Neural Computing and Applications (2022) 34:15907–15944
https://doi.org/10.1007/s00521-022-07241-1

ORIGINAL ARTICLE

A multi-variate heart disease optimization and recognition framework

Hossam Magdy Balaha[1] · Ahmed Osama Shaban[2] · Eman M. El-Gendy[1] · Mahmoud M. Saafan[1]

Received: 28 August 2021 / Accepted: 29 March 2022 / Published online: 2 May 2022

Abstract
Cardiovascular diseases (CVD) are the most widely spread diseases all over the world among the common chronic diseases. CVD represents one of the main causes of morbidity and mortality. Therefore, it is vital to accurately detect the existence of heart diseases to help to save the patient life and prescribe a suitable treatment. The current evolution in artificial intelligence plays an important role in helping physicians diagnose different diseases. In the present work, a hybrid framework for the detection of heart diseases using medical voice records is suggested. A framework that consists of four layers, namely "Segmentation" Layer, "Features Extraction" Layer, "Learning and Optimization" Layer, and "Export and Statistics" Layer is proposed. In the first layer, a novel segmentation technique based on the segmentation of variable durations and directions (i.e., forward and backward) is suggested. Using the proposed technique, 11 datasets with 14,416 numerical features are generated. The second layer is responsible for feature extraction. Numerical and graphical features are extracted from the resulting datasets. In the third layer, numerical features are passed to 5 different Machine Learning (ML) algorithms, while graphical features are passed to 8 different Convolutional Neural Networks (CNN) with transfer learning to select the most suitable configurations. Grid Search and Aquila Optimizer (AO) are used to optimize the hyperparameters of ML and CNN configurations, respectively. In the last layer, the output of the proposed hybrid framework is validated using different performance metrics. The best-reported metrics are (1) 100% accuracy using ML algorithms including Extra Tree Classifier (ETC) and Random Forest Classifier (RFC) and (2) 99.17% accuracy using CNN.

Keywords Aquila optimizer (AO) · Convolutional neural network (CNN) · Deep learning (DL) · Heart disease · Machine learning (ML) · Metaheuristic optimization

1 Introduction

Cardiovascular diseases (CVDs) are the most worldwide spread chronic diseases all over the world and represented the top cause of morbidity and death in the last ten years globally [1]. According to the World Health Organization (WHO), there are 17.9 million people died from CVDs every year representing 32% of all death cases worldwide. So, day after day, the cases of CVDs are at a rapid rate and by 2030, the yearly death rate will increase and reach 22.2 million people approximately [2, 3]. The center of disease control and prevention report confirms the expectations for increasing the mortality rate. It said that every 40 s, one person died due to the CVDs [4]. In Egypt as well, CVDs are considered as the leading cause of death in the last 30 years, and in 2017, the CVDs had an estimated value of 46.2% of all mortality cases [5].

CVDs are an umbrella term including a group of disorders for the heart and blood vessels containing many types such as (1) congestive heart failure, (2) coronary heart disease, (3) congenital heart disease, (4) cerebrovascular disease, and (5) rheumatic heart disease [6]. From five CVDs death, there are four cases resulting from strokes and heart attacks. So, heart disease can be considered as the most life-snatching chronic disease and its risk comes from the silently of the disease. It is not diagnosed until the symptoms of heart failure (or attack) are recognized [7]. In

✉ Hossam Magdy Balaha
hossam.m.balaha@mans.edu.eg

[1] Computer Engineering and Control Systems Department, Faculty of Engineering, Mansoura University, Mansoura, Egypt

[2] Biomedical Engineering Program, Faculty of Engineering, Mansoura University, Mansoura, Egypt

heart disease, the heart fails to do its normal function by supplying blood to other body parts because of the blockage of coronary arteries that are responsible for supplying blood to the heart [8]. The regular heart disease symptoms include (1) breath shortness, (2) body weakness, (3) confusion, and (4) fainting. This disease risk can be increased with people with perilous cases including (1) unhealthy diet, (2) smoking, (3) fitness issues, (4) high blood pressure, (5) exercise deficiency, and (6) high cholesterol level [9].

The early and accurate prediction for heart disease is very crucial to enhance the survival rate and reduce the mortality rate. This will help healthcare professionals in their decisions by providing an accurate and efficient diagnosis and treatment for patients to save their lives [10]. One of the approaches for the early and accurate prediction for heart disease is machine intelligence. This can be achieved using machine learning (ML) algorithms and deep learning (DL) approaches [11]. Various types of heart disease data such as images, waves, and sounds can be used to perform that [12].

Image data can be analyzed and the features are extracted, to train the ML (or DL) approach such as CNN, to determine if the images are belonging to a diseased or healthy patient [13]. Detecting heart disease can also be through gathering features obtained from cardiac sounds to be the input for a DL or ML algorithm beside those cardiac sounds can be converted into numerical data to be utilized as an input for a DL approach to check the patient condition if it had heart disease or not [14]. Another type of data which have been deployed for heart disease detection and know the patient condition through analyzing Electrocardiogram (ECG) and Electroencephalogram (EEG) waves to assign the correct label to be the input for the Recurrent Neural Network (RNN) model or extracting features from the signals and convert them into a numerical data which had been used as the input for ML algorithm [15]. ML algorithms such as support vector machines and decision trees have an essential role in predicting the existence of heart disease accurately by analyzing the medical data whether its voice or images numerically [16–18]. Also, DL approaches such as convolutional neural networks (CNN) can analyze them efficiently and can deal with large datasets [19–21].

1.1 Paper contributions

The current study focuses mainly on developing a hybrid system from ML algorithms and CNN models to predict and detect the existence of heart disease accurately based on the analysis of the medical voice records and images. The suggested approach will aid healthcare professionals to improve the provided medical care to patients. The contributions of the current study can be summarized in the following points:

- Proposing a hybrid system from ML algorithms and DL approach for predicting heart disease.
- Analyzing different types of datasets including medical images and voice records.
- Suggesting a hybrid DL and AO approach for the learning and optimization processes.
- Reporting state-of-the-art performance metrics compared with other related studies and approaches.

1.2 Paper organization

The rest of this paper is organized as follows: In the next section, the related studies that have heart disease diagnosis and prediction processes contributions are described. Section 3 depicts the basic concepts regarding voice feature techniques, ML algorithms, Convolutional Neural Network (CNN), Metaheuristic Optimization using Aquila Optimizer (AO), Image Augmentation, and Data Normalization. In Sect. 4, the suggested approach of this work during the heart disease learning and optimization phase is discussed. Section 5 illustrates the experiments and the reported results of different approaches. Finally, Sect. 6 represents the main conclusion and future work.

2 Related work

In this section, the existing studies and research papers, related to heart disease diagnosis and prediction processes based on various types of medical data, are introduced. The related studies are split into studies that focused on (1) deep learning approaches, (2) machine learning algorithms, and (3) hybrid approach.

2.1 Deep learning-based studies

Brunese et al. [22] proposed a methodology for detecting heart disease using DL and through cardio sounds. They used deep neural networks (DNN) to extract a set of features and analyzed the cardio sounds. They showed if they belonged to healthy patients or those with heart disease. 176 heartbeats were considered when they performed the experiments and their results showed that 145 of them related to heart disease patients and only 31 heartbeats for healthy patients. The overall accuracy was 98%. Miao et al. [23] developed a DNN for predicting and diagnosing coronary heart disease. They used Multi-Layer Perceptron (MLP), regularization, and dropout. They utilized 303 instances containing attributes from patients at the

Cleveland clinic foundation and achieved 83.67% accuracy and 93.51% sensitivity.

Abdel-Alim et al. [24] proposed a heart disease diagnosis system using ANN and through classifying several cases related to heart disorders using heart sounds. The used dataset contained 850 cases that were partitioned into 650 cases for the ANN training process and the other 200 cases for testing. They utilized different techniques to perform the diagnosis process such as (1) Fast Fourier Transform, (2) Discrete Wavelet Transfer, and (3) Linear Prediction Coding. They achieved a recognition rate of 95%. Ali et al. [19] suggested a smart monitoring system for predicting heart disease through DL approaches, feature selection, feature fusion, and weighting techniques on the used Cleveland and Hungarian datasets. The proposed approach achieved an accuracy of 98.5%. Zhang et al. [25] carried out ECG classification using CNNs to identify heart disease. They deployed the process using a dataset consisting of 102,548 heartbeats and achieved 97.7%, 97.6%, and 97.6% for positive predictive rate, sensitivity, and F1-score, respectively.

Zhang et al. [26] suggested an approach for diagnosing heart disease through signal processing and DL models which predicted the disease from the ECG signals. The used dataset contained 8524 single lead episodic ECG records and reported 0.87 on the F1-score performance metric. Kwon et al. [27] developed a DL approach for mortality rate prediction among heart disease patients from their ECG. The result showed that there are 1026 patients with a mortality rate among 25,776 cases and that confirmed the model achieved accurate results compared to existing or previous ML models. Sajeev et al. [28] proposed an approach for a heart disease prediction system that depended on DL models. It could determine the probabilities of disease risk on the patients. After applying the performance metrics, they achieved an accuracy of 94% and an Area Under the Curve (AUC) score of 0.964.

Rath et al. [29] carried out heart disease detection on the ECG samples through the DL model. The utilized model was depending on LSTM and Generative Adversarial Network (GAN) to achieve the best efficiency. The results reported the best accuracy of 99.2%, F1-score of 0.987, and AUC score of 0.984. Darmawahyuni et al. [30] developed a framework for detecting coronary heart disease based on DNN and UCI repository heart disease dataset. They achieved a specificity of 92%, sensitivity of 99%, and accuracy of 96%.

2.2 Machine learning-based studies

Jindal et al. [31] proposed a heart disease prediction system using ML algorithms to predict the condition of the patient and determine if it had heart disease or not. They depended on the medical history of each patient in a dataset that contained 13 medical attributes for 304 patients which were collected from the UCI repository. They used ML algorithms such as (1) K-Nearest Neighbor (KNN), (2) Logistic Regression (LR), and (3) Random Forest Classifier (RFC). From these algorithms, KNN achieved the best accuracy with a value of 88.52%. Also, they built a model from the used ML algorithms and it achieved 87.5% accuracy which was better compared to their related studies.

Muhammad et al. [32] developed an intelligent computational model for the early and accurate detection and diagnosis of heart disease based on ML algorithms. They utilized many ML algorithms such as (1) RFC, (2) Artificial Neural Network (ANN), (3) Support Vector Machine (SVM), (4) LR, (5) KNN, (6) Naïve Bayes (NB), (7) Extra-Tree Classifier (ETC), (8) Gradient Boosting (GB), (9) AdaBoost (AB), and (10) Decision Tree (DT) on the Cleveland and Hungarian heart disease datasets that were available on the UCI repository. They made a comparison between the algorithms and utilized performance evaluation metrics to show the best algorithms. They were the ETC and GB with overall accuracy values 94.41% and 93.36% respectively.

Pugazhenthi et al. [33] developed a framework for detecting ischemic heart disease from medical images using ML algorithms such as (1) MLP, (2) SVM, and (3) C5 classifier. The reported results showed that the highest accuracy was reported by SVM with an accuracy of 92.1%. Alarsan et al. [34] developed an approach of heart disease detection based on ECG classification and using ML algorithms to extract features that were required for the classification process. They used a dataset that contained 205,146 records for 51 patients. They deployed ML algorithms such as (1) RFC, (2) DT, and (3) Gradient-Boosted Trees (GDB). The highest accuracy was for the GDB algorithm which was 97.98%. Nikhar et al. [35] proposed a methodology for predicting heart disease using ML algorithms on the Cleveland heart disease database which contains 303 records with 76 medical attributes. They performed the experiments using NB and DT which achieved the highest accuracies.

Patel et al. [36] performed a heart disease prediction system by utilizing ML algorithms and data mining techniques on the Cleveland database of UCI repository which had 303 instances. The used algorithms are RFC and Logistic Model Tree (LMT) that performed the prediction process effectively. Singh et al. [37] proposed a prediction system for heart disease using ML approaches. They made a comparison between various ML algorithms such as KNN, SVM, DT, and LR on a dataset collected from the UCI repository. The results showed that the highest accuracy was achieved by KNN with an overall accuracy of

Table 1 Related Studies in (2021 and 2020) Summarization

References	Year	Approach	Pros.	Cons.	Best performance
Jindal et al. [31]	2021	Proposed heart disease prediction system using ML algorithms to predict the condition of the patient and determine if it has heart disease or not	Used different algorithms and achieved a good accuracy	Small dataset	Accuracy 88.52%
Rath et al. [29]	2021	Carried out heart disease detection on the ECG samples through the DL model	Better performance and the utilized model is depending on LSTM and GAN	–	Accuracy 99.2%, F1-score 0.987, and AUC score 0.984
Sharma et al. [44]	2020	Suggested a framework for heart disease prediction using DNN on heart disease	Used different ML algorithms, utilized different optimization techniques, and Good accuracy	Small dataset.	Accuracy 90.76%
Muhammad et al. [32]	2020	Develop an intelligent computational model for the early and accurate detection and diagnosis of heart disease based on ML algorithms	Utilized many ML algorithms, good accuracy, and good dataset size	–	Accuracy 94.41%
Pasha et al. [39]	2020	Proposed a framework for predicting cardiovascular disease using DL techniques and different algorithms like SVM, DT, KNN, and ANN	Made a comparison between different algorithms to know the best one between them when dealing with this dataset	Low accuracy	Overall accuracy equals 85.24%.
Brunese et al. [22]	2020	Proposed a methodology for detecting heart disease using DL and through cardio sounds	High accuracy comparing to the other works	Small dataset	Overall accuracy 98%
Sajja et al. [41]	2020	Proposed a DL approach for early prediction of cardiovascular diseases depend on CNN	Utilized different ML algorithms and good accuracy	Small dataset	Accuracy 94.78%
Ali et al. [19]	2020	Suggested smart monitoring system for predicting heart disease through DL approaches	Better accuracy, good dataset size, and utilized different techniques for the prediction process	–	Accuracy 98.5%
Singh et al. [37]	2020	Proposed prediction system for heart disease using ML approaches	Used different ML algorithms and good dataset size		Accuracy 87%

87%, SVM with 83%, DT with 79%, and LR with 78%. Krishnan et al. [38] proposed a prediction system for the probabilities of heart disease based on ML approaches such as DT and NB. They used data from the UCI repository that contained 300 instances with 14 clinical parameters. The DT algorithm had the highest accuracy with 91%.

2.3 Hybrid-based studies

Pasha et al. [39] proposed a framework for predicting cardiovascular disease using DL techniques and different algorithms such as (1) SVM, (2) DT, (3) KNN, and (4) ANN. They collected the dataset that contained attributes related to heart disease from Kaggle. In their work, they made a comparison between the algorithms to know the most optimum one which was ANN with an overall accuracy value of 85.24%. Raza et al. [40] developed a framework for classifying heartbeat sound signals using DL approaches. They utilized a recurrent neural network (RNN) that worked depending on the long short-term memory (LSTM), dense, dropout, and SoftMax layers. They also deployed the MLP, DT, and RFC models. The result showed that RNN is the most efficient one from them and reported an accuracy value of 80.80%. Arabasadi et al. [10] proposed a Computer-Aided System (CAS) for heart disease detection based on a hybrid model using Neural Networks (NN) and Genetic Algorithms (GA). They used a dataset containing information of 303 patients and achieved 93.85% accuracy, 97% sensitivity, and 92% specificity.

