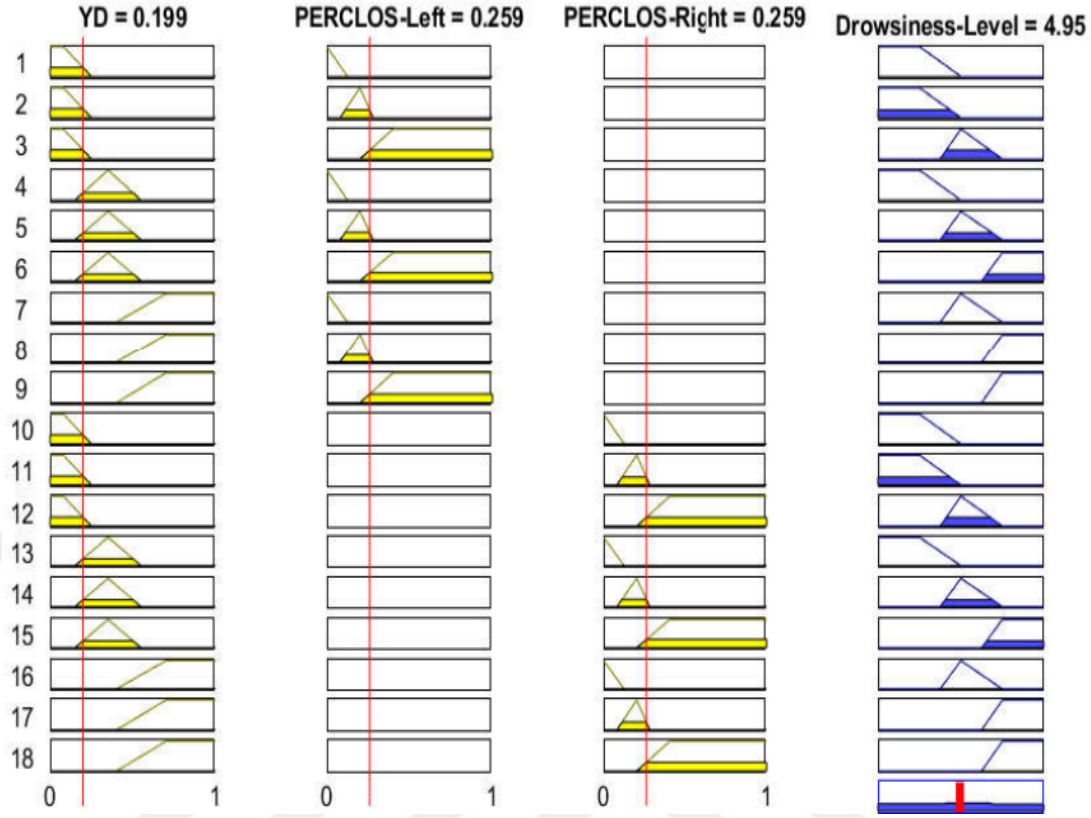


Figure 3.4 Rule viewer of the proposed fuzzy model

In the inference mechanism step, different methods can be utilized to formulate the mapping from a given input to an output, such as Mamdani, Larsen, Tsukamoto, and TSK method [91]. In this work Mamdani method is used for two reasons: it is well suited to human inputs, and it uses fuzzy set as rule consequent.

In the defuzzification step, different methods can be utilized to find the crisp value, such as the Mean of Maximum, Center of Area, and Bisector of Area [91]. In this work, the Center of Area method is used.

CHAPTER 4
EXPERIMENTS AND RESULTS

4.1 Validity

In this section, the validity of the proposed optimization is checked on IEEE CEC 2014 benchmark problems [92]. Consequently, the performance of the proposed algorithms has therefore been compared to that of state-of-the-art algorithms. Table 4.1 shows the average error of the 51 runs in objective function value. The stoping criteria is the same for all algorithms (10^4*Dimension). The dimension used in this comparsion is $(D = 30)$. The configuration of algorithms used in this comparsion is the same as that used in [92] while the following configuration was considered for the proposed algorithms.

1. The initialization within the search space was uniform random.
2. Limit = 100.
3. Population size NP = 100.
4. Number of food sources SN = NP/2.

It is noticed in Table 4.1 that compared to other metaheuristic algorithms, the proposed algorithm MABC produces very competitive results. Indeed, it gives the best results in 14 problems out of 30 while the second best algorithm gives the best results in just 4 problems out of 30. Moreover, Table 4.2 compares the proposed MABC with other algorithms as a paired comparison where the result is shown in the cell which is in row 2 and column 3 (2-0-1) means that MABC is better than CS in two problems, they do not have the same results, and worse in one problem.

The proposed classification models are benchmarked by four regular classification datasets derived from the UCI Machine Learning Repository [93]: XOR, Iris, Balloon, and Breast Cancer. The results of the proposed classification models are compared with ABC, TLABC, and DFSABC_ elite in addition to the results in [94].

Table 4.1 Comparison of average error in objective function value for 30 dimension of CEC 2014 test problems

Problem	GSA [95, 96, 97]	CS [95, 96, 98]	LX-BBO [95, 99]	B-BBO [95, 99]	SOS [95, 96, 100]	RW-GWO [95]	GWO [95]	ABC	MABC
1	1.57E+07	**2.29E+05**	1.01E+07	6.50E+06	1.95E+07	8.02E+06	3.32E+07	1.12E+07	6.18E+06
2	8.89E+03	1.39E+02	5.34E+04	2.36E+04	3.61E+09	2.23E+05	1.01E+09	1.69E+02	**2.97E+01**
3	7.32E+04	1.05E+04	1.64E+04	6.04E+03	2.38E+04	3.16E+02	2.85E+04	7.42E+02	**2.54E+02**
4	3.68E+02	7.07E+01	9.99E+01	1.03E+02	4.14E+02	3.41E+01	1.94E+02	2.82E+01	**1.22E+01**
5	2.00E+01	2.02E+01	**3.06E+00**	3.74E+00	2.07E+01	2.05E+01	2.10E+01	2.04E+01	2.03E+01
6	2.88E+01	2.48E+01	1.69E+01	2.00E+01	2.48E+01	9.84E+00	1.16E+01	1.54E+01	**8.19E+00**
7	1.99E-04	7.45E-02	1.76E-01	7.82E-02	3.38E+01	2.53E-01	7.67E+00	2.51E-05	**2.80E-06**
8	1.40E+02	7.12E+01	5.53E+01	4.71E-01	8.10E+01	4.38E+01	6.50E+01	**0.00E+00**	**0.00E+00**
9	1.65E+02	1.78E+02	7.66E+01	9.11E+01	1.73E+02	6.33E+01	8.54E+01	8.57E+01	**3.33E+01**
10	3.35E+03	2.02E+03	1.26E+04	6.69E+03	2.40E+03	9.61E+02	1.80E+03	1.49E+00	**9.07E-01**
11	4.06E+03	4.49E+03	1.23E+04	6.72E+03	4.48E+03	2.68E+03	2.90E+03	2.21E+03	**1.58E+03**
12	9.10E-02	8.11E-01	**1.11E-02**	1.11E-02	7.37E-01	5.45E-01	2.12E+00	4.33E-01	2.31E-01
13	4.03E-01	4.17E-01	6.55E-01	6.75E-01	6.85E-01	2.80E-01	3.74E-01	2.56E-01	**1.32E-01**
14	2.30E-01	5.18E-01	6.20E-01	3.93E-01	8.39E+00	4.23E-01	7.49E-01	1.98E-01	**8.74E-02**
15	1.26E+01	1.32E+01	1.55E+01	1.88E+01	2.56E+02	8.81E+00	2.06E+01	1.08E+01	**3.84E+00**
16	1.48E+01	1.23E+01	1.08E+01	1.07E+01	1.20E+01	1.03E+01	1.09E+01	1.06E+01	**9.81E+00**
17	7.29E+05	**1.24E+05**	1.46E+06	1.28E+06	5.59E+06	5.71E+05	6.28E+05	2.96E+06	1.59E+06
18	**3.86E+02**	1.42E+03	2.90E+03	8.22E+02	4.86E+05	6.52E+03	5.27E+05	9.21E+03	7.69E+02
19	1.57E+02	1.18E+01	5.19E+03	7.81E+03	4.07E+01	1.14E+01	2.56E+01	7.75E+00	**7.16E+00**
20	8.24E+04	**1.36E+02**	2.61E+04	1.63E+04	1.59E+04	6.27E+02	1.31E+04	4.82E+03	1.03E+03
21	1.79E+05	**1.67E+03**	1.11E+06	1.23E+06	7.86E+05	2.58E+05	4.97E+05	4.79E+05	2.35E+05
22	9.51E+02	3.11E+02	1.88E+03	**1.68E+02**	5.45E+02	2.08E+02	2.50E+02	2.92E+02	2.13E+02
23	**2.00E+02**	3.44E+02	4.11E+02	3.43E+02	3.42E+02	3.15E+02	3.28E+02	3.15E+02	3.15E+02
24	2.01E+02	2.21E+02	1.48E+04	3.41E+04	2.36E+02	**2.00E+02**	2.00E+02	2.27E+02	2.16E+02
25	**2.00E+02**	2.09E+02	5.29E+02	6.54E+02	2.15E+02	2.04E+02	2.11E+02	2.08E+02	2.07E+02
26	1.69E+02	1.01E+02	**2.13E+00**	3.64E+01	1.01E+02	1.00E+02	1.00E+02	1.00E+02	1.00E+02
27	7.69E+02	4.18E+02	**1.96E+02**	3.05E+02	4.96E+02	4.09E+02	4.33E+02	4.03E+02	4.04E+02
28	7.65E+02	9.13E+02	1.94E+03	2.12E+03	1.32E+03	**4.34E+02**	9.14E+02	9.62E+02	8.02E+02
29	**2.00E+02**	2.01E+02	1.98E+07	3.09E+07	2.16E+04	2.14E+02	2.90E+05	1.12E+03	1.02E+03
30	2.31E+04	3.59E+03	6.96E+06	1.38E+07	3.73E+04	**6.69E+02**	2.98E+04	4.56E+03	2.37E+03

The datasets specifications are shown in Table 4.3. These classification datasets have been carefully selected with various training/test samples and complexity levels to check the efficiency of the proposed classification models effectively. Each model is

trained ten times on the same dataset to obtain the results. Average (AVE) and standard deviation (STD) of the best MSEs obtained in the final iteration by algorithms were provided as statistical tests.