Sajja et al. [41] proposed a DL approach for the early prediction of cardiovascular diseases depending on CNNs. They used a dataset from the UCI repository and made a comparison between the traditional algorithms like (1) LR, (2) KNN, (3) SVM, (4) NB, (5) NN, and (6) the proposed approach which reported the best accuracy with 94.78%. Haq et al. [42] proposed a framework of a hybrid intelligent system for the prediction of heart disease based on ML algorithms to identify healthy people and heart disease patients through analyzing the used Cleveland heart disease dataset. They utilized 3 feature selection algorithms, 7 classifiers performance evaluation metrics, and the cross-validation method. The result showed that the best-used algorithms are LR and SVM with accuracies of 89% and

Table 2 Related Studies in (2019 or before) Summarization

References	Year	Approach	Pros.	Cons.	Best performance
Raza et al. [40]	2019	Develop a framework for classifying heartbeat sound signals using the DL approaches	Utilized different techniques	Low accuracy	Accuracy 80.80%
Alarsan et al. [34]	2019	Developed an approach of heart disease detection based on ECG classification and using ML algorithms	Large dataset, better accuracy. Ad utilized different ML algorithms		Accuracy 97.98%
Krishnan et al. [38]	2019	Proposed prediction system for the probabilities of heart disease based on ML approaches	Good accuracy and dataset size		Accuracy 91%
Kwon et al. [27]	2019	Developed DL approach for mortality rate prediction among heart disease patients from their ECG	A large dataset and accurate results compared to existing or the previous ML models		
Darmawahyuni et al. [30]	2019	Developed a framework for detecting coronary heart disease based on DNN	Good utilized dataset size and better performance		Specificity 92%, Sensitivity 99%, and Accuracy 96%
Sajeev et al. [28]	2019	Proposed an approach for heart disease prediction system depend on DL models	Better performance	Small dataset	Accuracy 94% and AUC score 0.964.
Zhang et al. [26]	2019	Suggested an approach for diagnosing heart disease through signal processing and DL models which predict the disease from the ECG signals.	Large dataset and good performance		F1-score 0.87
Zhang et al. [25]	2018	Carried out ECG classification using CNN to identify heart disease	Large dataset and good accuracy		Positive predictive rate 97.7%, sensitivity 97.6%, and F1-score 97.6%
Miao et al. [23]	2018	Develop a DNN for predicting and diagnosing the coronary heart disease	Utilized different algorithms and used a good size for the dataset	Low accuracy	83.67% accuracy and 93.51% sensitivity
Haq et al. [42]	2018	Proposed a framework of a hybrid intelligent system for the prediction of heart disease based on ML algorithms	Utilized 3 feature selection algorithms, 7 classifiers performance evaluation metrics, and cross-validation method	Low accuracy	Accuracy 89%
Gavhane et al. [43]	2018	Suggested a prediction framework for heart disease based on symptoms and using ML algorithms	Used different ML algorithms	Small dataset	
Arabasadi et al. [10]	2017	Proposed a CAS for heart disease detection based on a hybrid model from NN and GA	Good accuracy and used a hybrid model containing GA	Small dataset	Accuracy 93.85%, sensitivity 97%, and specificity 92%.
Pugazhenthi et al. [33]	2016	Developed a framework for detecting ischemic heart disease from medical images using ML algorithms such as MLP, SVM, and C5 classifier	Used different ML algorithms and achieved a good accuracy		Accuracy 92.1%.
Nikhar et al. [35]	2016	Proposed a methodology for predicting heart disease using ML algorithms	Used different ML models		
Patel et al. [36]	2015	Performed heart disease prediction system by utilizing ML algorithms and data mining techniques	Used different ML algorithms and good dataset size		
Abdel-Alim et al. [24]	2002	Proposed a heart disease diagnosis system using ANN and through classifying several valves related heart disorders in the heart sounds	Good dataset size and used different techniques to perform the diagnosis process better		Recognition Rate equals 95%

88% respectively. Gavhane et al. [43] suggested a prediction framework for heart disease based on symptoms and using ML algorithms such as NN and MLP. The results showed that NN was the most accurate algorithm when applied to the prediction process. Sharma et al. [44] suggested a framework for heart disease prediction using DNN

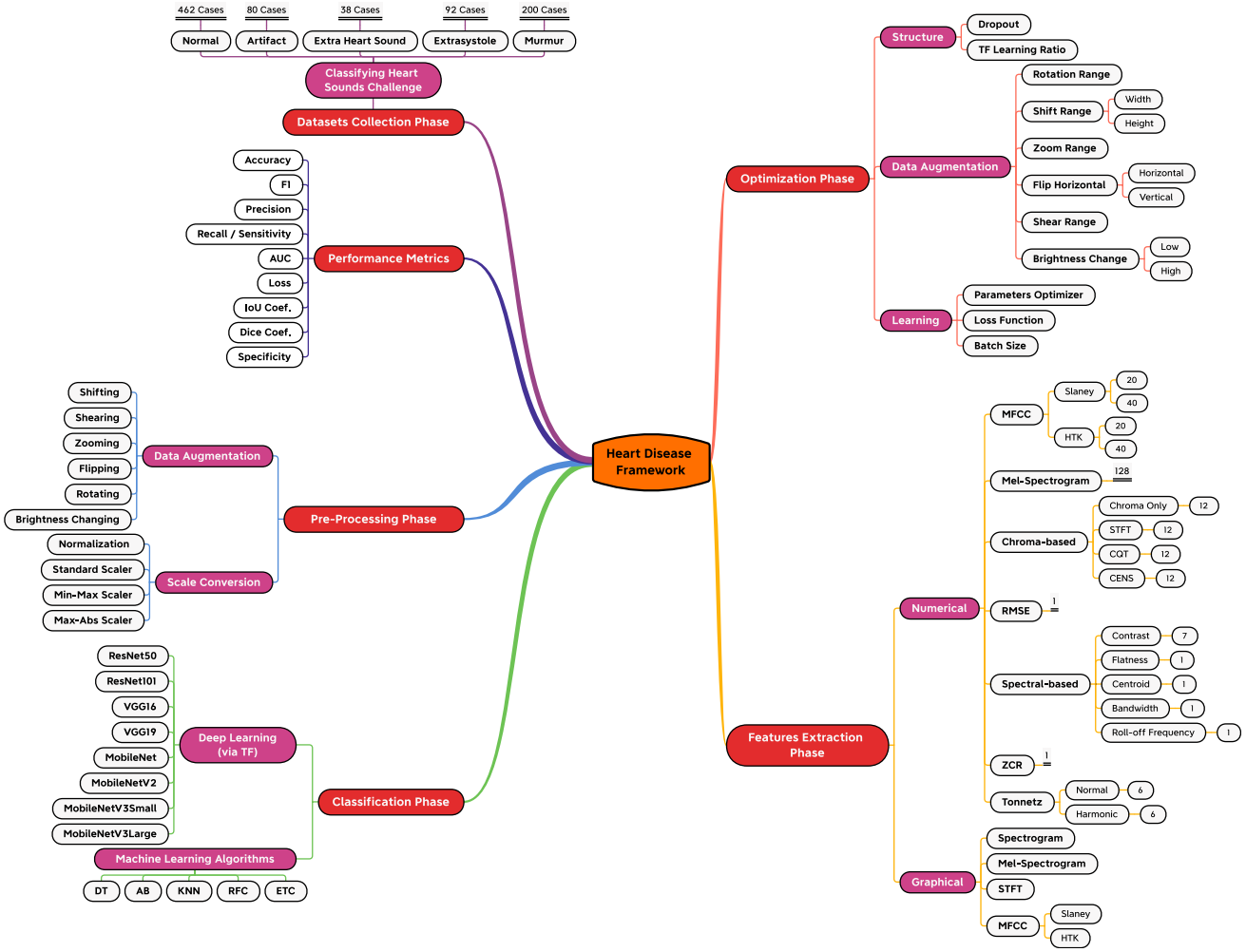

Fig. 1 The Suggested Framework Parts Summarization

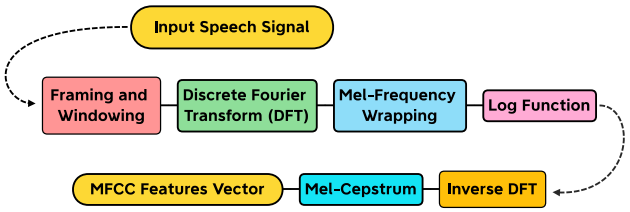

Fig. 2 The Steps of Extracting the MFCC Features from an Audio Record

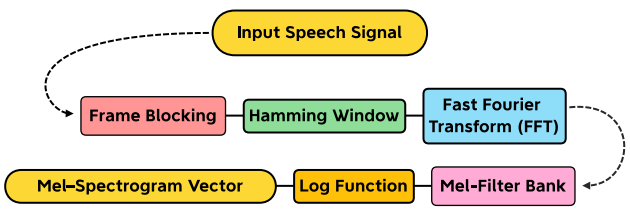

Fig. 3 The Steps of Extracting the MS Features from an Audio Record

on the heart disease UCI repository. They utilized different algorithms like (1) KNN, (2) SVM, (3) NB, and (4) RFC for the classification process. They used Talos optimization with DNN which led to achieving the best accuracy of 90.76%.

2.4 Related studies summarization

Table 1 summarizes the discussed related studies in 2021 and 2020 while Table 2 summarizes the discussed related studies otherwise (i.e., 2019 or before). They are ordered in descending order according to the publication year.

2.5 Plan of solution

The current study proposes a hybrid approach for heart disease learning and optimization through various phases (as shown in Fig. 1). In it, the first phase handles the dataset collection for classifying heart sounds challenge dataset and medical images. The second phase is

preprocessing that data. It includes data augmentation and scales conversion techniques. The third phase is the optimization phase which involves the structure, learning process, and data augmentation approaches by utilizing the pre-trained CNN models and ML algorithms. The fourth phase includes the numerical and graphical features extraction techniques. The fifth phase represents the classification process that is based on DL approaches via transfer learning and ML algorithms. Finally, The sixth phase involves measuring the performance of ML and DL approaches through various experiments and calculated performance metrics.

3 Background

This section provides the main background that can help the reader get in touch with the parts of the suggested hybrid approach. It is divided into the following points:

- Voice feature extraction techniques.
- Machine learning algorithms.
- Convolutional neural network.
- Metaheuristic optimization using aquila optimizer (AO).
- Image augmentation.
- Data normalization.

3.1 Voice feature extraction techniques

Feature extraction is one of the important steps in the learning process of the algorithms through minimizing calculations, the choice of the most optimum features in the dataset, and the choice of the required information to train the model with it [45]. The features can be extracted from different types of data such as audio, images, and waves [46]. In the current study, the features are extracted numerically and graphically from the audio records. There are a lot of audio features extraction techniques but the used ones in the current study are (1) Mel-Frequency Cepstral Coefficients (MFCC), (2) Mel-Spectrogram, (3) Zero Crossing Rate (ZCR), (4) Root Mean Square Energy (RMSE), (5) Spectral-based, (6) Tonnetz, and (7) Chroma-based techniques [47].

3.1.1 Mel-frequency cepstral coefficients (MFCC)

Mel-Frequency Cepstral Coefficients (MFCC) is the most common feature extraction technique used for extracting audio features and graphical features [48]. In MFCC, the signal is being framed and the Hamming window is used to reshape the signal to a very small window [49]. Figure 2 shows the steps of extracting the MFCC features [50]. The MFCC uses the Discrete cosine transform (DCT) internally. If the DCT type is 3, then it is named MFCC with the HTK-style while if the DCT type is 2, then it is named MFCC with the Slaney-style [51].

3.1.2 Mel-spectrogram (MS)

Mel-Spectrogram (MS) is one of the most efficient techniques for audio processing, extracting features from audios, and transferring them into feature images [52]. Figure 3 shows the steps of extracting the MS features [53].

3.1.3 Zero-crossing rate (ZCR)

Zero-Crossing Rate (ZCR) is one of the feature extraction techniques in which the signal changed from positive to zero to negative or vice versa for recognizing the voiced and unvoiced signals. ZCR is based on the idea of counting the times where the waves go from positive to negative or vice versa at a specific time [54]. Equation 1 shows how to calculate the ZCR value.

$$\mathrm{ZCR} = \frac{1}{(2 \times M)} \times \sum_{k=1}^{M} |\mathrm{sign}(a[k]) - \mathrm{sign}(a[k-1])| \quad (1)$$

where k is an index, M is the size and $sign(a[k])$ can be calculated using Eq. 2.

$$\mathrm{sign}(a[k]) = \begin{cases} 1, & \text{if } k \geq 0 \\ -1, & \text{if } k < 0 \end{cases} \quad (2)$$

3.1.4 Chroma-based techniques

There are many chroma-based techniques but the used ones in the current study are (1) chroma-only, (2) Short-Time Fourier Transform (STFT), (3) Constant-Q chromagram Transform (CQT), and (4) Chroma Energy Normalized Statistics (CENS) techniques. *Short-Time Fourier Transform (STFT)* is the sequence of Fourier transforms for an audio signal that allows performing time-frequency analysis for the situations in which signals frequency components change over time. It is a fixed resolution method for analyzing fixed signals and segmenting them into time intervals to take the Fourier Transform for every segment in the signal [55].

Constant-Q chromagram Transform (CQT) is the wavelet transform technique that transforms the time domain signal to the time-frequency domain. The center frequencies of frequency bins are spaced and their Q-factors are equal. The frequency resolution will be better for low frequencies whereas the time resolution is better for high frequencies. The CQT has a better result when the

logarithmic frequency mapping and low frequencies are being concerned [56]. *Chroma Energy Normalized Statistics (CENS)* is the group of scalable sound features utilized for the sound matching process. It computes the short-time energy spread signal. CENS is used for extracting chroma features that capture the melodic and harmonic characteristics of sounds and represent a short time window of the sound [57].

3.1.5 Root mean square energy (RMSE)

It is the square root of the average of the summation of signal amplitude for the short time sound wave energy. RMSE plays an essential role as a loudness indicator. The higher the energy, the louder the sound. RMSE has been utilized in sound segmentation and genre classification [58]. Computing the RMSE value from the voice records is faster as it does not require any STFT calculations. However, using a spectrogram can give a more accurate representation of the energy over time as its frames can be windowed. Equation 3 shows how to calculate the RMSE value.

$$\mathrm{RMSE} = \sqrt{\frac{1}{N} \times \sum_{k=1} N x_k^2} \tag{3}$$

where N is the number of samples and x is the sampled signal.

3.1.6 Tonnetz

Tonnetz is used to analyze the tonal centroid features and audio signals to learn the features that are being extracted from the audio files [59].

3.1.7 Spectral-based techniques

There are many spectral-based techniques but the used ones in the current study are (1) spectral centroid, (2) spectral bandwidth, (3) spectral contrast, (4) spectral flatness, and (5) roll-off frequency techniques. *Spectral Centroid* is the measure of the spectral shape and position. It measures the shape of the spectrum for waves to characterize it. It predicts the sound brightness and the frequency band where most of the energy is concentrated. Hence, the high value of spectral centroid refers to the more signal energy to be concentrated within the higher level of frequencies [60].

Spectral Bandwidth is the spectral range of interest around the centroid. It is derived from the spectral centroid. The bandwidth is directly proportional to the spreading energy around the frequency bands. Also, it is the weighted distances mean of frequency bands that are derived from the spectral centroid [61]. *Spectral Contrast* is the difference between valleys and peaks in the spectrum. It contains more spectral information and represents the relative spectral characteristics. It makes sound normalization and keeps the most peaks for a sound signal constant whereas making valleys attenuation in the spectrum [62].

Spectral Flatness is the estimation of audio spectrum characterization and signal energy distribution uniformity and noisiness of energy spectrum in the frequency domain. If the spectral flatness has a high value then it indicates that the spectrum has the same energy for all spectrum bands. Also, if the spectral flatness is low this means that the spectral energy has low uniformity in the frequency domain [63]. *Roll-off Frequency* is the frequency in which 95% of the energy for each signal is below that frequency. It is used to differentiate between unvoiced and voiced speech. The unvoiced speech has a high level of energy in the high frequency of spectrum [64].

3.2 Machine learning algorithms

ML algorithms are programs with a specific style of adjusting the parameters (i.e., weights) and have feedback depending on their previous experience in predicting the related dataset [65]. In this work, five ML algorithms are deployed to detect heart disease existence. They are (1) KNN, (2) DT, (3) AB, (4) RFC, and (5) Extra Trees Classifier (ETC) ML algorithms.

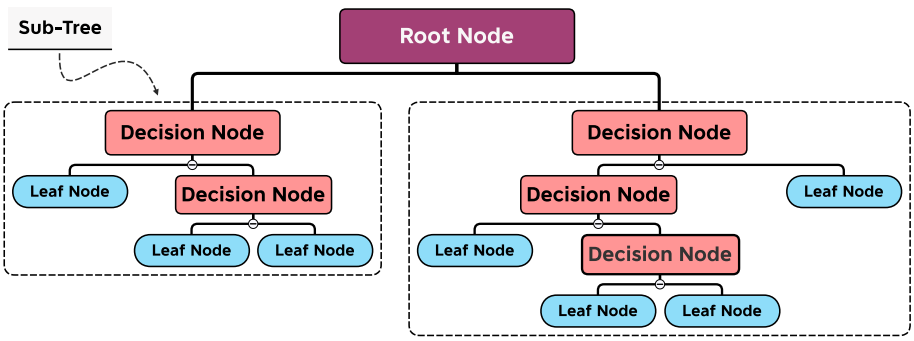

Fig. 4 A sample of the decision trees (DT) with its components

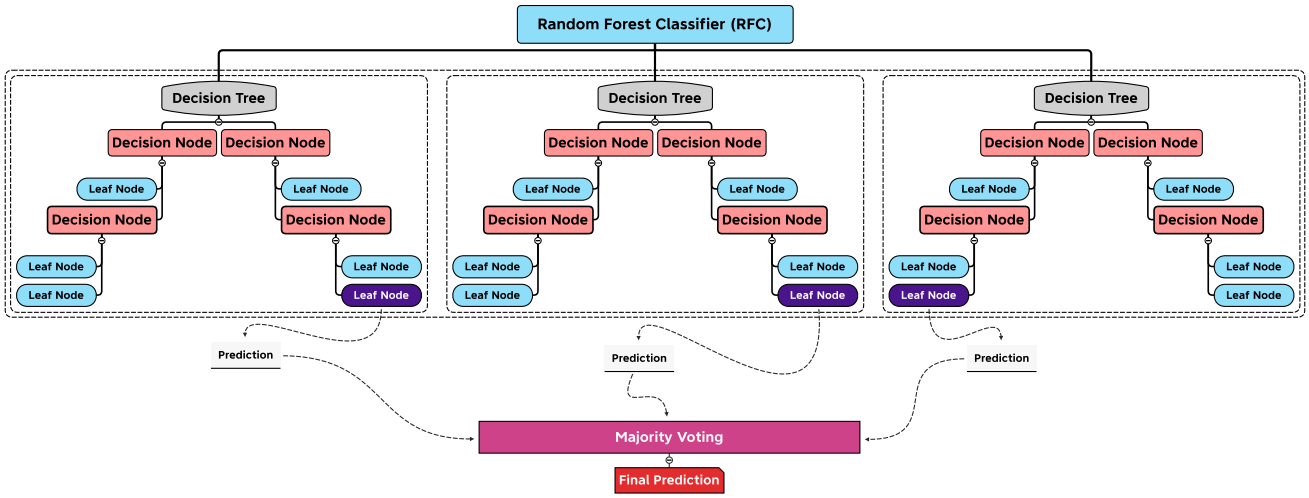

Fig. 5 A Sample of the Random Forest Classifier (RFC) with its Components

3.2.1 K-nearest neighbour (KNN)

It is one of the most used ML algorithms for versatile problems such as regression and classification problems but it is commonly utilized in classification cases [66]. KNN is one of the most simple and easy algorithms to be implemented. However, it is computationally expensive [67]. The working idea of KNN is based on storing the available cases and classifying new ones referring to the majority votes of its *k*-neighbors [68]. It is measured by the distance function to find the distances between a query and all cases in the data. After that, it chooses the closest one to the query and takes the most frequent label between them [69]. The "*k*" in the KNN algorithm represents the nearest neighbors numbers that are utilized for dealing with new cases. If the "*k*" value is high, then it overlooks the cases with a little sample. If the "*k*" value is low, this means it can be referring to the outliers [70].

3.2.2 Decision trees (DT)

It is a decision support technique that has a structure like a tree and consists of three parts. They are (1) leaf nodes, (2) root nodes, and (3) decision nodes [71]. The algorithm splits the training dataset into various branches that segregate to other branches. The nodes in the DT represent the attributes for predicting the outcome and the decision nodes provide the link into the leaves (as shown in Fig. 4).

The decision nodes and root nodes represent the features in the dataset [72]. Hence, the DT algorithm provides various outputs and the highest one will be selected as a final output. From the DT algorithm, a model, that can predict the target variable value from learning decision rules inferred from training data, can be built [73]. The tree representation of the algorithm helps to understand the problem and reach the most optimum solution. So, it is represented as one of the easiest and simplest models for implementation [71].

3.2.3 Random forest classifier (RFC)

It is an ML algorithm representing a collection of DTs so it combines multi-DT outputs to have the most accurate single solution. If a new case that depends on the attributes of DT is required to be classified, each tree in it will give a classification and say votes for this case. Then, the forest selects the highest votes between the trees [74]. RFC algorithm is very flexible, easy to implement and understand, and can achieve a stable prediction output [75]. The algorithm makes the training process based on the bagging method by combining the learning models that will improve the overall result [75]. Figure 5 shows a sample of the RFC and its inner components.

3.2.4 Extra trees classifier (ETC)

It is an ML ensemble algorithm that combines various predictions from many decision trees by averaging them in the case of regression tasks or through utilizing the majority votes for classification problems. ETC is related to Random Forests and bagging. Until the model performance is stable, the number of additional trees is added to increase the performance and aggregate the predictions of various trees to have the most optimum one [76]. ETC is one of the fastest and most accurate ML algorithms that is based on randomization and optimization [77].

3.2.5 AdaBoost (AB)

It is a boosting approach that is utilized as an ensemble algorithm in ML and a supervisory layer for other algorithms. Its work depends on the learning growing sequentially approach. It is done by building the model from training data and making another model that attempts to correct any errors that occurred on the first model. Then, the models are mixed until the training set makes the prediction efficiently or the maximum number of models are inserted [78]. AdaBoost changed the set of weak classifiers to the strong classifiers and predictions were performed based on the average weights of the weak classifiers. AdaBoost depends on the stump performance by updating weights and changing the training set depending on the result of previous ones [79].

3.3 Convolutional neural network (CNN)

Convolutional Neural Network (CNN) is classified as one of the most deep learning powerful tools that take an input image, extract features from it using filters (or kernels), and transfer it to lower dimensions without losing any information. CNN demonstrated its ability for classifying images effectively so it is the most popular one used for that as it can learn the intrinsic and latent image features [80]. CNN model has multi-layers starting by the input layer to convolutional layer, pooling layer, fully connected layer, batch normalization layer, activation layer, and ending with the output layer [81]. A CNN's architecture is composed of multilayers as follows:

Input Layer contains the input image and holds its pixel values. *Convolutional layer* is applied to the input image and extracts different levels of features using kernels and filters that have specific widths and heights. It will determine the output neurons that are connected to a specific region of the input data. Multiple convolution operations are applied by sliding the filters on the input to extract various features levels from the image and stack them to make the convolutional layer output [82]. *Pooling layer* down-samples the input image and reduces the parameters existing in the image aiming to decrease the training time and reduce the overfitting without losing important data. It can also affect the performance of the training process [83, 84].

Fully connected layer (FC) represents a flattened feed-forward layer that aids for classification processes after the pooling process. After the down-sampling and feature extraction processes, nonlinear combinations of features are learned as the output of the convolutional layer. All neurons in the fully connected layer are connected to the neurons in the last and next layer. It can have a nonlinear activation function to make predictions and classify the input data to different classes [85, 86]. *Batch normalization layer* is one of the main layers in the CNN architecture that makes the model perform better and the training process faster by (1) allowing an extensive range of learning rates and (2) re-parametrizing the optimization problems that lead to the process being more stable, smoother, and avoid the local minimum convergence [87]. *Activation layer* is required to get the output of the node through using one of the different activation or transfer functions including Sigmoid, Hyperbolic Tangent (Tanh), Rectified Linear Unit (ReLU), Leaky ReLU, Exponential Linear Unit, Scaled Exponential Linear Unit, and SoftMax functions [88, 89].

3.3.1 Transfer learning

It is representing an ML concept where the pre-trained model that is used for a specific task is reused for another task. It can be summarized as knowledge transfer [90]. The main idea about reusing the pre-trained models for a new task is to have a starting point and have a lot of labeled training data in a new one that did not have much data instead of building the model from scratch and creating these labeled data which is very expensive [91]. Transfer Learning is very popular in the DL field through its advantages including better performance and saving much time during the training process that can lead to rapid progress [92]. Many pre-trained CNN models were trained on the ImageNet image database [93] but the used ones in the current study are VGG16, VGG19, ResNet50, ResNet101, MobileNet, MobileNetV2, MobileNetV3Small, and MobileNetV3Large.

VGG is one of the popular pre-trained models that is used for image classification because of its simplicity. There are many different versions of the VGG architecture published by Oxford University researchers [94]. Although of the model simplicity, that is very expensive in computational and memory cost. The current study uses two types of VGG depending on the layers which are VGG16 and VGG19. VGG has a competitive advantage over other models representing in using only 3×3 convolution filters. VGG model achieved 9.9% top-five error on ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [95]. *ResNet* is a pre-trained deep residual network model proposed by the Microsoft Research team on the "Deep Residual Learning for Image Recognition" [96]. ResNet is a very deep, easy optimization model, and can increase the accuracy and depth of the model. It used forward and backward propagation techniques and the ReLU activation function [97]. ResNet achieved a 7.8% top-five error on ILSVRC.

MobileNet is one of the recently proposed pre-trained models with many modifications and advantages over the

previous models. It is proposed by the Google Research team [98, 99]. It is suitable for mobile and embedded applications. It consists of 2 blocks with 3 layers on each block including a residual block with one stride and other blocks with two strides. It is utilized depth-wise separable convolution modules [100]. It can handle many tasks at the same time and it is considered as the smallest memory size compared to other models. So, it is simple with no complexity or many parameters that can affect the overall performance. The model achieved a 0.901 top-five accuracy on ILSVRC [101]. Table 3 compares the discussed and used pre-trained CNN models.

3.3.2 Parameters optimization

It represents an expanded method for changing parameters (i.e., weights) of the model to report better accurate results and reduce the losses [103]. The used parameters optimizers in the current study are Adam [104], NAdam [105], AdaGrad [106], AdaDelta [107], AdaMax [108], RMSProp [109], SGD [110], Ftrl [111], SGD Nesterov [112], RMSProp Centered [113], and Adam AMSGrad [114].

3.3.3 Hyperparameters

Loss Function has a critical role in evaluating the proposed solution and calculating the model errors [115]. So, how good the model is determined, to try to change the parameters, to improve the model performance and minimize the overall loss. It can be started as the penalty for failing to reach the desired output so if the deviation in the predicted value by the model from the desired value is large, then the function will give a high loss value and a smaller number otherwise [116]. The used losses in the current study are Categorical Crossentropy [117], Categorical Hinge [118], KLDivergence [119], Poisson [120], Squared Hinge [121], and Hinge [122].

Batch Size represents the number of data records that are utilized to train the model in every iteration to ensure the model generalization, parameters value, and the convergence of loss function. It plays an important role in the learning process of the model by making it quicker and more stable [123]. *Dropout* is a regularization technique used for training the CNN model on any or all hidden layers of the architecture. It plays an important role in preventing the overfitting problem and addressing it to keep the performance at an optimum level. It can improve the generalization efficiency in all the data by setting randomly the output to be 0 for the given neuron [124].

3.4 Metaheuristic optimization using aquila optimizer (AO)

It is one of the most popular choices for optimization, modeling, and solving complex problems that are difficult to be solved using the traditional ways. "Meta" in Metaheuristic refers to the higher level that performs better than the simple heuristics. It utilizes a tradeoff for the global exploration and local search [125]. Metaheuristic algorithms have essential parts which are diversification and intensification. Diversification generates various solutions for exploring the search space whereas intensification focuses on the search in a local region through exploiting information where the good solution is found in this region. Metaheuristic optimization is utilized to find the optimal solution for many optimization problems that are very challenging functions based on the correct use of the optimum algorithm for this case [126].

Aquila Optimizer (AO) is a novel metaheuristic optimization method. The optimization process for the AO algorithm is presented in four ways which are (1) choosing the search space through high soar by the vertical stoop (Eq. 4), (2) discovering the various search space through contour flight by short glide attack (Eq. 5), (3) swooping by grabbing prey and walk (Eq. 7), and (4) exploiting through converge search space by low flight by descent attack (Eq. 6).