Table 4.2 Comparison of the number of success of MABC against the other algorithms on the CEC 2014 benchmark.

MABC	GSA	CS	LX-BBO	B-BBO	SOS	RW-GWO	GWO	ABC
G1	3-0-0	2-0-1	3-0-0	3-0-0	3-0-0	3-0-0	3-0-0	3-0-0
G2	11-0-2	12-0-1	11-0-2	11-0-2	13-0-0	13-0-0	13-0-0	12-1-0
G3	3-0-3	3-0-3	5-0-1	4-0-2	6-0-0	3-0-3	5-0-1	6-0-0
G4	3-0-5	7-0-1	6-0-2	6-0-2	8-0-0	1-2-5	6-1-1	5-2-1
Total	20-0-10	24-0-6	25-0-5	24-0-6	30-0-0	20-2-8	27-1-2	26-3-1

Since the MSE is the objective function, the problem is a minimization problem. This means a lower average and standard deviation from MSE implies better results. Moreover, for each of the algorithms during the ten training times, the classification rates are reported as additional comparison metrics. These results are presented in Table 4.4.

The emphasis of this work is not on determining the optimal number of hidden nodes, so we considered that the number of hidden nodes is equal to 2 ×N + 1 where N equals the number of dataset's features. The larger the neural network, the more weights, and biases the system would have. The training process is, therefore, become more difficult. Table 4.3 presents the structure of each MLP used for each dataset. The configuration of GWO, PSO, GA, ACO, ES, and PBIL algorithms are the same as in [94] while the following configuration was considered for the other models in the experiments:

5. The initialization within the search space was uniform random.
6. Number of iteration is 500 for Iris and Breast Cancer datasets.
7. Number of iteration is 125 for XOR and Balloon datasets.
8. Number of iteration is 300 for Yawning and Eye State datasets.
9. Limit = 100.

10. Population size NP = 100.
11. Number of food sources SN = NP/2.
12. The range search was inside the interval [-10, 10].

It is noted that the maximum number of iteration is the stopping criterion for all algorithms. Besides, during the experiments, the number of searching agents is fixed. Increasing the number of iterations and population size, obviously, would improve the absolute classification rate, but keeping them fixed for all algorithms was of interest to the comparative performance purpose among algorithms. The statistical results of training algorithms on the four datasets are presented in Table 4.4.

Table 4.3 Classification datasets and MLP structure

Datasets	**Number of attributes**	**Number of training samples**	**Number of test samples**	**Number of classes**	**MLP structure**	**Problem variables**
XOR	3	8	8 as training samples	2	3-7-1	36
Balloon	4	16	16 as training samples	2	4-9-1	55
Iris	4	150	150 as training samples	3	4-9-3	75
Breast cancer	9	599	100	2	9-19-1	209

The results observing from training the proposed algorithms on the XOR dataset show a similar classification rate, which is 100%, for the six proposed algorithms in the ten runs. This fact shows the efficiency of the proposed algorithms in avoiding the local optima and their stability. Nevertheless, the MSE average and standard deviation of the ten runs are different for algorithms, and DFSABC_elite shows the best results.

As a result of the simplicity of the Balloon dataset, a similar classification rate for all algorithms is observed in the results. Nevertheless, the MSE average and standard deviation of the ten runs are different for algorithms, and TLABC shows the best results.

Table 4.4 Classification benchmarks results

XOR Dataset			**Balloon Dataset**		
Algorithm	MSE (AVE ±STD)	Accuracy %	**Algorithm**	MSE (AVE ±STD)	Accuracy %
GWO[93]	0.009410 ± 029500	100	**GWO[93]**	9.38e-15 ±2.81e-14	100
PSO[93]	0.084050 ± 0.035945	37.5	**PSO[93]**	0.000585 ±0.000749	100
GA[93]	0.000181 ±0.000413	100	**GA[93]**	5.08e-24 ±1.06e-23	100
ACO[93]	0.180328 ±0.025268	62.5	**ACO[93]**	0.004854 ±0.007760	100
ES[93]	0.118739 ±0.011574	62.5	**ES[93]**	0.019055 ±0.170260	100
PBIL[93]	0.030228 ±0.039668	62.5	**PBIL[93]**	2.49e-05 ±5.27e-05	100
ABC	0.000897 ± 0.001582	100	**ABC**	3.41e-10 ± 5.50e-10	100
MABC	0.000611 ± 0.000599	100	**MABC**	4.52e-09 ± 7.46e-09	100
TLABC	0.002624 ± 0.005799	100	**TLABC**	2.04e-16 ± 6.45e-16	100
MTLABC	6.33e-05 ± 9.04e-05	100	**MTLABC**	2.64e-24 ± 8.32e-24	100
DFSABC_elite	**3.14e-05** ± 8.39e-05	100	**DFSABC_elite**	**6.43e-29** ± 1.45e-28	100
MDFSABC	4.95e-05 ± 8.34e-05	100	**MDFSABC**	3.16e-26 ± 9.99e-26	100
Cancer Dataset			**Iris Dataset**		
Algorithm	MSE (AVE ±STD)	Accuracy %	**Algorithm**	MSE (AVE ±STD)	Accuracy %
GWO[93]	**0.0012 ± 7.4498e-05**	99	**GWO[93]**	0.0229 ± 0.0032	91.333
PSO[93]	0.034881 ± 0.002472	11	**PSO[93]**	0.228680 ± 0.057235	37.33
GA[93]	0.003026 ± 0.001500	98	**GA[93]**	0.089912 ± 0.123638	89.33
ACO[93]	0.013510 ± 0.002137	40	**ACO[93]**	0.405979 ± 0.053775	32.66
ES[93]	0.040320 ± 0.002470	06	**ES[93]**	0.314340 ± 0.052142	46.66
PBIL[93]	0.032009 ± 0.003065	07	**PBIL[93]**	0.116067 ± 0.036355	86.66
ABC	0.016732 ± 0.000819	97	**ABC**	0.427624 ± 0.125466	42
MABC	0.017045 ± 0.000990	97	**MABC**	0.354687 ± 0.074696	41.33
TLABC	0.016078 ± 0.002626	99	**TLABC**	**0.012855** ± 0.002825	98
MTLABC	0.013824 ± 0.003006	99	**MTLABC**	0.013560 ± 0.004860	97.33
DFSABC_elite	0.010956 ± 0.001877	97	**DFSABC_elite**	0.054176 ± 0.018267	82.67
MDFSABC	0.011517 ± 0.001251	96	**MDFSABC**	0.051960 ± 0.014941	80.66

The Iris dataset is one of the most common benchmark datasets in machine learning, and it is more complicated than XOR and Balloon datasets. However, the discussed algorithms show high classification rates, and GWO, TLABC, and the proposed MTLABC show the best results.

Breast cancer is the most challenging dataset in terms of weight, bias, and features compared to the other three datasets. Consequently, its classification rates are different for algorithms, and TLABC shows the best results. However, the results in this table show the ability and efficiency of the proposed algorithms in training MLPs.

4.2 Dataset description

The dataset used in this study is called YawDD, which their details are described in [10]. This dataset is in video formats and was collected from an almost practical driving scenario. This dataset consists of two groups: in the first group of videos, the camera was installed on the dash while in the second group, the camera was installed under the front mirror. Since we will use the Viola and Jone's method to extract the face region from the images, the first group of the dataset will give better results. Because the face detector of Viola and Jone is trained to extract faces tilted to ±45 degrees in the frontal view and to around ±15 degrees in the horizontal view [10]. As a result, the dash dataset, which consists of 16 male videos and 13 female videos, is used in this study. Figure 4.1 shows some images of the dataset in different illumination conditions.

Since there is not expected rate of variations in every five successive frames, so we suggested analyzing every fifth frame of the video sequence monitors. Furthermore, if every frame is tested for drowsiness, the real response time of the system would also be at stake. Thus, the total number of obtained frames is 11824 frames.

Since our model uses eyes and mouth features, we proposed two classification models one of them for eye state detection while the other for yawning detection. Therefore, we adjusted the dataset to go with our models as the following. In fact, after we have processed and prepared the data of each model, each model was trained and tested separately. After that, the FIS model was proposed and tested to detect the drowsiness.

Since YawDD dataset was proposed just for yawning detection purposes; we used other datasets for the eye state detection model. This dataset consists of closed and open eye images for the left and right eyes. The number of closed eye images is 1193, while the number of open eye images is 1232. Thus, the total number of images for the left eye is 2423, and the same number for the right eye. Out of them, 2059 random images are used for training (85%), and 364 random images are used for testing (15%). This dataset published by [101].