Table 3 Comparing the used pre-trained CNN models

Name	Year	Number of parameters	Size (MB)
VGG16 [94]	2014	138,357,544	528
VGG19		143,667,240	549
ResNet50 [96]	2016	25,636,712	98
ResNet101		44,707,176	171
MobileNet [98, 99]	2017 and 2018	4,253,864	16
MobileNetV2		3,538,984	14
MobileNetV3Small [102]	2021	2,500,000	21
MobileNetV3Large		5,400,000	–

$$X(t+1) = X_{\text{best}}(t) \times \left(1 - \frac{t}{T}\right) + \left(\frac{\sum_{i=1}^{N} X(t)}{N} - X_{\text{best}}(t) \times \text{rand}\right) \tag{4}$$

$$X(t+1) = X_{\text{best}}(t) \times \text{Levy}(D) + X_R(t) + (r_1 + U \times D_1) \times (\cos(-\omega \times D_1 + 1.5 \times \pi) - \sin(-\omega \times D_1 + 1.5 \times \pi)) \times \text{rand} \tag{5}$$

$$X(t+1) = \left(X_{\text{best}}(t) - \frac{\sum_{i=1}^{N} X(t)}{N}\right) \times \alpha - \text{rand} + ((UB - LB) \times \text{rand} + LB) \times \sigma \tag{6}$$

$$X(t+1) = QF \times X_{\text{best}}(t) - X(t) \times \text{rand} \times (2 \times \text{rand} - 1) - \text{Levy}(D) \times 2 \times \left(1 - \frac{t}{T}\right) + \text{rand} \times (2 \times \text{rand} - 1) \tag{7}$$

where $X(t+1)$ is the solution of the next iteration, N is the population size, t is the iteration number, T is the total number of iterations, rand is a random number in the range [0, 1], $X_R(t)$ is a random solution in the current iteration t, $X_{\text{best}}(t)$ is the best solution in the current iteration t, D is the dimension space size, $\text{Levy}(D)$ is the levy flight distribution function, r_1 is a value in the range [1, 20], U equals 0.00565, D_1 is a value in the range [1, D], QF is the quality function, α (and σ) equal to 0.1, UB is the upper bound, and LB is the lower bound. The fixed values are taken from the original AO paper.

The optimization procedures in AO start by generating a random predefined set of candidate solutions that are called population and by the repetition trajectory, the AO search strategies explore the positions of the best solution (or the near-optimal one). Every solution updates its position depending on the best solution in the optimization procedure of the AO [127]. The series of experiments are conducted to enable the AO to validate the optimizer's ability to find the best solution for various optimization tasks. AO performance can be enhanced through combining it with a flight, mutation, levy, stochastic (and evolutionary) components, and global (or local) search [128].

3.5 Image augmentation

Data Augmentation (DA) is a process of augmenting the dataset and increasing it to a large, rich, and diverse one [129]. DA can increase the performance of the CNN model through generalization and the variety of the data that enables the model to detect or classify any objects on the image in different orientations and dimensions [130]. This process is representing a pre-processing step as it is applied only to the training subset of the dataset to increase its size and variations. DA can be performed using different transformation techniques including (1) flipping the image, (2) zooming it in or out, (3) rotating the image by a specific degree, (4) shifting the image, (5) cropping the image, (6)

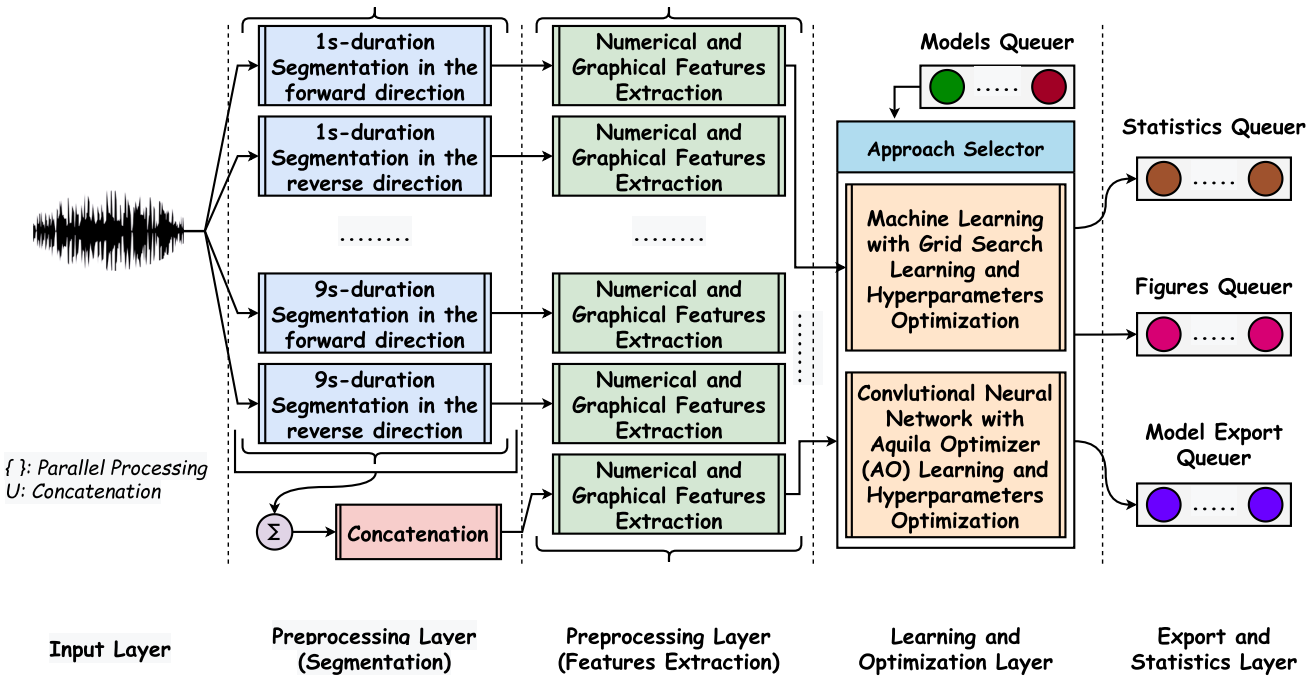

Fig. 6 The 4-Phases Suggested Framework Flow Summarization

Table 4 The used dataset classes and the corresponding number of records

Category	Number of Records
Murmur	200
Normal	462
Artifact	80
Extra Heart Sound	38
Extrasystole	92
Total	872

changing the brightness of the image, and (7) shearing the image horizontally (or vertically) [131].

Flipping can be done by flipping the image vertically (or horizontally) depending on the object's location on the image. Rotation can be done by rotating the image to a specific degree. Shearing is done by shifting any part of the image. Cropping can be applied by removing any columns (or rows) of pixels from the image to see the object in different locations. Shifting is done by moving the pixels of width (and height) of the image in only one direction vertically (or horizontally) without affecting the dimensions of the image. Brightness changing is performed by changing the image and making it lighter (or darker) to enable the model to recognize different lighting levels in the image. Zooming is done by zooming the image in (or out) within a specific range. Also, it can be applied to each axis of the image independently [132].

3.6 Data normalization (DN)

Data Normalization (DN) is one of the pre-processing techniques that change the attribute value to a known range or scale to improve the performance of ML algorithms. There are different DT techniques but the used ones in the current study are (1) Standard Scaler, (2) Min-Max Scaler, (3) Max-Abs Scaler, and (4) Normalization [133].

3.6.1 Standard scaler

It is one of the DN techniques deployed on the vectors. It standardizes the features by making the mean equal to zero and scaling each vector into the unit variance. Equation 8 shows how to calculate it.

$$\text{out} = \frac{\text{in-mean}}{\text{std}} \tag{8}$$

where out is the output image, in is the input image, mean is the mean value and std is the standard deviation.

3.6.2 Min-max scaler

It transforms the dataset values into a range between 0 and 1 where the smallest value is normalized into 0 and the largest value is normalized into 1. Equation 9 shows how to calculate it.

$$\text{out} = \frac{\text{in} - \text{in}_{\min}}{\text{in}_{\max} - \text{in}_{\min}} \tag{9}$$

where in_{max} is the maximum value and in_{min} is the minimum value.

3.6.3 Max-abs scaler

It is similar to the min-max scaler except the values are mapped into the range between 0 and 1 as it scales and translates the data features to the range between -1 and 1 by dividing it by the maximum absolute value. The maximum value for any feature equals 1. Equation 10 shows how to calculate it.

$$\text{out} = \frac{\text{in}}{|\text{in}_{\max}|} \tag{10}$$

3.6.4 Normalization

It is deployed by squeezing the data between 0 and 1 It is very useful in classification and data containing negative values [134]. Equation 11 shows how to calculate it.

$$\text{out} = \frac{\text{in}}{\text{in}_{\max}} \tag{11}$$

4 Suggested approach

The current study suggests a framework for heart disease learning and optimization. It is divided into four major phases (or layers). They are (1) dataset collection, (2) pre-processing (segmentation and features extraction), (3) learning and hyperparameters optimization, and (4) export and statistics phases. The framework flow is summarized in Fig. 6

In summary, the input layer accepts the voice records. These records flow sequentially to the pre-processing phase which is partitioned into two sub-phases. Its target is to segment the records into sub-records with equal time durations and extract the features from them numerically and graphically. These features and graphs are the inputs of the third phase. Its role is to learn and optimize the selected model. The optimized model will be exported in the last phase. Also, training, validation, and testing statistics and figures are exported in that phase. The phases are discussed in the following subsections.

4.1 Dataset collection phase

Classifying Heart Sounds Challenge Dataset [135] is the used dataset in the current study. It contains two challenges. The authors combined both of them into one dataset. It has five classes (i.e., categories). They are (1) Murmur, (2) Normal, (3) Artifact, (4) Extra Heart Sound, and (5) Extrasystole. The data reflects voice records with the extensions "wav", "aif", and "aiff". Table 4 shows the categories and the corresponding number of records.

4.2 Pre-processing phase

The dataset is pre-processed in two sub-phases. The first sub-phase is segmenting the records into sub-records with equal time durations. The second sub-phase is to extract the features numerically for the ML used techniques and graphically for the pre-trained CNN models.

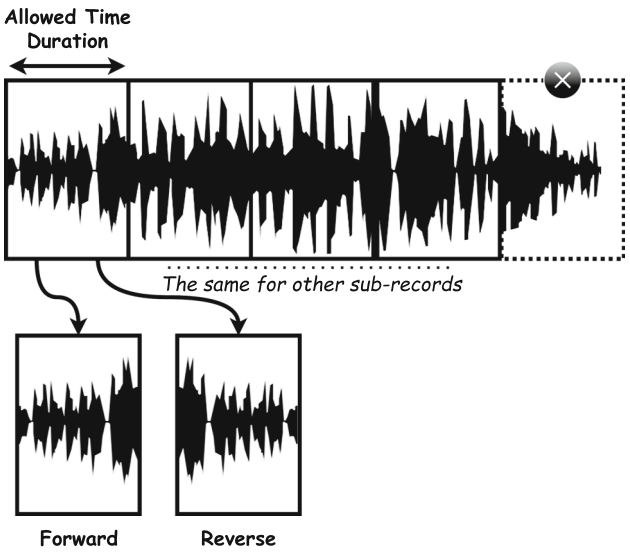

Fig. 7 Example on the segmentation process on a single record for a specific allowed time duration

4.2.1 Voice segmentation

The records should be segmented to a fixed time duration such as 1 s or 3 s. The suggested approach in the current study is to segment the records in different time durations in both directions and concatenate them. *How does this happen?* For each record in the dataset, a specified time window moves from the beginning on it and segment the record into sub-records. For example, if the record's duration is 9 seconds and the allowed time window is 1 second. Then, there are 9 generated sub-records. Also, if the allowed time window is 2 seconds. Then, there are 4 generated sub-records. *What about the remaining small-time segment?* It is ignored. In the last example, there are only 4 generated sub-records and hence the remaining 1 second is neglected as it is smaller than the allowed time window (i.e., 2 seconds). *How to get the number of segments?* Eq. 12 shows how to get the number of segments for a record.

$$\mathrm{no}_{\mathrm{segments}} = \left\lfloor \left(\frac{\mathrm{duration}_{\mathrm{record}}}{\mathrm{duration}_{\mathrm{window}}} \right) \right\rfloor \tag{12}$$

Algorithm 1 shows the followed pseudocode function during the segmentation process for a single record.

What is the suggested allowed time durations for segmentation? It is worth mentioning that the current study

Table 5 The used feature techniques and the corresponding number of extracted numerical features

Technique	Number of Extracted Features
MFCC HTK-Style 20	20
MFCC Slaney-Style 20	20
MFCC HTK-Style 40	40
MFCC Slaney-Style 40	40
ZCR	1
Spectral Centroid	1
Spectral Bandwidth	1
Spectral Contrast	7
Spectral Flatness	1
Roll-Off Frequency	1
RMSE	1
Tonnetz	6
Harmoic Tonnetz	6
Mel-Spectrogram	128
Chroma Only	12
Chroma CQT	12
Chroma STFT	12
Chroma CENS	12
Total	321

Algorithm 1: The Followed Pseudocode During the Segmentation Process

Input: *record*, *recordDuration*, *windowDuration* // Audio record, Record duration, and Window duration
Output: *segments* // The generated segments
$noOfSegments = \left\lfloor \left(\frac{recordDuration}{windowDuration} \right) \right\rfloor$ // Getting the number of segments
$segments = []$ // Initialize the Segments' Array
$i = 1$ // Initialize the Segments' Counter
while $(i \leq noOfSegments)$ **do**
  $fromOffset = i \times windowDuration$ // Calculate the starting index
  $to = (i+1) \times windowDuration$ // Calculate the ending index
  $segment =$ **CutSegment** $(record, fromOffset, to)$ // Get a segment from the full record from the offset with a specified window duration
  // Check if the duration of the cut segment is less than the required window duration of not
  if $(\textbf{\textit{GetDuration}}(segment) < windowDuration)$ **then**
    return *segments* // Ignore the current cut and return the generated segments
  Append $(segments, segment)$ // Append the cut segment into the generated segments
  $i = i+1$ // Increment the Segments' Counter
return *segments* // Return the generated segments if the loop is completed

suggests segmenting each record with 1s, 3s, 5s, 7s, and 9s time durations in both directions. Also, the whole segmented records are concatenated in a single dataset. *How the segmentation is done in both directions?* For each sub-record, the voice is reversed. Hence, two sub-records from a single one can be generated. Figure 7 summarizes this process. The overall number of generated datasets is 11 (i.e., 2 for each time duration and 1 concatenated).

4.2.2 Numerical features pre-processing

The numerical features are extracted from each segment in each record. The used numerical voice features extraction techniques in the current study are (1) MFCC (HTK-style and Slaney-style), (2) Mel-Spectrogram, (3) ZCR, (4) RMSE, (5) Spectral-based (spectral centroid, spectral bandwidth, spectral contrast, spectral flatness, and roll-off frequency), (6) Tonnetz (normal and harmonic), and (7) Chroma-based (chroma-only, STFT, CQT, and CENS) techniques. Table 5 shows the used feature techniques and the corresponding number of extracted numerical features.