Figure 4.1 Examples of YawDD dataset

After that, the gray co-occurrence matrix (GLCM) for each frame was built and the statistical metrics are extracted from it. In this work, the parameters' values used in building the GLCM are d=1 and θ=0^0. And the statistical metrics extracted in this work are 22 metrics and described in Table 4.5 [38, 102].

$$\sum_i g_i = \sum_i^N g_i \quad (4.1)$$

$$g_y(j) = \sum_i^N g_{ij} \quad (4.2)$$

$$g_{x+y}(k) = \sum_i \sum_j g_{ij}, \quad k = (i+j), \text{k= 2, 3, …, 2N} \quad (4.3)$$

$$g_{x-y}(k) = \sum_i \sum_j g_{ij}, \quad \text{k} = |i-j|, \text{k= 0, 1, …, N – 1} \quad (4.4)$$

$$IX = -\sum_i g_x(i)\, log\{ g_x(i)\} \quad (4.5)$$

$$IY = -\sum_i g_y(i)\, log\{ g_y(i)\} \quad (4.6)$$

$$IXY = -\sum_i \sum_j g_{ij}\, log\, \{g_{ij}\} \quad (4.7)$$

$$IXY1 = -\sum_i \sum_j g_{ij}\, log\, \{g_y(j)\ g_x(j)\} \quad (4.8)$$

$$IXY1 = -\sum_i \sum_j\ g_y(j)\ g_x(j)\ log\ \{g_y(j)\ g_x(j)\} \quad (4.9)$$

The dataset was prepared for FIS, as well as the following. We segmented the videos into 4 seconds length segments. Each segment was labelled by one of two labels: normal and drowsiness. In each segment, there are 24 frames. From each frame left eye, right eye and yawning state are detected. Then, three variables are calculated at the end of each segment: YD, PERCLOS for the right eye, and PERCLOS for the left eye. These variables are used as crisp input variables of the proposed FIS. And the equations used to calculate them are formulated in Eq. 4.10, Eq. 4.11.

$$PERCLOS = \frac{T_1+T_2+\cdots+T_n}{T} \quad (4.10)$$

$$YD = \frac{Y_1+Y_2+\cdots+Y_n}{Y} \quad (4.11)$$

Where $\boldsymbol{T_1, T_2, \ldots, T_n}$ are the frames detected as close eyes, and $\boldsymbol{T}$ is the total number of frames inside the segment, which equals 24. $\boldsymbol{Y_1, Y_2, \ldots, Y_n}$ are the frames detected as yawning, and $\boldsymbol{Y}$ is the total number of frames inside the segment, which equals 24.

Table 4.5 Summary of 22 features extracted from GLCM

ID	Feature name	Formula
F1	**Uniformity / Energy / Angular Second Moment**	$f_1 = \sum_i \sum_j g_{ij}^2$
F2	**Entropy**	$f_2 = -\sum_i \sum_j g_{ij}\, log\{g_{ij}\}$
F3	**Dissimilarity**	$f_3 = \sum_i \sum_j \lvert i-j \rvert\, g_{ij}$
F4	**Contrast / Inertia**	$f_4 = \sum_i \sum_j (i-j)^2 g_{ij}$
F5	**Homogeneity MATLAB**	$f_5 = \sum_i \sum_j \frac{g_{ij}}{1+\lvert i-j \rvert}$
F6	**correlation**	$f_6 = \sum_i \sum_j \frac{i\, j\ g_{ij} - \mu_x \mu_y}{\sigma_x \sigma_y}$
F7	**Homogeneity / Inverse difference moment**	$f_7 = \sum_i \sum_j \frac{g_{ij}}{1+(i-j)^2}$
F8	**Autocorrelation**	$f_8 = \sum_i \sum_j i\, j\ g_{ij}$
F9	**Cluster Shade**	$f_8 = \sum_i \sum_j \left(i + j - \mu_x - \mu_y\right)^3 g_{ij}$
F10	**Cluster Prominence**	$f_9 = \sum_i \sum_j \left(i + j - \mu_x - \mu_y\right)^4 g_{ij}$
F11	**Maximum probability**	$f_9 = \max g_{ij}$

F12	**Sum of Squares (Variance)**	$f_{12} = \sum_i \sum_j (i - \mu)^2 g_{ij}$
F13	**Sum Average**	$f_{13} = \sum_{i=2}^{2N} i \, g_{x+y}(i)$
F14	**Sum Variance**	$f_{14} = \sum_{i=2}^{2N} (i - f_{15})^2 \, g_{x+y}(i)$
F15	**Sum Entropy**	$f_{15} = -\sum_{i=2}^{2N} g_{x+y}(i) \, log\{ g_{x+y}(i)\}$
F16	**Difference variance**	$f_{16} = \sum_{i=2}^{2N} i^2 \, g_{x-y}(i)$
F17	**Difference entropy**	$f_{17} = -\sum_{i=2}^{2N} g_{x-y}(i) \, log\{ g_{x-y}(i)\}$
F18	**Information measures of correlation (1)**	$f_{18} = \frac{IXY - IXY1}{\max\{IX, IY\}}$
F19	**Information measures of correlation (2)**	$f_{19} = (1 - \exp(-2.0[IXY2 - IXY]))^{0.5}$
F20	**Correlation MATLAB**	$f_{20} = \sum_i \sum_j \frac{g_{ij} \, (\mu_x - i \,) \, (\mu_y - j)}{\sigma_x \sigma_y}$
F21	**Inverse difference normalized**	$f_{21} = \sum_i \sum_j \frac{g_{ij}}{1 + \frac{\lvert i - j \rvert}{N}}$
F22	**Inverse difference moment normalized**	$f_{22} = \sum_i \sum_j \frac{g_{ij}}{1 + \frac{(i - j)^2}{N}}$

When the attributes of the dataset are in various ranges, the small values will be neglected compared to the large values. Moreover, the training of MLP will take more training time; hence gradients will take a long time before the global/local minima can finally be reached. Consequently, normalization of the dataset is a fundamental step for MLP training when the attributes of the dataset are in various ranges. The normalization equation used in this work is formulated in Eq. 4.12. This formula maps the dataset in the interval of [0, 1].

$$z_i^j = \frac{x_i^j - x_{min}^j}{x_{max}^j - x_{min}^j} \quad (4.12)$$

4.3 Yawning Detection Results

In this section, the performance of the proposed classification models is evaluated. To do this, the classification rate (accuracy), precision, recall, F1 score, mean square error, and convergence rate metrics are used.

Classification rate: it is the basic performance metric used in many models to compare between them. This metric tells us how the model correctly classifies the driver's image, whether it is yawning or not. Its value is between 0 and 100, where the 100 means the model has the best classification rate and can correctly classify all drivers' images. The formula used to calculate the classification rate of the model is given in equation 4.13.

$$Classification\ Rate = \frac{TP+TN}{TP+TN+FP+FN} \quad (4.13)$$

Where TP is the number of correctly predicted yawning frames, TN is the number of correctly predicted not yawning frames, FP is the number of wrongly predicted yawning frames, and FN is the number of wrongly not predicted yawning frames.

The recall is a performance metric that tells us the proportion of predicted yawning states that may actually be yawny. Thus, recall tells us about the missed positive classified. Moreover, the complement of the recall is the miss rate, which defines as the proportion of real yawning states being classified as not yawning states. The formulas used to calculate the recall and miss rate of the model are given in equations 4.14 and 4.15, respectively.

$$Recall = \frac{TP}{TP+FN} \quad (4.14)$$

$$Miss\ Rate = 1 - Recall \quad (4.15)$$

Precision is a performance metric that tells us the proportion of predicted not yawning states that may actually be not yawny. Thus, precision calculates the accuracy of the minority (yawning) class. Moreover, the complement of the precision is the false alarm rate, which defines as the proportion of real, not yawning states being classified as yawning states. The formulas used to calculate the precision and false alarm rate of the model are given in equations 4.16 and 4.17, respectively.

$$Precision = \frac{TP}{TP+FP} \quad (4.16)$$

$$False\ Alarm\ Rate = 1 - Precision \quad (4.17)$$

F1-score gives a metric value, which balances between precision and recall in one number. Neither precision nor recall alone shows the real performance of the model. The model can have terrible precision with excellent recall or vice versa. But F1-score provides a single value that concerns on both. Therefore, whenever the F1-score is high, the model performance is high too. The formula used to calculate the F1-score of the model is given in equation 4.18.

$$F1\ Score = \frac{2*Precision*Recall}{Precision+Recall} \quad (4.18)$$

Since the objective of the MLP training process using meta-heuristics algorithms is to minimize the mean of MSE, we used MSE as a performance metric to compare the proposed models. The model with a lower value of MSE is trained better than the other models. Moreover, the convergence rate to the optimal values of weights during training is shown as another performance metric. Figure 4.2 displays the convergence curves of the six models for the yawning detection dataset. The convergence curves show how the average values of the objective function (MSE) change with respect to the number of iterations. It is noticed that the best convergence model is MDFSABC, which also has the best MSE value, the best accuracy, the best precession, and the best F1-score value. Consequently, the MDFSABC model is chosen to be used for the yawning detection feature in the final drowsiness detection model.

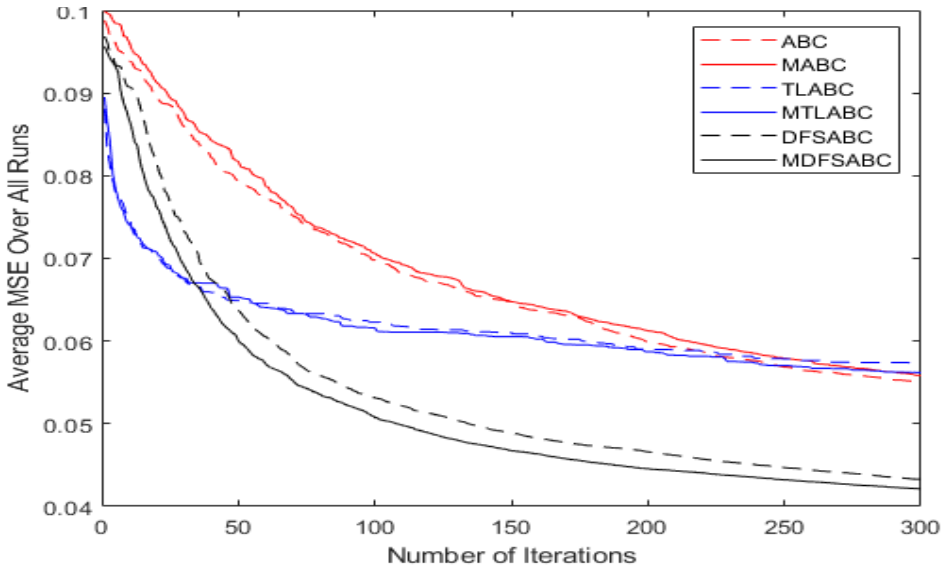


Figure 4.2 Convergence curves for yawning detection models.

Table 4.6 MSE and accuracy of yawning detection models

Classification Model	**MSE (AVE ±STD)**	**Classification rate %**	
		Best	**Mean**
ABC	0.0550 ± 0.0041	94.5126	93.1805
MABC	0.0558 ± 0.0040	94.3189	93.1827
TLABC	0.0608 ± 0.0035	94.0607	92.8642
MTLABC	0.0579 ± 0.0034	94.4480	93.1784
DFSABC_elite	0.0433 ± 0.0016	95.0291	94.4287
MDFSABC	0.0421 ± 0.0014	95.2227	94.5664

Table 4.6 shows the average and standard deviation of the best scores of the MSE metric obtained from the models over 30 runs. Moreover, the table shows the average and best accuracy obtained from the test dataset. It is important to mention that different threshold values were tested, and the values shown in Table 4.6 are the values of the

best threshold. Figure 4.3 displays the effect of changing the threshold on the classification accuracy.

Table 4.7 shows the average and best values of the F1 score obtained from the test dataset of all models over 30 runs. It is important to mention that different threshold values were tested, and the values shown in Figure 4.4 are the values of the average of 30 runs with different values of thresholds. It is noticed in Table 4.7 that the MDFSABC model has the best F1 score values. After getting the run number and threshold value that gave the best F1 score, we added their corresponding values of precision and recall to the table.

Table 4.7 Precision, Recall, and F1 Score of yawning detection models

Classification Model	**Precision**	**Recall**	**F1 Score**	
	Best	**Best**	**Best**	**Mean**
ABC	0.74615	0.6510	0.6953	0.5981
MABC	0.75630	0.6040	0.6716	0.5906
TLABC	0.64336	0.6174	0.6301	0.5634
MTLABC	0.63030	0.6980	0.6624	0.5888
DFSABC_elite	0.76866	0.6913	0.7279	0.6797
MDFSABC	**0.78195**	**0.6980**	**0.7376**	**0.6876**

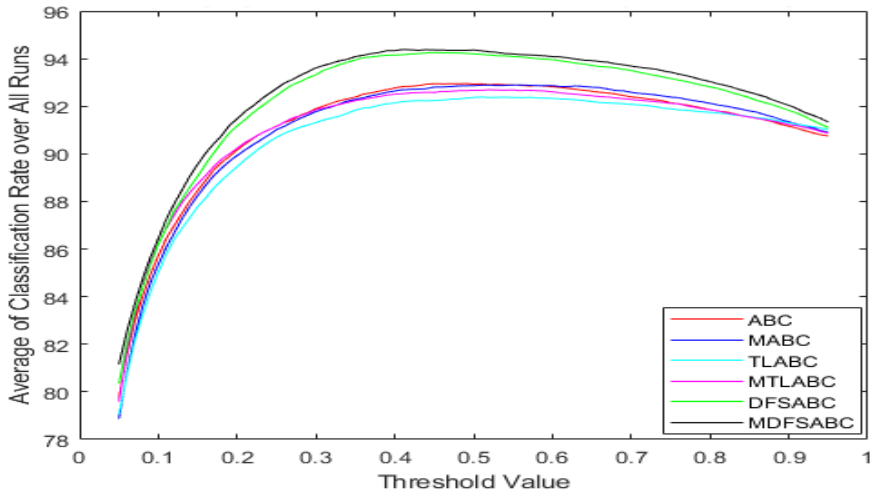


Figure 4.3 Effect of threshold on accuracy of yawning detection models

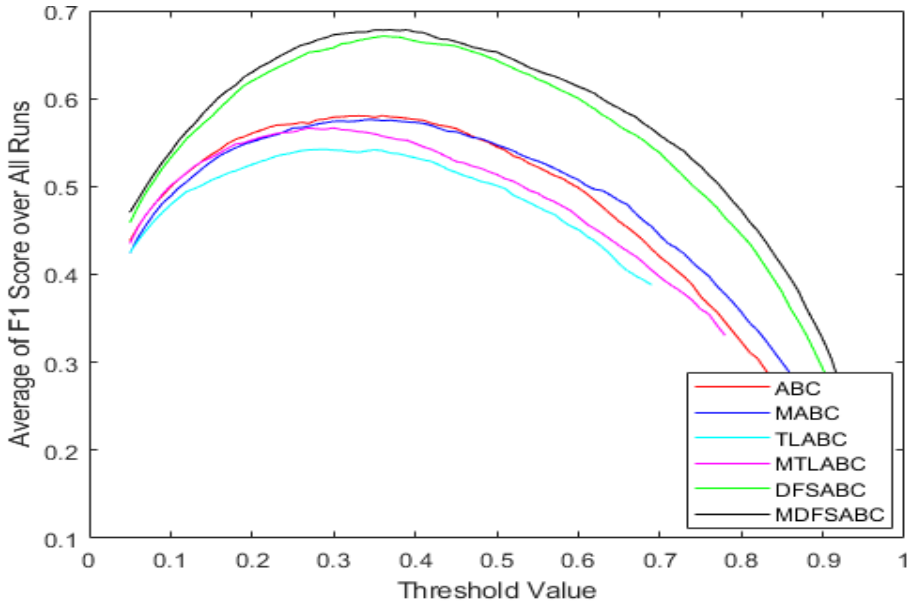


Figure 4.4 Effect of threshold on F1 score of yawning detection models.

4.4 Right Eye State Detection Results

Figure 4.5 displays the convergence curves of the six models for the right eye detection dataset. The convergence curves show how the average values of the objective function (MSE) change with respect to the number of iterations. It is noticed that the best convergence model is MDFSABC, while TLABC has the best F1-score value. Consequently, the TLABC model is chosen to be used for the right eye detection feature in the final drowsiness detection model.

Table 4.8 shows the average and standard deviation of the best scores of the MSE metric obtained from the models over 30 runs. Moreover, the table shows the average and best accuracy obtained from the test dataset. It is important to mention that different threshold values were tested, and the values shown in Table 4.8 are the values of the best threshold. Figure 4.6 displays the effect of changing the threshold on the classification accuracy.

Table 4.8 MSE and accuracy of right eye detection models

Classification Model	MSE (AVE ±STD)	Classification rate %	
		Best	Mean
ABC	0.1990 ± 0.0053	77.1350	74.8393
MABC	0.1965 ± 0.0056	77.4105	75.2158
TLABC	0.1946 ± 0.0072	77.9614	75.1423
MTLABC	0.1842 ± 0.0049	**78.5124**	**76.6850**
DFSABC_elite	0.1816 ± 0.0047	77.4105	75.6933
MDFSABC	**0.1805 ± 0.0046**	77.6860	76.0331

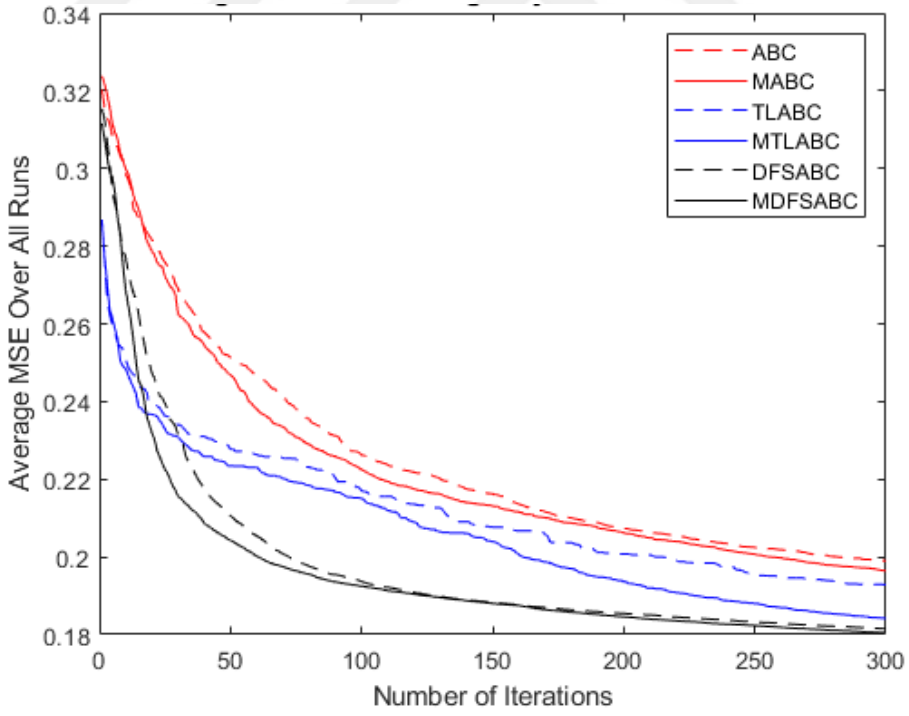


Figure 4.5 Convergence curves for right eye detection models

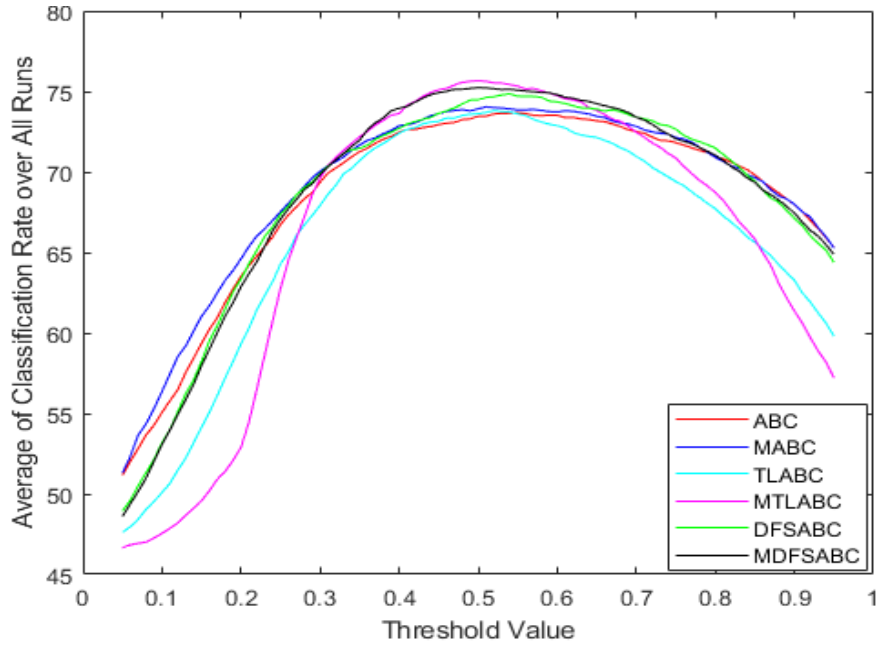


Figure 4.6 Effect of threshold on accuracy of right eye detection models

Table 4.9 shows the average and best values of the F1 score obtained from the test dataset of all models over 30 runs. It is important to mention that different threshold values were tested, and the values shown in Figure 4.7 are the values of the average of 30 runs with different values of thresholds. It is noticed in Table 4.9 that the TLABC model has the best F1 score values. After getting the run number and threshold value that gave the best F1 score, we added their corresponding values of precision and recall to the table.