The segmentation process, as mentioned, is applied on each record for the time windows 1, 3, 5, 7, and 9 in both directions (i.e., forward and reverse). Also, the whole segmented records are concatenated in one dataset. The number of generated numerical datasets is 11. Algorithm 2

Table 7 The generated images for each class

Category	Number of images
Artifact	600
Extra heart sound	196
Extrasystole	359
Murmur	1044
Normal	1437
Total	3636

Table 6 The generated numerical datasets with the correspond number of records

Segment duration (s)	Direction	Number of numerical records	File name
1	Forward	4668	1N
1	Reverse	4668	1R
3	Forward	1348	3N
3	Reverse	1348	3R
5	Forward	572	5N
5	Reverse	572	5R
7	Forward	398	7N
7	Reverse	398	7R
9	Forward	222	9N
9	Reverse	222	9R
Concatenated	Both	14,416	Concatenated

shows the followed pseudocode function during the numerical features extraction process for all records. In the pseudocode.

Table 4 shows the generated numerical datasets and the corresponding number of records per each.

Algorithm 2: The Followed Pseudocode to Extract the Numerical Features for All Records

```
Input: records                                                          // Audio records
Output: datasets                                                        // The generated datasets
datasets = []                                                           // Initialize the Datasets' Array
while (windowDuration ∈ [1,3,5,7,9]) do
    i = 1                                                               // Initialize the Records' Counter
    forwardDataset = []                                                 // Initialize the Forward Dataset' Array
    reverseDataset = []                                                 // Initialize the Reverse Dataset' Array
    while (i ≤ Count(records)) do
        record = records[i]                                             // Get the current record
        segments = SegmentRecord(record, recordDuration, windowDuration)  // Segment the record (discussed in the previous subsection)
        j = 1                                                           // Initialize the Segments' Counter
        while (j ≤ Count(segments)) do
            forwardFeatures = ExtractNumericalFeatures(segment[j])      // Extract the numerical 321 features in the forward direction (discussed in the current subsection)
            reverseFeatures = ExtractNumericalFeatures(Reverse(segment[j]))  // Extract the numerical 321 features in the reverse direction (discussed in the current subsection)
            Append(forwardDataset, forwardFeatures)                     // Append the forward features into the forward dataset
            Append(reverseDataset, reverseFeatures)                     // Append the reverse features into the reverse dataset
            j = j + 1                                                   // Increment the Segments' Counter
        i = i + 1                                                       // Increment the Records' Counter
    Append(datasets, forwardDataset)                                    // Append the forward dataset into the datasets
    Append(datasets, reverseDataset)                                    // Append the reverse dataset into the datasets
concatenated = ConcatenateAll(datasets)                                 // Concatenate all datasets into a single one
Append(datasets, concatenated)  // Append the concatenated dataset into the datasets to set the number of generated datasets to be 11 as discussed
return datasets                                                         // Return the generated datasets if the process is completed
```

4.2.3 Graphical features pre-processing

The graphical features are extracted as images from each segment (i.e., sub-record) in each record similar to the numerical one. The used graphical voice features extraction techniques in the current study are (1) MFCC (HTK-style and Slaney-style), (2) Mel-Spectrogram, (3) Spectrogram, and (4) STFT. Table 7 shows the number of generated images for each category. The number of generated images for each technique is the same. Fig. 8 shows samples from each category for each technique.

4.3 Learning and optimization phase

Algorithm 3 shows the pseudocode of the learning and optimization processes using the pre-trained CNN models and ML algorithms. It accepts three inputs (1) the selected model, (2) the dataset, and (3) the experimental configurations (from Table 9 that will be discussed in the experiments section). Inside it, it (1) split the dataset into training, testing, and validation subsets, (2) checks if the model is an ML algorithm or not, (3) if the model is an ML algorithm, it applies the grid search optimization algorithm to find the best combination that will lead the ML model to the top-1 performance metrics, (4) if the model is a pre-trained CNN model, it applied the AO metaheuristic optimizer to find the best solution that will lead the ML model to the top-1 performance metrics.

4.4 Export and statistics phase

In the current phase, the optimized model is exported to be used in the future or production. Different statistics are calculated such as accuracy, precision, and F1-score. Learning curves and figures are generated and stored. The current study calculates different state-of-the-art

Algorithm 3: The Suggested Hybrid Learning and Optimization Pseudocode

Input: $model, dataset, configs$ // Model name, Dataset (i.e., images or numerical data), and Experimental vonfigurations
Output: $statistics, figures, models$ // The statstictis, figures, and models to be stored
$trainX, trainY, validationX, validationY, testX, testY =$ **SplitDataset**$(dataset, configs[SPLIT\ RATIO])$ // Split the dataset into training, testing, and validation subsets concerning the split ratio
// Check if the model belongs to the ML algorithms or not
if $(model \in configs[ML\ MODELS])$ **then**
  $modelToTrain =$ **CreateInitialMLModel**$(model)$ // Create the initial ML model
  $solutions =$ **GridSearchForBest**$(trainX, trainY, validationX, validationY, modelToTrain, configs)$ // Apply the grid search optimization to find the best combination
  $metrics =$ **CalculateMetrics**$(modelToTrain, solutions, testX, testY, configs)$ // Calculate the performance metrics
  $figures =$ **GenerateFigures**$(modelToTrain, solutions, testX, testY, configs)$ // Generate the figures
else
  $modelToTrain =$ **CreateInitialCNNModel**$(model)$ // Create the initial pre-trained CNN model
  $population =$ **GenerateInitialPopulation**$(configs)$ // Generate the initial population for the model
  $i = 1$ // Initialize the iterations' counter
  // Apply the AO optimization process for the specified number of terations
  while $(i \leq configs[NO\ OF\ ITERATIONS])$ **do**
    $scores =$ **CalculateScores**$(modelToTrain, population, trainX, trainY, validationX, validationY, configs)$ // Calculate the score (i.e., accuracy) for each solution in the population
    $population =$ **UpdatePopulation**$(population, scores, configs)$ // Update the scores using AO optimizer and set it to the population
    $i = i + 1$ // Increment the iterations' counter
  $metrics =$ **CalculateMetrics**$(modelToTrain, population, testX, testY, configs)$ // Calculate the performance metrics
  $figures =$ **GenerateFigures**$(modelToTrain, population, testX, testY, configs)$ // Generate the figures
StoreMetricsAndFigures$(metrics, figures, configs)$ // Store the calculated performance metrics and generated figures

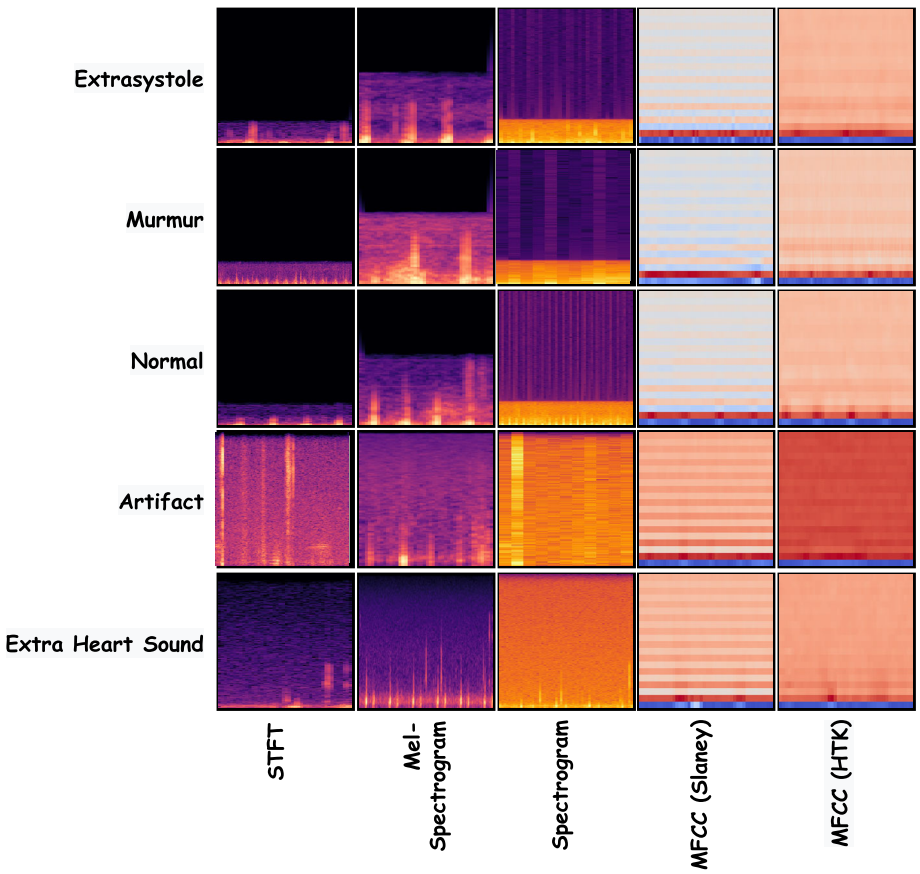

Fig. 8 Samples from the extracted images for each technique and class

Table 8 Summarization of the Performance Metrics

		Predicted class		
		Positive	Negative	
Actual class	Positive	True positive (TP)	False negative (FN)	Sensitivity (recall or total positive rate) $TPR = \frac{TP}{TP+FN}$
	Negative	False positive (FP)	True negative (TN)	Specificity (true negative rate) $TNR = \frac{TN}{TN+FP}$
		False discovery rate $FDR = \frac{FP}{TP+FP}$	Negative predictive value $NPV = \frac{TN}{TN+FN}$	Accuarcy
		Precision (positive predictive value) $PPV = \frac{TP}{TP+FP}$	False omission rate $FOR = \frac{FN}{TN+FN}$	$ACC = \frac{TP+TN}{TP+TN+FP+FN}$

performance metrics. They are accuracy, F1-score, recall, specificity, the area under the curve (AUC), sensitivity, intersection over union (IoU), Dice coefficient, and precision. They are summarized in Table 8.

5 Experiments and discussions

The experiments are divided into two categories (1) experiments related to the extracted numerical features using the ML algorithms and (2) experiments related to the images and extracted graphs using the pre-trained CNN models.

5.1 Experiments configurations

Generally, Python is the used programming language in the current study. The learning and optimization environments are Google Colab (with its GPU) and Toshiba Qosmio X70-A with 32 GB RAM and Intel Core i7 Processor. Tensorflow, Keras, NumPy, OpenCV, Pandas, and Matplotlib are the used Python packages [136]. The dataset split ratio is set to 85% (training and validation) and 15% (testing). Dataset shuffling is applied. The images are resized to (100, 100, 3) in RGB. Table 9 summarizes the configurations of the experiments.

5.2 ML experiments

The current subsection presents and discusses the experiments related to the extracted 321 numerical features using the mentioned ML algorithms (i.e., DT, AB, RFC, ETC, and KNN). For each ML algorithm, 11 experiments are applied on the 1, 3, 5, 7, 9, and mixed durations in the forward and reverse directions. The algorithms are optimized using the grid search for 5 cross-validation runs, to find the best combinations with the highest metrics. The metrics (i.e., accuracy, precision, recall, and F1-score) are captured and reported. It is worth mentioning that the "files" word refers to the 1, 3, 5, 7, 9, and mixed durations shown in Table 4.

5.2.1 K-nearest neighbor experiment

Table 10 shows the summarization of the reported results related to the KNN experiment. It is sorted in a descending order concerning the accuracy values. It shows that the "Ball Tree" algorithm and "Distance" weights are the best among other variations. The "Max-Abs" scaler is reported as the best one in 7 files while the "0" variance threshold is reported as the best one in 8 files. The maximum reported accuracy, precision, recall, and F1-score are 100%, 100%, 100%, and 100% respectively. The segmentation durations 9 in both directions are the best while the concatenated dataset reported only 99.97%. Figure 9 shows the accuracy, precision, recall, and F1-score curves of the different files.

5.2.2 Decision tree (DT) experiment

Table 11 shows the summarization of the reported results related to the DT experiment. It is sorted in a descending order concerning the accuracy values. It shows that the "Best" splitter and "Entropy" criteria are the best among other variations. The "Normalize" scaler is reported as the best one in 4 files while the "0.001" variance threshold is reported as the best one in 4 files. The maximum reported accuracy, precision, recall, and F1-score are 99.89%, 99.89%, 99.89%, and 99.89%, respectively. The

Table 9 The used experiments configurations

Key	ML, DL, or Both	Range
Dataset sources	Both	Classifying heart sounds challenge dataset [135]
Dataset size	Both	3636 Images and 14,416 extracted numerical features
Categories	Both	Artifact, extra heart sound, extrasystole, murmur, and normal
Number of classes	Both	5
Hyperparameters optimizer	Both	Aquila optimizer (AO) for DL and grid search (GS) for ML
Train split ratio	Both	85%:15%
Shuffle dataset	Both	True
AO population size	DL	10
AO number of iterations	DL	15
Number of epochs	DL	5
Image size	DL	(100, 100, 3)
Output activation function	DL	SoftMax
Early stopping patience	DL	5
Pre-trained parameters initializers	DL	ImageNet
Pre-trained models	DL	VGG16, VGG19, ResNet50, ResNet101, MobileNet, MobileNetV2, MobileNetV3Small, and MobileNetV3Large
Loss	DL	Categorical crossentropy, Categorical hinge, KLDivergence, Poisson, Squared hinge, and Hinge
Parameters optimizer	DL	Adam, NAdam, AdaGrad, AdaDelta, AdaMax, RMSProp, SGD, Ftrl, SGD Nesterov, RMSProp Centered, and Adam AMSGrad
Dropout range	DL	[0, 0.6]
Batch size	DL	4 to 48 with a step of 4
Pre-trained model learn ratio	DL	1 to 100 with a step of 1
Apply data augmentation	DL	True and False
Rotation range	DL	0° to 45° with a step of 1°
Width shift range	DL	[0, 0.25]
Height shift range	DL	[0, 0.25]
Shear range	DL	[0, 0.25]
Zoom range	DL	[0, 0.25]
Horizontal flip range	DL	True and false
Vertical flip range	DL	True and false
Brightness range	DL	[0.5, 2.0]
Scaling techniques	DL	Normalize, Standard, Min Max, and Max Abs
Number of features per Record	ML	321
GS cross validation	ML	5
KNN algorithms	ML	Ball Tree, KD Tree, and Brute
KNN weights	ML	Uniform and distance
DT, ETC, and RFC criterion	ML	Gini and entropy
DT splitter	ML	Best and random
AB, ETC, and RFC No. estimators	ML	50, 100, and 250
Programming language	Both	Python
Learning and optimization environment	Both	Google colab (Intel(R) Xeon(R) CPU @ 2.00GHz, Tesla T4 16 GB GPU with CUDA v.11.2, and 12 GB RAM)
Python packages	Both	Tensorflow, Keras, NumPy, OpenCV, Scikit-Learn, SciPy, Pandas, and Matplotlib

Table 10 Summarization of the reported results of the KNN experiment

File	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Variance threshold	Scaler	Algorithm	Weights
9N	100	100	100	100	0	Max Abs	Ball tree	Distance
9R	100	100	100	100	0	Max Abs	Ball tree	Distance
Concatenated	99.97	99.97	99.97	99.97	0	Standard	Ball tree	Distance
1N	99.85	99.85	99.85	99.85	0.005	Max Abs	Ball tree	Distance
3N	99.70	99.70	99.70	99.70	0	Max Abs	Ball tree	Distance
3R	99.70	99.70	99.70	99.70	0	Standard	Ball tree	Distance
1R	99.61	99.61	99.61	99.61	0	Max Abs	Ball tree	Distance
5N	99.30	99.30	99.30	99.30	0	Min Max	Ball tree	Distance
5R	99.30	99.30	99.30	99.30	0	Standard	Ball tree	Distance
7R	98.99	98.99	98.99	98.99	0.01	Max Abs	Ball tree	Distance
7N	97.99	97.99	97.99	97.99	0.01	Max Abs	Ball tree	Distance

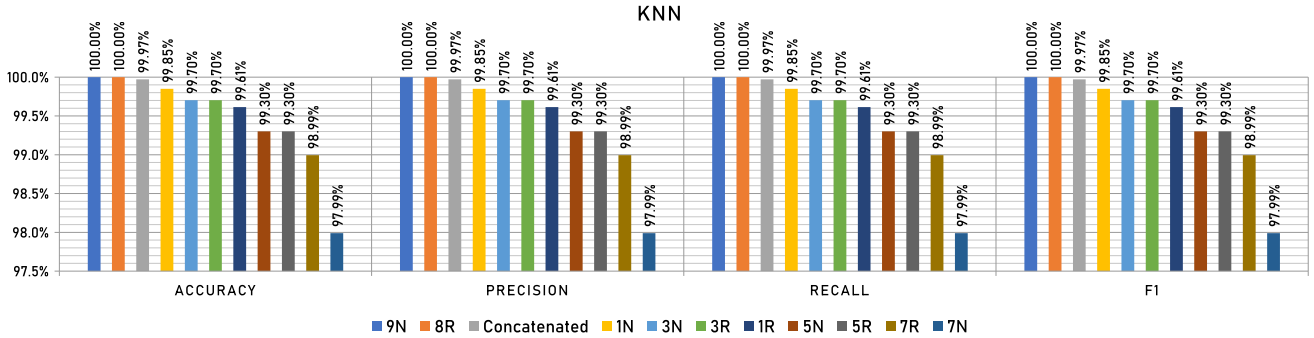

Fig. 9 The accuracy, precision, recall, and F1-score KNN curves of the different files

Table 11 Summarization of the reported results of the DT experiment

File	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Variance threshold	Scaler	Criterion	Splitter
Concatenated	99.89	99.89	99.89	99.89	0.001	Normalize	Gini	Best
3N	99.70	99.70	99.70	99.70	0.01	Min Max	Entropy	Random
9N	99.55	99.55	99.55	99.55	0	Standard	Entropy	Random
5N	99.48	99.48	99.48	99.48	0.01	Min Max	Entropy	Best
1N	99.36	99.36	99.36	99.36	0.001	Normalize	Entropy	Best
1R	99.21	99.21	99.21	99.21	0.001	Normalize	Gini	Best
5R	99.13	99.13	99.13	99.13	0.001	Normalize	Gini	Best
9R	99.10	99.10	99.10	99.10	0.01	Standard	Gini	Best
7N	98.99	98.99	98.99	98.99	0	Min Max	Entropy	Best
3R	98.81	98.81	98.81	98.81	0.005	Standard	Entropy	Random
7R	97.99	97.99	97.99	97.99	0	Max Abs	Gini	Best

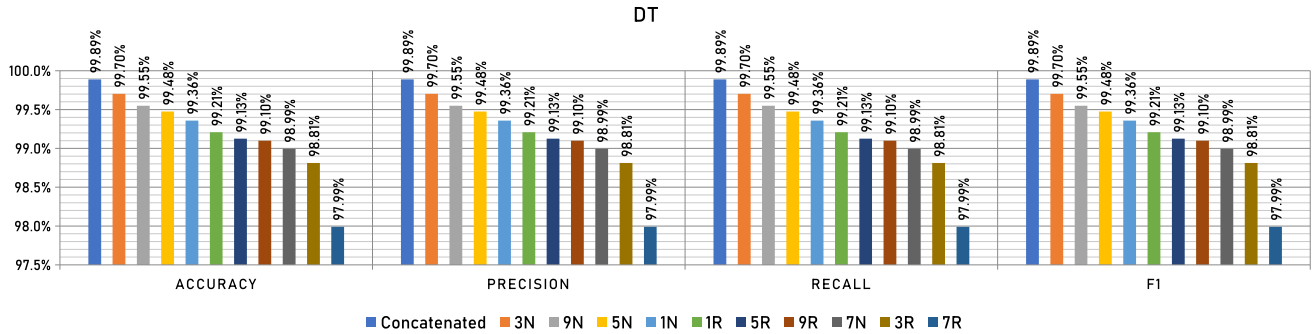

Fig. 10 The Accuracy, precision, recall, and F1-score DT curves of the different files

Table 12 Summarization of the reported results of the AB experiment

File	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Variance Threshold	Scaler	No. Estimators
9N	62.61	62.61	62.61	62.61	0.005	Normalize	100
9R	61.71	61.71	61.71	61.71	0.001	Normalize	50
Concatenated	60.47	60.47	60.47	60.47	0.01	Max Abs	100
1N	59.10	59.10	59.10	59.10	0.01	Min Max	50
7R	56.28	56.28	56.28	56.28	0	Normalize	50
1R	55.74	55.74	55.74	55.74	0.01	Min Max	100
5R	53.15	53.15	53.15	53.15	0	Normalize	100
5N	52.80	52.80	52.80	52.80	0	Normalize	50
3N	50.82	50.82	50.82	50.82	0	Normalize	50
3R	47.18	47.18	47.18	47.18	0.01	Max Abs	250
7N	41.46	41.46	41.46	41.46	0.01	Normalize	50

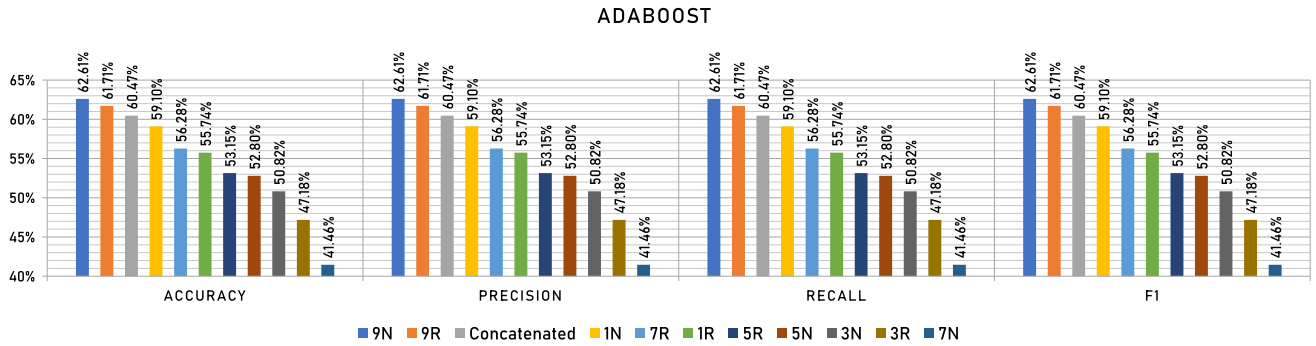

Fig. 11 The accuracy, precision, recall, and F1-score AB curves of the different files

Table 13 Summarization of the reported results of the RFC experiment

File	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Variance Threshold	Scaler	No. Estimators	Criterion
3N	100	100	100	100	0.005	Max Abs	50	Entropy
9N	100	100	100	100	0	Normalize	100	Gini
Concatenated	100	100	100	100	0	Normalize	250	Entropy
3R	99.78	99.78	99.78	99.78	0.01	Max Abs	50	Gini
1N	99.61	99.61	99.61	99.61	0.005	Min Max	250	Entropy
7N	99.50	99.50	99.50	99.50	0.005	Min Max	50	Gini
1R	99.49	99.49	99.49	99.49	0.005	Max Abs	250	Entropy
9R	99.10	99.10	99.10	99.10	0.005	Max Abs	50	Entropy
7R	98.99	98.99	98.99	98.99	0.005	Standard	250	Gini
5R	98.95	98.95	98.95	98.95	0	Min Max	50	Entropy
5N	97.55	97.55	97.55	97.55	0.01	Standard	100	Gini

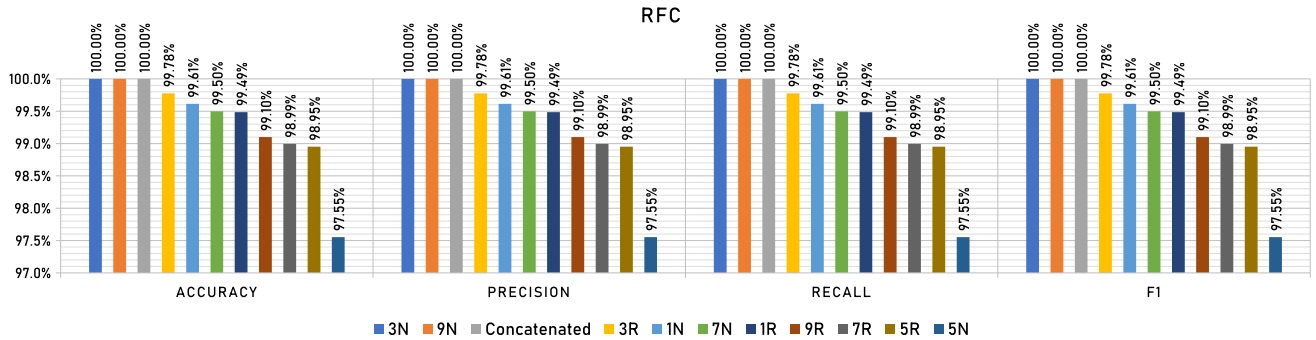

Fig. 12 The accuracy, precision, recall, and F1-score RFC curves of the different files

Table 14 Summarization of the reported results of the ETC experiment

File	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Variance threshold	Scaler	No. Estimators	Criterion
5N	100	100	100	100	0.005	Max Abs	100	Entropy
9N	100	100	100	100	0	Max Abs	100	Entropy
Concatenated	100	100	100	100	0.01	Standard	50	Entropy
3R	99.85	99.85	99.85	99.85	0.01	Max Abs	250	Entropy
1R	99.64	99.64	99.64	99.64	0.001	Standard	50	Entropy
7N	99.50	99.50	99.50	99.50	0	Max Abs	100	Entropy
7R	99.50	99.50	99.50	99.50	0.01	Min Max	250	Entropy
1N	99.49	99.49	99.49	99.49	0	Normalize	50	Gini
3N	99.41	99.41	99.41	99.41	0.01	Max Abs	100	Entropy
5R	99.30	99.30	99.30	99.30	0.005	Min Max	100	Entropy
9R	99.10	99.10	99.10	99.10	0.001	Standard	250	Gini

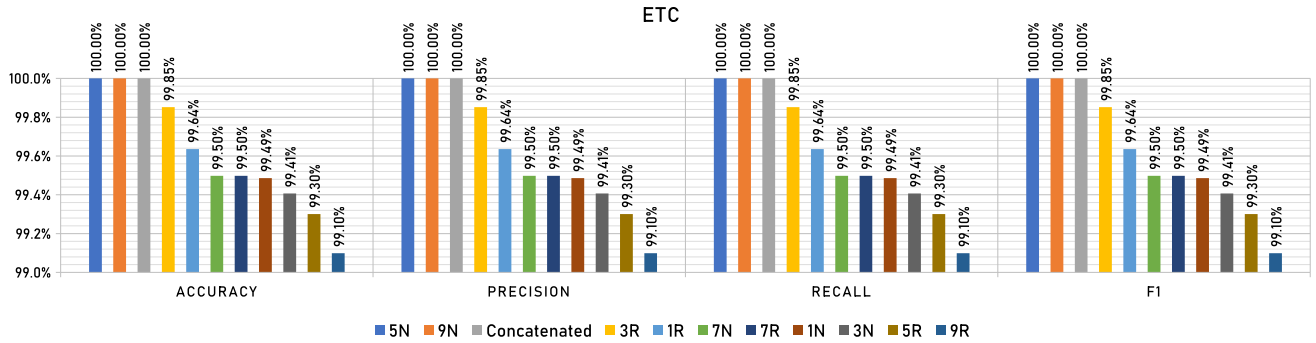

Fig. 13 The accuracy, precision, recall, and F1-score ETC curves of the different files

Table 15 Summarization of the reported results of All Experiment concerning the Top-1 accuracy

Model	File(s)	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)
KNN	9N and 9R	100	100	100	100
DT	Concatenated	99.89	99.89	99.89	99.89
AB	9N	62.61	62.61	62.61	62.61
RFC	3N, 9N, and concatenated	100	100	100	100
ETC	5N, 9N, and concatenated	100	100	100	100

concatenated dataset reported the best dataset among others. Figure 10 shows the accuracy, precision, recall, and F1-score curves of the different files.

5.2.3 AdaBoost (AB) experiment

Table 12 shows the summarization of the reported results related to the AB experiment. It is sorted in a descending order concerning the accuracy values. It shows that the "50" number of estimators is the best among other variations. The "Normalize" scaler is reported as the best one in 7 files while the "0.01" variance threshold is reported as the best one in 5 files. The maximum reported accuracy, precision, recall, and F1-score are 62.61%, 62.61%, 62.61%, and 62.61%, respectively. The segmentation duration 9 in the forward direction is the best while the concatenated dataset reported only 60.47%. Figure 11 shows the accuracy, precision, recall, and F1-score curves of the different files.

5.2.4 Random forest classifier (RFC) experiment

Table 13 shows the summarization of the reported results related to the RFC experiment. It is sorted in a descending order concerning the accuracy values. It shows that the "Entropy" criterion and "50" number of estimators are the best among other variations. The "Max-Abs" scaler is reported as the best one in 4 files while the "0.005" variance threshold is reported as the best one in 6 files. The maximum reported accuracy, precision, recall, and F1-score are 100%, 100%, 100%, and 100% respectively. The segmentation durations 3N, 9N, and concatenated files are the best. Figure 12 shows the accuracy, precision, recall, and F1-score curves of the different files.