Table 4.9 Precision, Recall, and F1 Score of right eye detection models

Classification Model	**Precision**	**Recall**	**F1 Score**	
	Best	**Best**	**Best**	**Mean**
ABC	0.7219	0.7988	0.7584	0.7214
MABC	0.6827	0.8402	0.7533	0.7290
TLABC	0.7322	**0.7929**	**0.7614**	0.7303
MTLABC	0.7416	0.7811	0.7608	**0.7391**
DFSABC_elite	0.7135	0.7811	0.7458	0.7292
MDFSABC	**0.7576**	0.7396	0.7485	0.7343

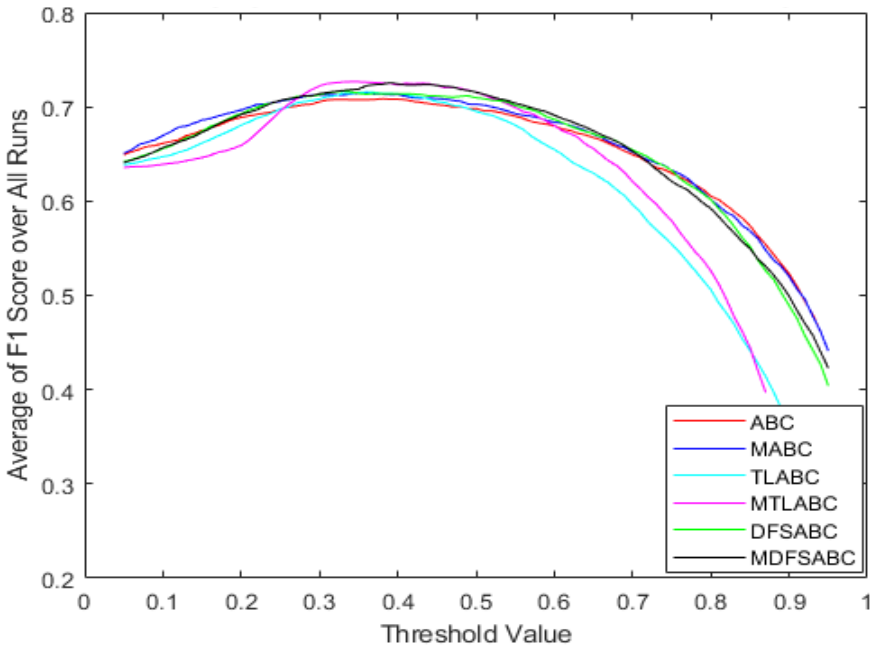


Figure 4.7 Effect of threshold on F1 score of right eye detection models

4.5 Left Eye State Detection Results

Figure 4.8 displays the convergence curves of the six models for the left eye detection dataset. The convergence curves show how the average values of the objective function (MSE) change with respect to the number of iterations. It is noticed MDFSABC has the best convergence curve. However, MTLABC has the best F1-score value. Consequently, the MTLABC model is chosen to be used for the left eye detection feature in the final drowsiness detection model.

Table 4.10 shows the average and standard deviation of the best scores of the MSE metric obtained from the models over 30 runs. Moreover, the table shows the average and best accuracy obtained from the test dataset. It is important to mention that different threshold values were tested, and the values shown in Table 4.10 are the values of the best threshold. Figure 4.9 displays the effect of changing the threshold on the classification accuracy.

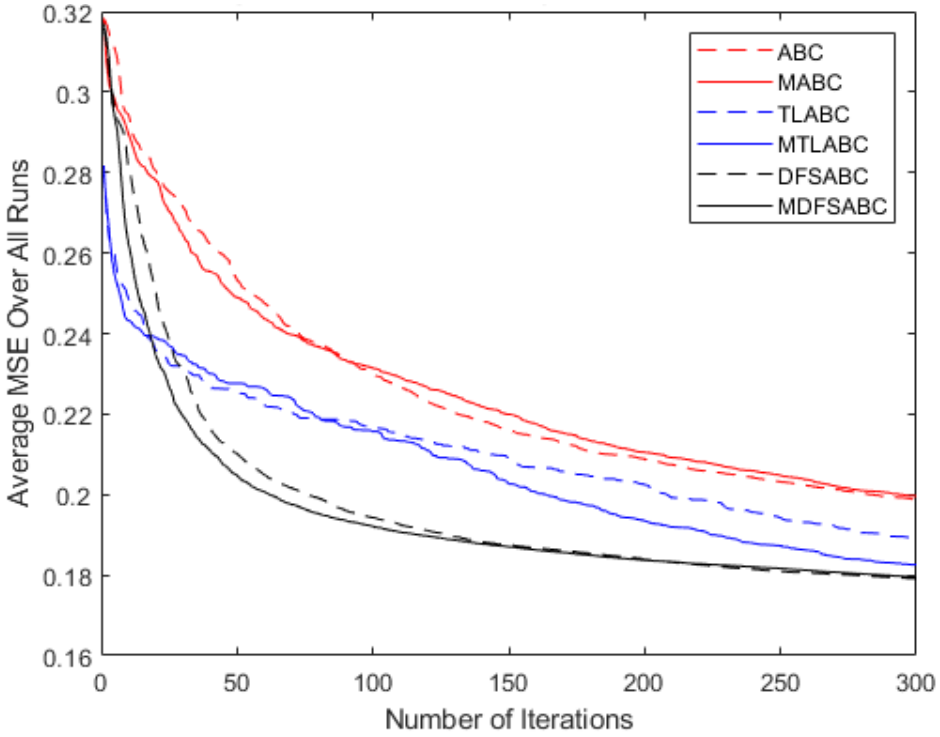


Figure 4.8 Convergence curves for left eye detection models

Table 4.10 MSE and accuracy of left eye detection models

Classification Model	**MSE (AVE ±STD)**	**Classification rate %**	
		Best	**Mean**
ABC	0.1987 ± 0.0048	73.3516	70.1923
MABC	0.1996 ± 0.0054	73.0769	69.9908
TLABC	0.1901 ± 0.0079	72.5275	70.1374
MTLABC	0.1826 ± 0.0039	72.5275	71.0989
DFSABC_elite	**0.1791 ± 0.0044**	73.0769	**71.1996**
MDFSABC	0.1795 ± 0.0050	**73.3516**	70.8700

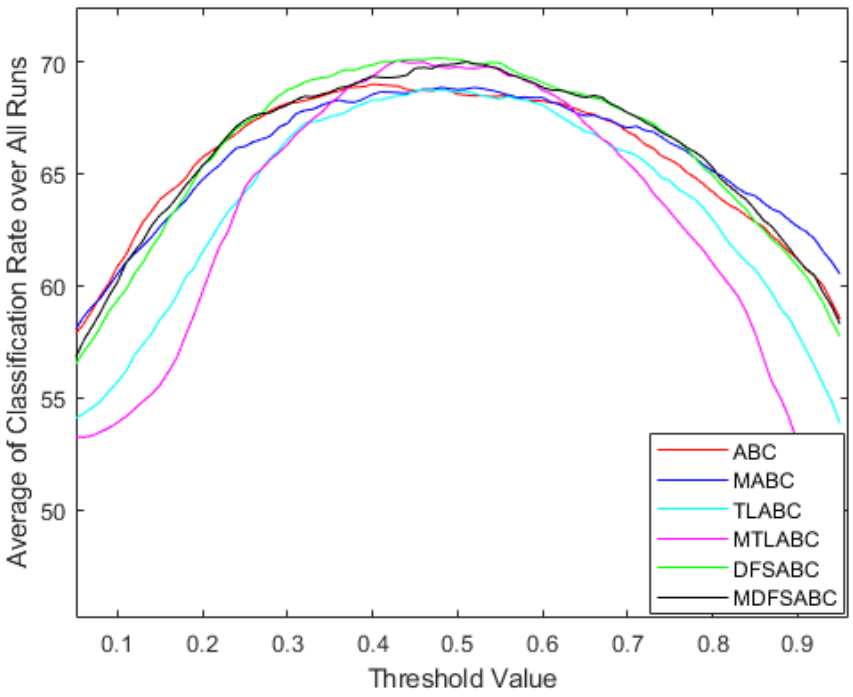


Figure 4.9 Effect of threshold on accuracy of left eye detection models

Table 4.11 Precision, Recall, and F1 Score of left eye detection models

Classification Model	Precision	Recall	F1 Score	
	Best	Best	Best	Mean
ABC	**0.7230**	0.7938	0.7568	0.7334
MABC	0.6602	0.8711	0.7511	0.7288
TLABC	0.6880	0.8299	0.7523	0.7355
MTLABC	0.6667	0.8866	**0.7611**	0.7378
DFSABC_elite	0.6554	**0.9021**	0.7592	**0.7416**
MDFSABC	0.6805	0.8454	0.7540	0.7394

Table 4.11 shows the average and best values of the F1 score obtained from the test dataset of all models over 30 runs. It is important to mention that different threshold values were tested, and the values shown in Figure 4.10 are the values of the average of 30 runs with different values of thresholds. It is noticed in Table 4.11 that the DFSABC

model has the best F1 score values. After getting the run number and threshold value that gave the best F1 `1score, we added their corresponding values of precision and recall to the table.

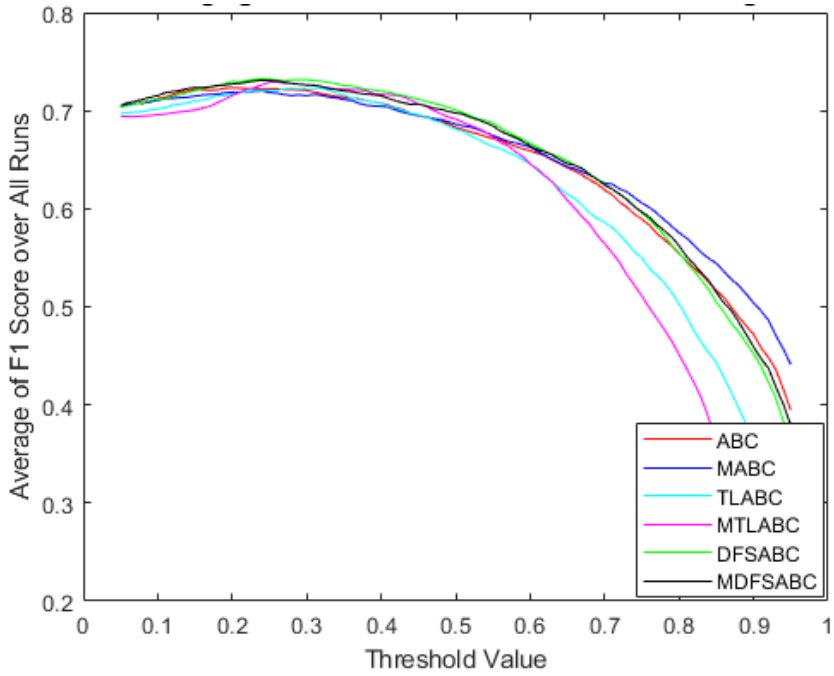


Figure 4.10 Effect of threshold on F1 score of left eye detection models

4.6 Drowsiness Detection Results

As we mentioned above, our model divides the work into three main steps. Each step needs to be tested independently. The first step is the detection of mouth and eyes by using Viola and Jones method. The second step is the detection of yawning and eyes state. The third and final step is the detection of drowsiness level. Table 4.12 shows the accuracy obtained from the first step.

Table 4.12 The accuracy of mouth and eyes detection

Detected part	**Accuracy%**
Mouth	87.18
Left Eye	89.27
Right Eye	84.48

As we mentioned above, we extracted from every five respective frames, just one frame to decrease the time of processing. Thus, the total number of obtained frames is 11824 frames. Out of them, there are 1516 frames could not detect the mouth correctly. So the total number of frames used in training and testing is 10323 (1076 yawning frames and 9247 normal frames). The accuracy of mouth detection is 87.18 %. The training dataset used to train the yawning detection model is 8774 frames (85%) and the rest frames used to test the yawning detection models.

The second step is tested in the previous sections in this chapter, while in this section, the third step is tested as the following: the FIS model is tested on 6 videos: two of them with glasses and four without glasses. Each video is divided into small segments; each segment is 4 seconds in length. Thus, each segment is labelled with a degree of drowsiness, according to Karolinska Sleepiness Scale (KSS) [103]. After that, each segment classified using the proposed FIS. Consequently, the accuracy of the classification of the overall system is 77.19%.

4.7 Statistical Results

It is shown in the tables above that the results of the models are so close to each other. Consequently, some statistical tests should be applied to prove that there is a significant difference between them. Thus, two statistical tests are applied to the results to check whether there is a significant difference between the models or not. These two statistical tests are called the Kruskal-Wallis test and the Wilcoxon Sum Rank test.

In Kruskal-Wallis test, the null hypothesis, Ho, is that the medians of the proposed models are equal (the results obtained from them come from the same distribution) whilst the alternative hypothesis, H1, is that their medians are different. Therefore, rejecting the null hypothesis in favour of the alternative demonstrates that the distinct models are different from each other. In addition to hypotheses, the significance level of the test, α, is defined as the probability of rejecting the null hypothesis while it is, in fact, true. Accordingly, as the significant level goes down, it becomes more difficult to reject the null hypothesis. Thus the smallest significance level at which we still can reject the null hypothesis is an important factor, which is known as the p-value. The p-value indicates the level of confidence when rejecting the null hypothesis. The smaller the p-value is, the more confident we reject the null hypothesis. Moreover, when the p-

value for some sample is less than the significant level of the test, the null hypothesis gets rejected by that test. We applied the Kruskal-Wallis test to the three datasets individually, and Table 4.12 shows the p-values along with the chi-square ($\chi 2$) values of each of them. From a $\chi 2$-distribution, the 0.95 quantile (the critical value at significant level 0.05) of $\chi 2$-distribution with (6-1) degrees of freedom is 11.07, and all the $\chi 2$ values in Table 4.13 are significantly greater than this critical value. That means the null hypothesis is confidently rejected, and the proposed models are proved different. Likewise, the very small p-values mean that we can reject the null hypothesis with high confidence.

Table 4.13 Kruskal-Wallis Test Results of The Yawning, Left Eye, and Right Eye datasets

Dataset	**Chi**	**p-value**	**α=0.05**
YawDD	133.6346	4.02736e-27	Yes
Left Eye	126.0776	1.6168e-25	Yes
Right Eye	120.8342	2.0895e-24	Yes

Wilcoxon Sum Rank test was used to validate the results statistically and show the existence of a significant difference between the proposed models with each other. If the p-value of this test is less than 0.1, then there, indeed, exists a statistically significant difference between the two algorithms in which the test compared. Otherwise, the test fails to prove any significant difference between the two algorithms. Table 4.14 shows the p-values of the multiple comparisons using the Wilcoxon Sum Rank test. It is obvious that the p-values are much smaller than 0.1 in most cases such that the significant difference between each pair of models is demonstrated.

Table 4.14 Wilcoxon Sum Rank Test Results of The Three Dataset

Classificatinn Model	Classification Model	YawDD			Left Eye			Right Eye		
		P-value	α=0.05	α=0.1	P-value	α=0.05	α=0.1	P-value	α=0.05	α=0.1
ABC	MABC	0.62040	No	No	0.41191	No	No	0.18576	No	No
ABC	TLABC	2.67e-06	Yes	Yes	3.36e-05	Yes	Yes	0.02415	Yes	Yes
ABC	MTLABC	0.00422	Yes	Yes	5.49e-11	Yes	Yes	3.47e-10	Yes	Yes
ABC	DFSABC_elite	3.33e-11	Yes	Yes	3.33e-11	Yes	Yes	3.01e-11	Yes	Yes
ABC	MDFSABC	3.33e-11	Yes	Yes	4.97e-11	Yes	Yes	5.49e-11	Yes	Yes
MABC	TLABC	2.95e-05	Yes	Yes	2.59e-05	Yes	Yes	0.21156	No	No
MABC	MTLABC	0.03643	Yes	Yes	7.38e-11	Yes	Yes	5.46e-09	Yes	Yes
MABC	DFSABC_elite	3.01e-11	Yes	Yes	4.07e-11	Yes	Yes	1.95e-10	Yes	Yes
MABC	MDFSABC	3.015e-11	Yes	Yes	4.50e-11	Yes	Yes	1.95e-10	Yes	Yes
TLABC	MTLABC	0.00350	Yes	Yes	4.94e-05	Yes	Yes	1.15e-07	Yes	Yes
TLABC	DFSABC_elite	3.01e-11	Yes	Yes	4.31e-08	Yes	Yes	3.19e-09	Yes	Yes
TLABC	MDFSABC	3.01e-11	Yes	Yes	9.06e-08	Yes	Yes	9.75e-10	Yes	Yes
MTLABC	DFSABC_elite	3.33e-11	Yes	Yes	0.00151	Yes	Yes	0.08235	No	Yes
MTLABC	MDFSABC	3.33e-11	Yes	Yes	0.01383	No	No	0.00215	Yes	Yes
DFSABC_elite	MDFSABC	0.00403	Yes	Yes	0.59968	No	No	0.22257	No	No

CHAPTER 5
CONCLUSION AND FUTURE STUDIES

5.1 Summary and Discussion

The aim of this thesis was to find a non-intrusive drowsiness detection model that is based on eye and mouth features. Thus, many kinds of literature have been studied in order to clarify the environment for the identification of drivers' drowsiness. Such literature works have created further concern because the drowsiness of the driver leads significantly to road accidents. This issue revealed that, although most previous drowsiness studies achieved high precision, they couldn't apply in real-world scenarios. In this thesis, a low cost and non-intrusive drowsiness detection model is proposed. It is noted that the obtained accuracy of the proposed model is 77.19%. In order to achieve the overall drowsiness detection model, three stages were built separately. In the first stage, the Viola and Jones method is used to detect the mouth and the eyes of the driver.

The second stage is the detection of yawning and eyes state. To achieve this stage multi-layer neural network was trained using six versions of the ABC algorithm. Out of these six versions, the proposed MDFSABC shows the best results. The results of this stage show that the proposed models are promising and give high accuracies. The third and final stage is focused on proposing a FIS and using the output of the models of the previous stage as inputs, while the output is the drowsiness level.

In order to analyze the results statistically, two non-parametric statistical tests were used, namely Kruskal-Wallis test and the Wilcoxon Sum Rank test. These statistical tests proved that there is a statistical difference between the six versions of the ABC algorithm.

5.2 Future Study

In this section, some ideas are recommended as future work. There are, in fact, three recommendations:

1. The dataset used in this work is not suitable for drowsiness detection since it is proposed for yawning detection. The proposed framework and methodology of this study can be applied to more general datasets.
2. Six meta-heuristic optimization algorithms are used in this work to train the multi-layer neural network. In fact, too many optimization algorithms have already been proposed in the literature. Some of these algorithms can also be used for the multi-layer neural network training process.
3. The proposed optimization algorithms in this work can be applied to train other kinds of machine learning models.
4. Finally, the proposed model in this work can be improved and implemented in real-world prototypes.

REFERENCES

1. World Health Organization. *Road traffic injuries*. 2019, Augustus 2 Available from: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries.

2. Knipling, R.R. and J.-S. Wang. 1995. *Revised estimates of the US drowsy driver crash problem size based on general estimates system case reviews*. in *Annual proceedings of the Association for the Advancement of Automotive Medicine*. Association for the Advancement of Automotive Medicine.