5.2.5 Extra trees classifier (ETC) experiment

Table 14 shows the summarization of the reported results related to the ETC experiment. It is sorted in a descending order concerning the accuracy values. It shows that the "Entropy" criterion and "100" number of estimators are the best among other variations. The "Max-Abs" scaler is reported as the best one in 5 files while the "0.01" variance threshold is reported as the best one in 4 files. The maximum reported accuracy, precision, recall, and F1-score are 100%, 100%, 100%, and 100% respectively. The segmentation durations 5N, 9N, and concatenated files are the

Table 16 Summarization of the reported results of all experiment concerning the concatenated dataset

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)
KNN	99.97	99.97	99.97	99.97
DT	99.89	99.89	99.89	99.89
AB	60.47	60.47	60.47	60.47
RFC	100	100	100	100
ETC	100	100	100	100

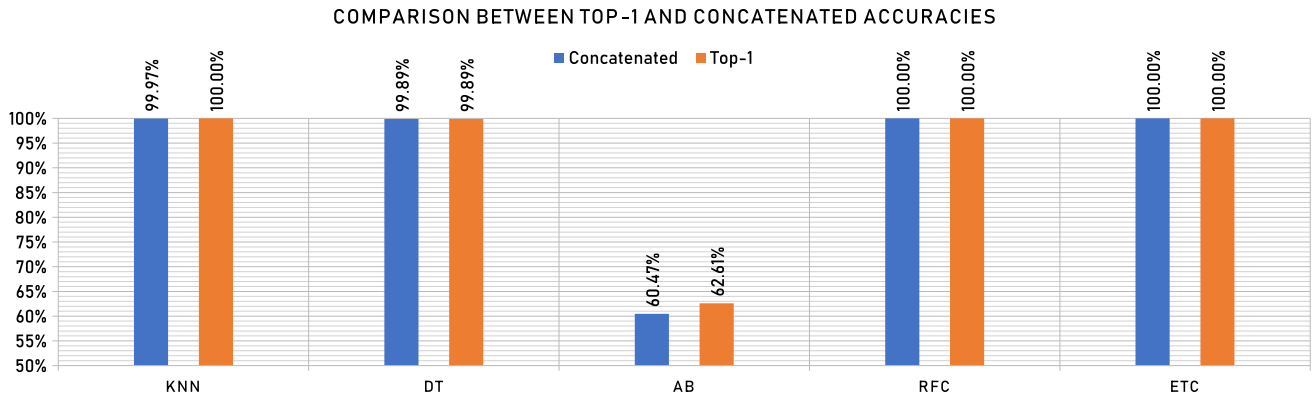

Fig. 14 Comparison between the Top-1 and concatenated accuracies

best. Figure 13 shows the accuracy, precision, recall, and F1-score curves of the different files.

5.2.6 ML experiments summarization

Table 15 shows the summarization of the best-reported results related to the ML numerical experiments concerning the top-1 accuracy. Table 16 shows the summarization of the best-reported results related to the ML numerical experiments concerning the concatenated dataset. Figure 14 compares the two tables (i.e., comparison between the Top-1 and concatenated accuracies). It shows that the concatenated dataset is better compared to other datasets. The current study recommends concatenating the segmented records in different time durations and both directions.

5.3 CNN experiments

The current subsection presents and discusses the experiments related to the images and extracted graphical features using the mentioned pre-trained CNN models (i.e., VGG16, VGG19, ResNet50, ResNet101, MobileNet, MobileNetV2, MobileNetV3Small, and MobileNetV3Large) and AO meta-heuristic optimizer. The number of epochs is set to 5. The numbers of AO iterations and population size are set to 15 and 10, respectively, and hence 150 records are reported. The captured metrics are the loss, accuracy, F1-score, recall, specificity, AUC, IOU coef., Dice coef., and precision, as mentioned in the experiments' configurations subsection [137].

5.3.1 MFCC using slaney experiment

Table 17 shows the summarization of the reported results related to the MFCC using the Slaney experiment. The table is sorted vertically in descending order concerning the accuracies. It shows that the VGG16 model reports the highest accuracy which is 99.17%. Figure 15 shows the accuracy, F1-score, recall, specificity, AUC, sensitivity, IoU, dice, and precision curves of the different pre-trained CNN models.

5.3.2 MFCC using HTK experiment

Table 18 shows the summarization of the reported results related to the MFCC using the HTK experiment. The table is sorted vertically in descending order concerning the accuracies. It shows that the ResNet50 model reports the highest accuracy which is 98.25%. Figure 16 shows the accuracy, F1-score, recall, specificity, AUC, sensitivity, IoU, dice, and precision curves of the different pre-trained CNN models.

5.3.3 STFT experiment

Table 19 shows the summarization of the reported results related to the STFT experiment. The table is sorted

Table 17 Summarization of the reported results of the MFCC using slaney experiment

Model Name	VGG16	ResNet50	MobileNet	ResNet101	MobileNetV3Large	MobileNetV3Small	VGG19	MobileNetV2
Loss	KLDivergence	Poisson	KLDivergence	Poisson	Poisson	Squared Hinge	Poisson	Categorical Hinge
Batch size	40	40	24	28	36	28	24	16
Dropout	0.56	0.15	0.36	0.31	0.46	0.06	0.37	0.25
TF learn ratio	74	81	74	53	52	22	100	8
Optimizer	SGD	SGD Nesterov	AdaGrad	SGD	SGD Nesterov	SGD	SGD Nesterov	SGD
Apply augmentation	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE	FALSE
Scaling technique	Standard	Min Max	Normalize	Min Max	Min Max	Min Max	Standard	Min Max
Rotation range	15	N/A	N/A	N/A	N/A	44	N/A	N/A
Width shift range	0	N/A	N/A	N/A	N/A	0.12	N/A	N/A
Height shift range	0.04	N/A	N/A	N/A	N/A	0.08	N/A	N/A
Shear range	0.04	N/A	N/A	N/A	N/A	0.04	N/A	N/A
Zoom range	0.11	N/A	N/A	N/A	N/A	0.04	N/A	N/A
Horizontal flip	TRUE	N/A	N/A	N/A	N/A	TRUE	N/A	N/A
Vertical flip	TRUE	N/A	N/A	N/A	N/A	FALSE	N/A	N/A
Brightness range	1.0-1.32	N/A	N/A	N/A	N/A	0.59-1.15	N/A	N/A
Loss	0.04	0.22	0.11	0.23	0.25	0.90	0.27	0.33
Accuracy	99.17%	98.39%	97.41%	96.15%	92.35%	88.04%	86.87%	84.31%
F1	99.13%	98.42%	97.28%	96.14%	91.85%	88.14%	86.66%	84.36%
Recall	99.11%	98.39%	97.19%	96.12%	90.57%	87.90%	85.21%	84.20%
Specificity	99.79%	99.61%	99.37%	99.04%	98.40%	97.13%	97.23%	96.17%
AUC	99.86%	99.63%	99.49%	99.52%	99.23%	93.74%	98.56%	93.22%
Sensitivity	99.11%	98.39%	97.19%	96.12%	90.57%	87.90%	85.21%	84.20%
IoU	99.03%	98.26%	97.49%	95.86%	88.23%	90.85%	83.70%	88.37%
Dice	99.17%	98.49%	97.81%	96.47%	90.40%	91.27%	86.47%	88.88%
Precision	99.17%	98.44%	97.45%	96.15%	93.39%	88.46%	88.51%	84.62%
TP	3568	3542	3522	3472	3293	3175	3088	3058
TN	14,370	14,344	14,404	14,309	14,311	14,034	14,095	13,972
FP	30	56	92	139	233	414	401	556
FN	32	58	102	140	343	437	536	574

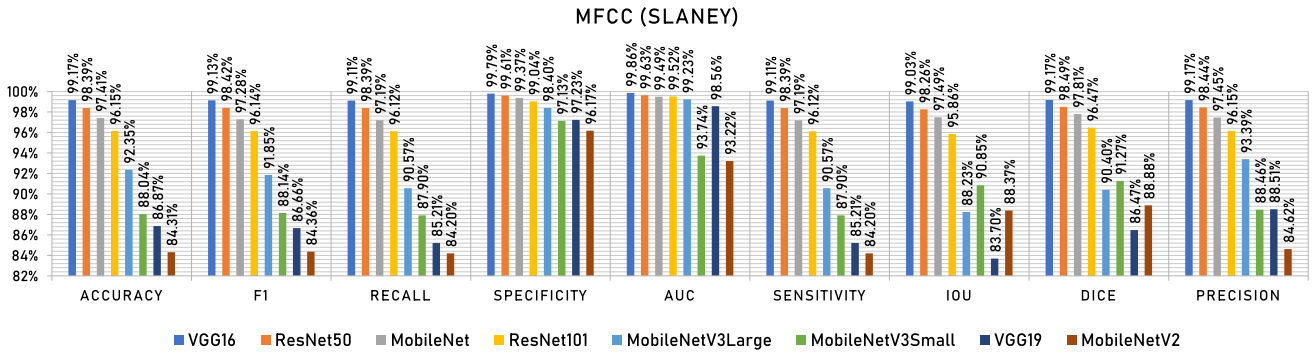

Fig. 15 The MFCC using slaney curves of the different pre-trained CNN models

Table 18 Summarization of the reported results of the MFCC using HTK experiment

Model name	ResNet50	ResNet101	MobileNet	VGG19	MobileNetV3Small	VGG16	MobileNetV2	MobileNetV3Large
Loss	KLDivergence	Poisson	Categorical Crossentropy	Poisson	Poisson	Poisson	Poisson	Categorical Crossentropy
Batch size	40	44	44	28	28	28	16	4
Dropout	0.21	0.6	0.19	0.34	0.32	0.37	0.24	0.53
TF learn ratio	55	66	88	53	54	60	69	20
Optimizer	AdaMax	SGD	SGD	SGD	RMSProp	SGD	SGD	RMSProp Centered
Apply augmentation	TRUE	FALSE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
Scaling technique	Min Max	Max Abs	Max Abs	Min Max	Min Max	Min Max	Min Max	Min Max
Rotation range	27	N/A	27	N/A	N/A	N/A	N/A	N/A
Width shift range	0.14	N/A	0.21	N/A	N/A	N/A	N/A	N/A
Height shift range	0.11	N/A	0.2	N/A	N/A	N/A	N/A	N/A
Shear range	0	N/A	0.06	N/A	N/A	N/A	N/A	N/A
Zoom range	0.1	N/A	0.05	N/A	N/A	N/A	N/A	N/A
Horizontal flip	TRUE	N/A	FALSE	N/A	N/A	N/A	N/A	N/A
Vertical flip	TRUE	N/A	TRUE	N/A	N/A	N/A	N/A	N/A
Brightness range	1.17-1.48	N/A	0.95-1.07	N/A	N/A	N/A	N/A	N/A
Loss	0.09	0.22	0.13	0.22	0.29	0.24	0.27	1.29
Accuracy	98.25%	98.06%	97.53%	96.73%	94.16%	92.28%	90.58%	83.94%
F1	98.27%	98.00%	97.53%	96.70%	94.13%	92.36%	90.50%	83.83%
Recall	98.30%	98.05%	97.53%	96.73%	94.15%	92.72%	90.70%	84.10%
Specificity	98.25%	97.95%	97.53%	96.68%	94.10%	92.03%	90.34%	83.75%
AUC	99.58%	99.51%	99.38%	99.18%	98.54%	98.21%	97.71%	96.03%
Sensitivity	99.67%	99.48%	99.50%	99.69%	97.96%	99.49%	98.33%	94.03%
IoU	98.25%	97.95%	97.53%	96.68%	94.10%	92.03%	90.34%	83.75%
Dice	98.38%	97.86%	97.59%	96.54%	95.62%	92.28%	90.61%	87.94%
Precision	98.57%	98.15%	97.88%	97.05%	95.83%	93.46%	91.80%	88.59%
TP	14,339	14,362	14,343	14,330	14,237	14,189	14,195	13,967
TN	61	70	89	118	211	259	333	577
FP	63	74	89	120	213	288	351	591
FN	1	1	1	1	1	1	1	1

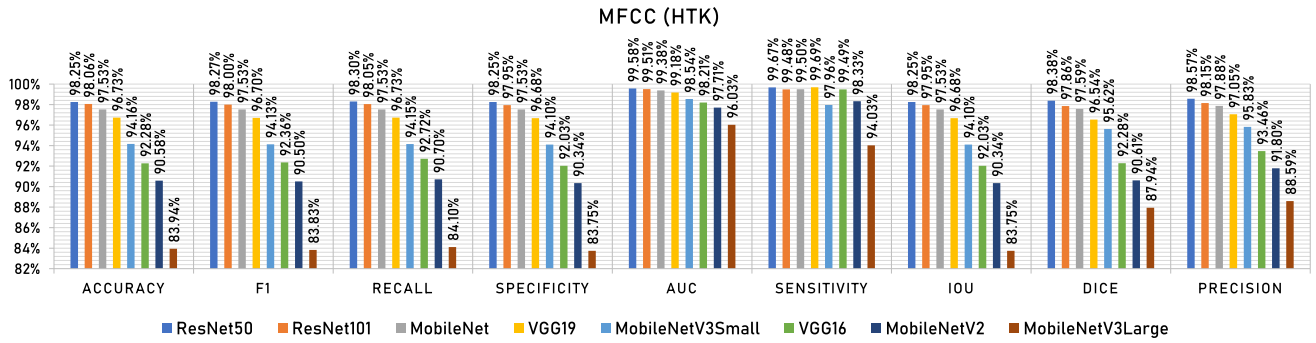

Fig. 16 The MFCC using HTK curves of the different pre-trained CNN models

vertically in descending order concerning the accuracies. It shows that the VGG19 model reports the highest accuracy which is 98.78%. Figure 17 shows the accuracy, F1-score, recall, specificity, AUC, sensitivity, IoU, dice, and precision curves of the different pre-trained CNN models.

5.3.4 Mel-specgram experiment

Table 20 shows the summarization of the reported results related to the Mel-Specgram experiment. The table is sorted vertically in descending order concerning the accuracies. It shows that the ResNet50 model reports the highest accuracy which is 98.68%. Figure 18 shows the accuracy, F1-score, recall, specificity, AUC, sensitivity, IoU, dice, and precision curves of the different pre-trained CNN models.

5.3.5 Specgram experiment

Table 21 shows the summarization of the reported results related to the Specgram experiment. The table is sorted vertically in descending order concerning the accuracies. It shows that the ResNet50 model reports the highest accuracy which is 99.00%. Figure 19 shows the accuracy, F1-score, recall, specificity, AUC, sensitivity, IoU, dice, and precision curves of the different pre-trained CNN models.

5.3.6 CNN experiments summarization

Table 22 shows the summarization of the best-reported results related to the performed CNN experiments. It shows that the best reported overall accuracy from the applied CNN experiments is 99.17% by VGG16 in the MFCC using Slaney experiment. The average accuracy is 98.78%. Applying augmentation and "Poisson" loss function are recommended by 3 experiments.

5.4 Error analysis

The authors investigated the reasons behind the mis-classification rates in the reported results and they can be: (1) the size of the dataset is not large enough, (2) the dataset is imbalanced as shown in Table 4, (3) there is a similarity percent between multiple rows after applying the segmentation, and (4) the complexity of some models are not enough for the generalization.

5.5 Related studies comparisons

Table 23 shows a comparison between the suggested approach and related studies concerning the same used datasets.