3. Statistics Related to Drowsy Driver Crashes. 2019, Augustus 2; Available from: http://www.americanindian.net/sleepstats.html.

4. Sahayadhas, A., K. Sundaraj, and M. Murugappan, 2012. *Detecting driver drowsiness based on sensors: a review.* **Sensors**, **12**(12): p. 16937-16953.

5. Ramzan, M., et al., 2019. *A survey on state-of-the-art drowsiness detection techniques.* IEEE Access, **7**: p. 61904-61919.

6. Barr, L., S. Popkin, and H. Howarth, 2009. *An evaluation of emerging driver fatigue detection measures and technologies*. United States. Federal Motor Carrier Safety Administration.

7. Soares, S., S. Ferreira, and A. Couto, 2020. *Driving simulator experiments to study drowsiness: a systematic review.* **Traffic injury prevention**, **21**(1): p. 29-37.

8. Wang, X. and C. Xu, 2016. *Driver drowsiness detection based on non-intrusive metrics considering individual specifics.* **Accident Analysis & Prevention**, **95**: p. 350-357.

9. Caponecchia, C. and A. Williamson, 2018. *Drowsiness and driving performance on commuter trips.* **Journal of safety research**, **66**: p. 179-186.

10. Abtahi, S., et al. 2014. *YawDD: A yawning detection dataset*. in *Proceedings of the 5th ACM Multimedia Systems Conference*.

11. Shi, S.-Y., W.-Z. Tang, and Y.-Y. Wang. 2017. *A review on fatigue driving detection*. in *ITM Web of Conferences*. EDP Sciences.

12. Mbouna, R.O., S.G. Kong, and M.-G. Chun, 2013. *Visual analysis of eye state and head pose for driver alertness monitoring.* **IEEE transactions on intelligent transportation systems**, **14**(3): p. 1462-1469.

13. Jo, J., et al., 2014. *Detecting driver drowsiness using feature-level fusion and user-specific classification.* **Expert Systems with Applications**, **41**(4): p. 1139-1152.

14. Lawoyin, S.A., D.-Y. Fei, and O. Bai, 2014. *A novel application of inertial measurement units (IMUs) as vehicular technologies for Drowsy driving detection via steering wheel movement.* **Open Journal of Safety Science and Technology**, **4**(04): p. 166.

15. Lenné, M.G. and E.E. Jacobs, 2016. *Predicting drowsiness-related driving events: a review of recent research methods and future opportunities.* **Theoretical Issues in Ergonomics Science**, **17**(5-6): p. 533-553.

16. Osuga, S. and K. Hiraki. 2013. *Physiological and accident prevention effects of sleep alert systems: Comparative study of eye closure and heart rate based alarms*. in *20th ITS World CongressITS Japan*.

17. Lawoyin, S., et al. 2014. *Detection methods for a low-cost accelerometer-based approach for driver drowsiness detection*. in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE.

18. Chacon-Murguia, M.I. and C. Prieto-Resendiz, 2015. *Detecting Driver Drowsiness: A survey of system designs and technology.* **IEEE Consumer Electronics Magazine**, **4**(4): p. 107-119.

19. Teyeb, I., et al. 2014. *A novel approach for drowsy driver detection using head posture estimation and eyes recognition system based on wavelet network*. in *IISA 2014, The 5th International Conference on Information, Intelligence, Systems and Applications*. IEEE.

20. Matsuo, H. and A. Khiat. 2012. *Prediction of drowsy driving by monitoring driver's behavior*. in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. IEEE.

21. Viola, P. and M. Jones. 2001. *Rapid object detection using a boosted cascade of simple features*. in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*. IEEE.

22. Wu, Y.-S., et al. 2010. *An eye state recognition method for drowsiness detection*. in *2010 IEEE 71st Vehicular Technology Conference*. IEEE.

23. Pradhan, T., A.N. Bagaria, and A. Routray. 2012. *Measurement of PERCLOS using eigen-eyes*. in *2012 4th International Conference on Intelligent Human Computer Interaction (IHCI)*. IEEE.

24. Omidyeganeh, M., A. Javadtalab, and S. Shirmohammadi. 2011. *Intelligent driver drowsiness detection through fusion of yawning and eye closure*. in *2011 IEEE International Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems Proceedings*. IEEE.

25. Vural, E., et al. 2007. *Drowsy driver detection through facial movement analysis*. in *International Workshop on Human-Computer Interaction*. Springer.

26. Littlewort, G., et al. 2011. *The computer expression recognition toolbox (CERT)*. in *Face and gesture 2011*. IEEE.

27. Ekman, P., et al., 1987. *Universals and cultural differences in the judgments of facial expressions of emotion.* **Journal of personality and social psychology**, **53**(4): p. 712.

28. Vural, E., et al. 2010. *Discrimination of moderate and acute drowsiness based on spontaneous facial expressions*. in *2010 20th International Conference on Pattern Recognition*. IEEE.

29. Memon, S., et al. 2017. *Tracker for sleepy drivers at the wheel*. in *2017 11th International Conference on Signal Processing and Communication Systems (ICSPCS)*. IEEE.

30. Anitha, C., M. Venkatesha, and B.S. Adiga, 2016. *A two fold expert system for yawning detection.* Procedia computer science, **92**: p. 63-71.

31. Omidyeganeh, M., et al., 2016. *Yawning detection using embedded smart cameras.* **IEEE Transactions on Instrumentation and Measurement**, **65**(3): p. 570-582.

32. Park, S., et al. 2016. *Driver drowsiness detection system based on feature representation learning using various deep networks*. in *Asian Conference on Computer Vision*. Springer.

33. Bamidele, A., et al., 2019. *Non-intrusive Driver Drowsiness Detection based on Face and Eye Tracking.* **Int J. Adv. Comput. Sci. Appl**, **10**: p. 549-569.

34. Ji, Y., et al., 2019. *Fatigue state detection based on multi-index fusion and state recognition network.* IEEE Access, **7**: p. 64136-64147.

35. ping Tian, D., 2013. *A review on image feature extraction and representation techniques.* **International Journal of Multimedia and Ubiquitous Engineering**, **8**(4): p. 385-396.

36. Zhang, D. and G. Lu, 2004. *Review of shape representation and description techniques.* **Pattern recognition**, **37**(1): p. 1-19.

37. Saleh, R.A.A. and R. Akay, 2019. *Classification of Melanoma Images Using Modified Teaching Learning Based Artificial Bee Colony.* **Avrupa Bilim ve Teknoloji Dergisi**, p. 225-232.

38. Clausi, D.A., 2002. *An analysis of co-occurrence texture statistics as a function of grey level quantization.* **Canadian Journal of remote sensing**, **28**(1): p. 45-62.

39. Zhou, X., et al., 2017. *Enhancing the modified artificial bee colony algorithm with neighborhood search.* **Soft Computing**, **21**(10): p. 2733-2743.

40. Mousavirad, S.J. and H. 2017. Ebrahimpour-Komleh, *Multilevel image thresholding using entropy of histogram and recently developed population-based metaheuristic algorithms.* **Evolutionary Intelligence**, **10**(1-2): p. 45-75.

41. Elaziz, M.A., S. Bhattacharyya, and S. Lu, 2019. *Swarm selection method for multilevel thresholding image segmentation.* **Expert Systems with Applications**, **138**: p. 112818.

42. Mirjalili, S., S.M. Mirjalili, and A. Lewis, 2014. *Grey wolf optimizer.* **Advances in engineering software**, **69**: p. 46-61.

43. Yang, X.-S., 2014. *Nature-inspired optimization algorithms*. Elsevier.

44. Jadon, S.S., et al., 2017.*Hybrid artificial bee colony algorithm with differential evolution.* **Applied Soft Computing**, **58**: p. 11-24.

45. Akay, B. and D. Karaboga, 2012. *A modified artificial bee colony algorithm for real-parameter optimization.* **Information sciences**, **192**: p. 120-142.

46. Gao, W., S. Liu, and L. Huang, 2012. *A global best artificial bee colony algorithm for global optimization.* **Journal of Computational and Applied Mathematics**, **236**(11): p. 2741-2753.

47. Zhu, G. and S. Kwong, 2010. *Gbest-guided artificial bee colony algorithm for numerical function optimization.* **Applied mathematics and computation**, **217**(7): p. 3166-3173.

48. Karaboga, D. and B. Basturk, 2008. *On the performance of artificial bee colony (ABC) algorithm.* **Applied soft computing**, **8**(1): p. 687-697.

49. TSai, P.-W., et al., 2009. *Enhanced artificial bee colony optimization.* **International Journal of Innovative Computing, Information and Control**, **5**(12): p. 5081-5092.

50. Gao, W. and S. Liu, 2011. *Improved artificial bee colony algorithm for global optimization.* **Information Processing Letters**, **111**(17): p. 871-882.

51. Luo, J., Q. Wang, and X. Xiao, 2013. *A modified artificial bee colony algorithm based on converge-onlookers approach for global optimization.* **Applied Mathematics and Computation**, **219**(20): p. 10253-10262.

52. Gao, W.-f., S.-y. Liu, and L.-l. Huang, 2013. *A novel artificial bee colony algorithm based on modified search equation and orthogonal learning.* **IEEE Transactions on Cybernetics**, **43**(3): p. 1011-1024.

53. Xiang, Y., et al., 2014. *A particle swarm inspired multi-elitist artificial bee colony algorithm for real-parameter optimization.* **Computational Optimization and Applications**, **57**(2): p. 493-516.

54. Gao, W., et al., 2015. *Bare bones artificial bee colony algorithm with parameter adaptation and fitness-based neighborhood.* **Information Sciences**, **316**: p. 180-200.

55. Gao, W.-F., S.-Y. Liu, and L.-L. Huang, 2013. *A novel artificial bee colony algorithm with Powell's method.* **Applied Soft Computing**, **13**(9): p. 3763-3775.

56. dos Santos Coelho, L. and P. Alotto, 2011. *Gaussian artificial bee colony algorithm approach applied to Loney's solenoid benchmark problem.* **IEEE Transactions on Magnetics**, **47**(5): p. 1326-1329.

57. Karaboga, D. and B. Gorkemli, 2014. *A quick artificial bee colony (qABC) algorithm and its performance on optimization problems.* **Applied Soft Computing**, **23**: p. 227-238.

58. Kıran, M.S. and O. Fındık, 2015. *A directed artificial bee colony algorithm.* **Applied Soft Computing**, **26**: p. 454-462.

59. Xiang, W.-L. and M.-Q. An, 2013. *An efficient and robust artificial bee colony algorithm for numerical optimization.* **Computers & Operations Research**, **40**(5): p. 1256-1265.

60. Kang, F., J. Li, and H. Li, 2013. *Artificial bee colony algorithm and pattern search hybridized for global optimization.* **Applied Soft Computing**, **13**(4): p. 1781-1791.

61. Wang, H., et al., 2014. *Multi-strategy ensemble artificial bee colony algorithm.* **Information Sciences, 279**: p. 587-603.

62. Kiran, M.S., et al., 2015. *Artificial bee colony algorithm with variable search strategy for continuous optimization.* **Information Sciences**, **300**: p. 140-157.

63. Jadon, S.S., et al., 2015. *Accelerating artificial bee colony algorithm with adaptive local search.* **Memetic Computing**, **7**(3): p. 215-230.

64. Cui, L., et al., 2016. *A novel artificial bee colony algorithm with depth-first search framework and elite-guided search equation.* **Information Sciences**, **367**: p. 1012-1044.

65. Abraham, A., R.K. Jatoth, and A. Rajasekhar, 2012. *Hybrid differential artificial bee colony algorithm.* **Journal of computational and theoretical Nanoscience**, **9**(2): p. 249-257.

66. Shi, X., et al. 2010. *An integrated algorithm based on artificial bee colony and particle swarm optimization*. in *2010 Sixth international conference on natural computation*. IEEE.

67. Zhao, H., et al. 2010. *A hybrid swarm intelligent method based on genetic algorithm and artificial bee colony*. in *International Conference in Swarm Intelligence*. Springer.

68. Sharma, H., et al., 2016. *Lévy flight artificial bee colony algorithm.* **International Journal of Systems Science**, **47**(11): p. 2652-2670.

69. Sharma, H., J.C. Bansal, and K.V. Arya, 2013. *Opposition based lévy flight artificial bee colony.* **Memetic Computing**, **5**(3): p. 213-227.

70. Saleh, R.A.A. and R. Akay, 2019. *Indoor Localization Using Artificial Bee Colony with Levy Flight.* **Avrupa Bilim ve Teknoloji Dergisi**, p. 152-156.

71. Chen, X. and B. Xu. 2018. *Teaching-learning-based artificial bee colony*. in *International Conference on Swarm Intelligence*. Springer.

72. Elmolla, E. S., & Chaudhuri, M. 2011. The use of artificial neural network (ANN) for modelling, simulation and prediction of advanced oxidation process performance in recalcitrant wastewater treatment. Artificial Neural Networks-Application.

73 Mehrotra, K., C.K. Mohan, and S. Ranka, 1997. *Elements of artificial neural networks*. MIT press.

74. Haykin, S., 2010. *Neural Networks and Learning Machines, 3/E*. Pearson Education India.

75. G. Rohra, J., et al., 2017. *User Localization in an Indoor Environment Using Fuzzy Hybrid of Particle Swarm Optimization & Gravitational Search Algorithm with Neural Networks*. p. 286-295.

76. Shah, H., et al. 2012. *Global hybrid ant bee colony algorithm for training artificial neural networks*. in *International Conference on Computational Science and Its Applications*. Springer.

77. Blum, C. and K. Socha. 2005. *Training feed-forward neural networks with ant colony optimization: An application to pattern classification*. in *Fifth International Conference on Hybrid Intelligent Systems (HIS'05)*. IEEE.

78. Mendes, R., et al. 2002. *Particle swarms for feedforward neural network training*. in *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No. 02CH37290)*. IEEE.

79. Ilonen, J., J.-K. Kamarainen, and J. Lampinen, 2003. *Differential evolution training algorithm for feed-forward neural networks.* **Neural Processing Letters**, **17**(1): p. 93-105.

80. Yao, X., 1993. *Evolutionary artificial neural networks.* **International journal of neural systems**, **4**(03): p. 203-222.

81. Montana, D.J. and L. Davis, 1989. *Training Feedforward Neural Networks Using Genetic Algorithms*. in *IJCAI*.

82. Karaboga, D. and B. Akay, 2007. *Artificial bee colony (ABC) algorithm on training artificial neural networks*. in *2007 IEEE 15th Signal Processing and Communications Applications*. IEEE.

83. Ozturk, C. and D. Karaboga, 2011. *Hybrid artificial bee colony algorithm for neural network training*. in *2011 IEEE congress of evolutionary computation (CEC)*. IEEE.

84. Shah, H., R. Ghazali, and N.M. Nawi, 2013. *Global artificial bee colony algorithm for boolean function classification*. in *Asian Conference on Intelligent Information and Database Systems*. Springer.

85. Nawi, N.M., A. Khan, and M.Z. Rehman, 2013. *A new back-propagation neural network optimized with cuckoo search algorithm*. in *International conference on computational science and its applications*. Springer.

86. Nawi, N.M., M.Z. Rehman, and A. Khan, 2014. *A new bat based back-propagation (BAT-BP) algorithm*, in ***Advances in Systems Science***. Springer. p. 395-404.

87. Chaudhari, S., M. Patil, and J. Bambhori, 2014. *Study and review of fuzzy inference systems for decision making and control.* **American International Journal of Research in Science, Technology, Engineering & Mathematics**, **14**(147): p. 88-92.

88. Cherkassky, V., 1998. *Fuzzy inference systems: a critical review*, in ***Computational Intelligence: Soft Computing and Fuzzy-Neuro Integration with Applications***. Springer. p. 177-197.

89. Fullér, R., *Neural fuzzy systems.* 1995.

90. Wu, J.-D. and T.-R. Chen, 2008. *Development of a drowsiness warning system based on the fuzzy logic images analysis.* **Expert Systems with Applications**, **34**(2): p. 1556-1561.

91. Lee, K.H., 2004. *First course on fuzzy theory and applications*. Vol. 27. Springer Science & Business Media.

92. Liang, J. J., Qu, B. Y., & Suganthan, P. N. 2013. *Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization.* Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, 635.

93. Blake C, M.C., 1998. *{UCI} Machine Learning Repository*.

94. Mirjalili, S., 2015. *How effective is the Grey Wolf optimizer in training multi-layer perceptrons.* **Applied Intelligence**, **43**(1): p. 150-161.

95. Gupta, S., & Deep, K. 2019. *A novel random walk grey wolf optimizer*. **Swarm and evolutionary computation**, 44, 101-112.

96. A.A. Heidari, P. Pahlavani, 2017. *An efficient modified grey wolf optimizer with Lévy flight for optimization tasks*. **Applied Soft Computing** 60: 115-134.

97. E. Rashedi, Esmat, H. Nezamabadi-Pour, S. Saryazdi, 2009. *GSA: a gravitational search algorithm.* **Information Sciences** 179, no. 13: 2232-2248

98. X.S. Yang, S. Deb, 2009. *Cuckoo search via Lévy flights.* In Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on, pp. 210-214. IEEE.

99. V. Garg, K. Deep, 2016. P*erformance of Laplacian Biogeography-Based Optimization Algorithm on CEC 2014 continuous optimization benchmarks and camera calibration problem*. **Swarm and Evolutionary Computation** 27: 132-144.

100. M.Y. Cheng, D. Prayogo, 2014. *Symbiotic organisms search: a new metaheuristic optimization algorithm*. **Computers & Structures** 139: 98-112.

101. Tan, X., J. Liu, and S. Chen, 2006. *Sub-intrapersonal space analysis for face recognition.* **Neurocomputing**. **69**(13-15): p. 1796-1801.

102. Haralick, R.M., K. Shanmugam, and I.H. Dinstein, 1973. *Textural features for image classification.* **IEEE Transactions on systems, man, and cybernetics**, (6): p. 610-621.

103. Åkerstedt, T. & Gillberg, M., 1990. Subjective and objective sleepiness in the active individual. **International Journal of Neuroscience**, 52(1-2).

CURRICULUM VITAE

PERSONAL DATA

Name and Surname	:	Radhwan Ali Abdulghani Saleh
Nationality	:	Yemen
Birth Date and Place	:	01/01/1991 in Yemen-Taiz
Marital Status	:	Married
Tel:	:	+90 5314854504
email	:	4012031712@erciyes.edu.tr redhwan9191@gmail.com
Address	:	Yenidoğan Mah. Ertuğrul Gazi Cad. 16/ 23 Talas/Kayseri

EDUCATION

Degree	Institute	Date of Graduation
Master's degree	Erciyes University	2018-2020
Turkish Preparation	Erciyes University/ERSEM	2017-2018
Bachelor's degree	Taiz University	2010-2015
Applied electronics diploma	International Training & Development Center	2013-2014
High School	Taiz Secondary School	2006-2009

PROFESSIONAL EXPERIENCE

Year	Institute	Mission
2015-2017	Taiz University	Teaching Assistant
2014-2015	Communing Organization for the Care and Education of young	Data entry supervisor
2014–Present	Alrhal Company For Software	Embedded systems software developer

LANGUAGES:

Arabic, English, Turkish

PUBLICATIONS:

a. Saleh, R. A. A., & Akay, R. (2019). Classification of Melanoma Images Using Modified Teaching Learning Based Artificial Bee Colony. Avrupa Bilim ve Teknoloji Dergisi, 225-232.
b. Saleh, R. A. A., & Akay, R. (2019). Indoor Localization Using Artificial Bee Colony with Levy Flight. Avrupa Bilim ve Teknoloji Dergisi, 152-156.
c. Rustu Akay, Radhwan A.A. Saleh, Shawqi M.O. Farea, and Muzaffer Kanaan. Multilevel Thresholding Segmentation of Color Plant Disease Images Using Metaheuristic Optimization Algorithms. Neural Computing and Applications. (Under review).