Table 19 Summarization of the reported results of the STFT experiment

Model Name	VGG19	ResNet50	VGG16	ResNet101	MobileNet	MobileNetV3Small	MobileNetV2	MobileNetV3Large
Loss	Poisson	KLDivergence	Poisson	Poisson	Categorical Crossentropy	Poisson	KLDivergence	Poisson
Batch size	20	44	28	28	36	32	40	8
Dropout	0.18	0.6	0.35	0.32	0.25	0.37	0.45	0.19
TF learn ratio	81	85	60	51	31	49	52	45
Optimizer	SGD Nesterov	AdaMax	SGD	SGD	RMSProp	AdaMax	AdaGrad	RMSProp Centered
Apply augmentation	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE
Scaling technique	Min Max	Min Max	Min Max	Min Max	Max Abs	Min Max	Standard	Min Max
Rotation range	20	N/A	N/A	N/A	19	23	14	43
Width shift range	0.18	N/A	N/A	N/A	0.05	0.03	0.21	0.03
Height shift range	0.04	N/A	N/A	N/A	0.08	0.02	0.15	0.24
Shear range	0.06	N/A	N/A	N/A	0.13	0.25	0.04	0.08
Zoom range	0.19	N/A	N/A	N/A	0.06	0.25	0.2	0.18
Horizontal flip	TRUE	N/A	N/A	N/A	FALSE	FALSE	TRUE	TRUE
Vertical flip	FALSE	N/A	N/A	N/A	TRUE	FALSE	FALSE	FALSE
Brightness range	1.05-1.93	N/A	N/A	N/A	0.8-1.79	1.24-1.59	0.62-1.86	0.8-1.92
Loss	0.21	0.08	0.22	0.22	0.38	0.26	0.28	0.37
Accuracy	98.78%	98.61%	98.56%	96.82%	94.22%	94.14%	90.92%	90.09%
F1	98.80%	98.61%	98.56%	96.62%	94.26%	94.15%	90.96%	90.01%
Recall	98.81%	98.63%	98.56%	96.82%	94.46%	94.19%	91.29%	90.06%
Specificity	98.78%	98.59%	98.56%	96.43%	94.09%	94.11%	90.67%	89.98%
AUC	99.70%	99.66%	99.64%	99.21%	98.62%	98.55%	97.84%	97.52%
Sensitivity	99.79%	99.69%	99.64%	99.76%	98.44%	98.59%	98.85%	95.93%
IoU	98.78%	98.59%	98.56%	96.43%	94.09%	94.11%	90.67%	89.98%
Dice	98.58%	98.86%	98.41%	96.39%	95.11%	95.10%	91.03%	92.77%
Precision	98.79%	98.98%	98.62%	96.94%	95.55%	95.54%	92.30%	93.07%
TP	14,437	14,383	14,396	14,334	14,343	14,254	14,089	14,168
TN	43	49	52	114	201	210	311	360
FP	44	51	52	129	215	213	336	364
FN	1	1	1	1	1	1	1	1

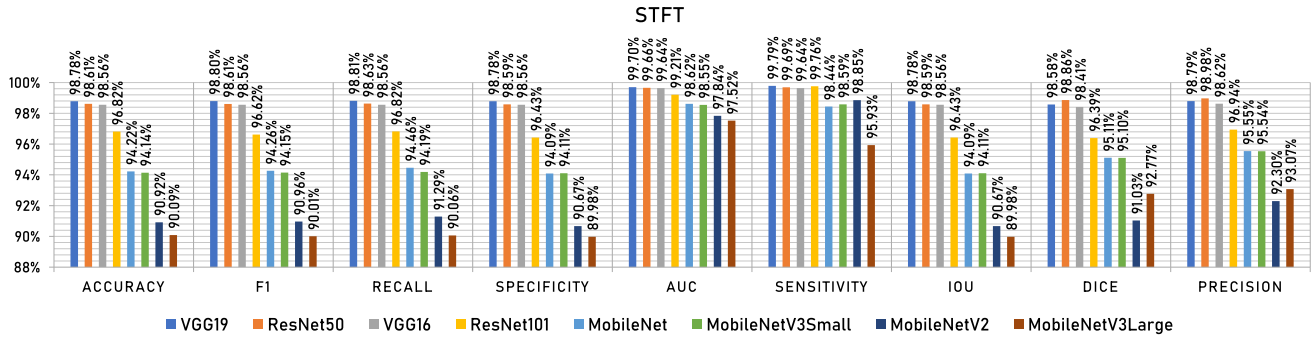

Fig. 17 The STFT curves of the different pre-trained CNN models

6 Study limitations

The results of the suggested framework are encouraging but there are still some limitations. First, the use of voice records only. Second, the selection of only eight transfer learning CNN models among the available models. Third, the study did not include the use of Long short-term memory (or other variations) for data with frequency. Fourth, the current study does not utilize the Graphical neural networks which can be used in future studies [146]. However, the results of the current study are promising and the proposed framework can be applied in hospitals.

7 Conclusions and future work

With the appliance of artificial intelligence in medical diagnosis, the detection of diseases has become more accurate. In this work, a framework for the detection of one of the widely spread diseases (i.e., Cardiovascular diseases) is proposed. The reason behind this choice is the high morbidity and mortality rate due to these diseases. The hybrid framework uses medical voice records for the detection of heart diseases. The different layers of the suggested framework are Segmentation Layer, Features Extraction Layer, Learning and Optimization Layer, and Export and Statistics Layer. The segmentation Layer is the layer in which the different records are segmented with specific durations. A novel segmentation technique using variable durations forward and backward is proposed. In the Features Extraction Layer, numerical and graphical features are extracted from the resulting datasets. These features are passed to the Learning and Optimization Layer, where numerical features are passed to 5 different Machine Learning (ML) algorithms with Grid Search optimization algorithm, while graphical features are passed to 8 different Convolutional Neural Networks (CNN) with Aquila Optimizer (AO) using transfer learning. Different performance metrics are used in the Export and Statistics Layer to validate the response of the proposed framework. The best-reported metrics are 100% accuracy, precision, recall, and F1 score using ML algorithms such as ETC and RFC. Also, the proposed approach achieved 99.17% accuracy using CNN.

Table 20 Summarization of the reported results of the mel-specgram experiment

Model Name	ResNet50	ResNet101	MobileNetV2	MobileNet	VGG19	MobileNetV3Small	MobileNetV3Large	VGG16
Loss	Poisson	Categorical Crossentropy	KLDivergence	KLDivergence	Poisson	KLDivergence	Poisson	Poisson
Batch size	36	48	40	32	28	28	16	24
Dropout	0.4	0.14	0.43	0.36	0.26	0.54	0.32	0.28
TF learn ratio	70	20	68	71	32	35	61	42
Optimizer	SGD	AdaGrad	AdaGrad	AdaGrad	AdaMax	AdaMax	AdaGrad	SGD Nesterov
Apply augmentation	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
Scaling technique	Min Max	Max Abs	Max Abs	Normalize	Standard	Min Max	Min Max	Standard
Rotation range	N/A	N/A	32	5	20	23	11	N/A
Width shift range	N/A	N/A	0.2	0.06	0.11	0.12	0.02	N/A
Height shift range	N/A	N/A	0.21	0.03	0.16	0.01	0.12	N/A
Shear range	N/A	N/A	0.17	0.23	0.08	0.07	0.04	N/A
Zoom range	N/A	N/A	0.11	0.09	0.23	0.09	0.1	N/A
Horizontal flip	N/A	N/A	FALSE	TRUE	FALSE	TRUE	FALSE	N/A
Vertical l	N/A	N/A	FALSE	FALSE	FALSE	TRUE	TRUE	N/A
Brightness range	N/A	N/A	1.48-1.48	0.57-1.34	1.28-1.89	1.05-1.19	0.54-1.21	N/A
Loss	0.22	0.08	0.16	0.14	0.24	0.18	0.24	0.26
Accuracy	98.68%	97.97%	96.28%	95.99%	94.74%	94.71%	92.35%	89.05%
F1	98.65%	97.93%	96.27%	95.99%	94.85%	94.81%	92.25%	88.66%
Recall	98.67%	97.98%	96.47%	96.10%	95.13%	95.02%	92.60%	90.09%
Specificity	98.62%	97.89%	96.08%	95.88%	94.60%	94.63%	91.93%	87.42%
AUC	99.67%	99.50%	99.12%	99.03%	98.79%	98.77%	98.17%	97.58%
Sensitivity	99.63%	99.67%	99.37%	99.47%	99.23%	99.30%	99.29%	98.84%
IoU	98.62%	97.89%	96.08%	95.88%	94.60%	94.63%	91.93%	87.42%
Dice	98.74%	97.77%	95.22%	95.51%	94.29%	95.47%	89.73%	82.19%
Precision	98.88%	98.12%	96.02%	96.19%	95.16%	95.96%	91.56%	85.52%
TP	14496	14328	14273	14324	14273	14270	14262	14145
TN	48	72	127	140	175	178	266	351
FP	50	76	141	149	195	194	293	456
FN	1	1	1	1	1	1	1	1

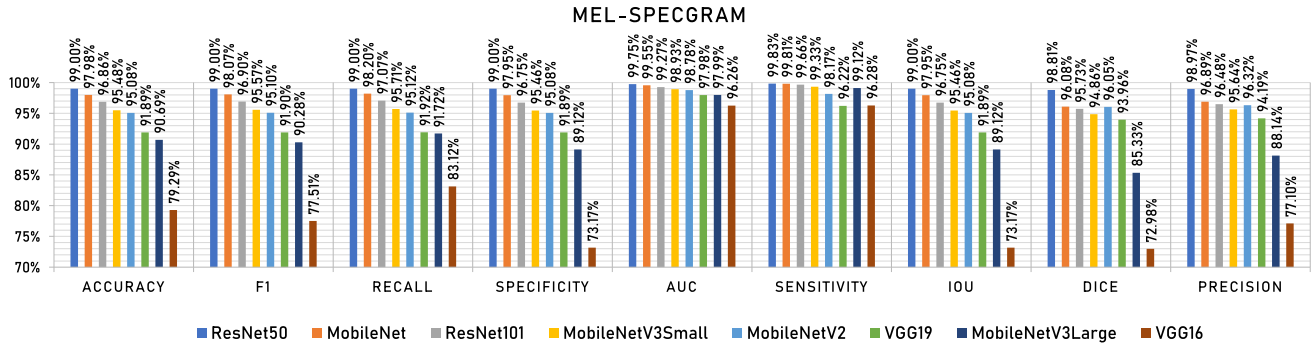

Fig. 18 The mel-specgram curves of the different pre-trained CNN models

Table 21 Summarization of the reported results of the specgram experiment

Model name	ResNet50	MobileNet	ResNet101	MobileNetV3Small	MobileNetV2	VGG19	MobileNetV3Large	VGG16
Loss	Poisson	Categorical crossentropy	Poisson	Poisson	Squared hinge	Squared hinge	Poisson	Poisson
Batch size	28	32	40	16	36	28	16	28
Dropout	0.36	0.12	0.46	0.12	0.34	0.11	0.02	0.07
TF learn ratio	67	57	81	44	54	94	40	59
Optimizer	SGD	AdaGrad	AdaGrad	Adam AMSGrad	SGD	SGD Nesterov	SGD	SGD Nesterov
Apply augmentation	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
Scaling technique	Min Max	Min Max	Standard	Min Max	Max Abs	Standard	Max Abs	Min Max
Rotation range	N/A	17	27	9	25	1	N/A	N/A
Width shift range	N/A	0.11	0.08	0.19	0.09	0.09	N/A	N/A
Height shift range	N/A	0.05	0.13	0.13	0.03	0.22	N/A	N/A
Shear range	N/A	0.02	0.05	0.23	0.09	0.13	N/A	N/A
Zoom range	N/A	0.17	0.1	0.19	0.13	0.23	N/A	N/A
Horizontal flip	N/A	TRUE	FALSE	FALSE	FALSE	FALSE	N/A	N/A
Vertical flip	N/A	FALSE	FALSE	TRUE	FALSE	FALSE	N/A	N/A
Brightness range	N/A	1.13-1.65	1.15-1.31	1.12-1.42	0.86-1.7	0.62-1.31	N/A	N/A
Loss	0.21	0.08	0.22	0.24	0.84	0.87	0.25	0.31
Accuracy	99.00%	97.98%	96.86%	95.48%	95.08%	91.89%	90.69%	79.29%
F1	99.00%	98.07%	96.90%	95.57%	95.10%	91.90%	90.28%	77.51%
Recall	99.00%	98.20%	97.07%	95.71%	95.12%	91.92%	91.72%	83.12%
Specificity	99.00%	97.95%	96.75%	95.46%	95.08%	91.89%	89.12%	73.17%
AUC	99.75%	99.55%	99.27%	98.93%	98.78%	97.98%	97.99%	96.26%
Sensitivity	99.83%	99.81%	99.66%	99.33%	98.17%	96.22%	99.12%	96.28%
IoU	99.00%	97.95%	96.75%	95.46%	95.08%	91.89%	89.12%	73.17%
Dice	98.81%	96.08%	95.73%	94.86%	96.05%	93.96%	85.33%	72.98%
Precision	98.97%	96.89%	96.48%	95.64%	96.32%	94.19%	88.14%	77.10%
TP	14412	14399	14295	14373	14367	14156	14236	13907
TN	36	65	105	155	177	292	292	541
FP	36	74	117	165	179	293	395	969
FN	1	1	1	1	1	1	1	1

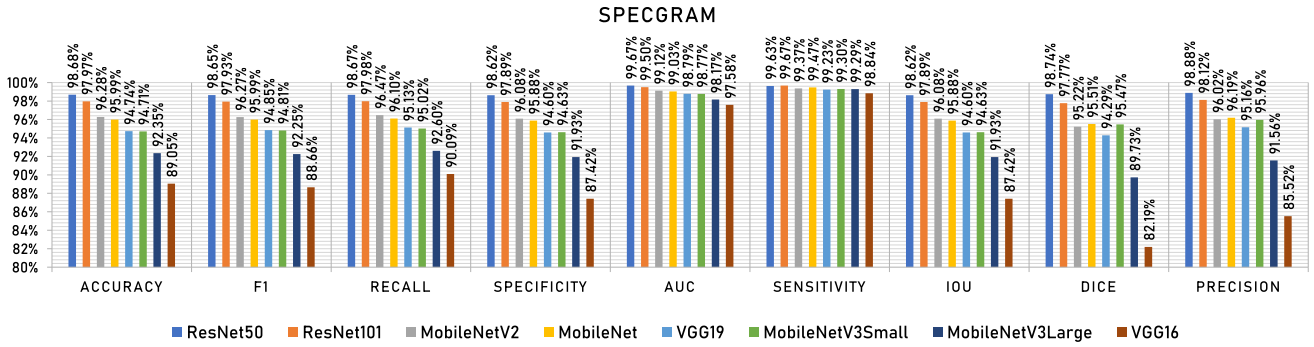

Fig. 19 The specgram curves of the different pre-trained CNN models

Table 22 Summarization of the reported results of all experiments

Features	MFCC (Slaney)	MFCC (HTK)	STFT	Specgram	Mel-Specgram	Best	Average
Model name	VGG16	ResNet50	VGG19	ResNet50	ResNet50		
Loss	KLDivergence	KLDivergence	Poisson	Poisson	Poisson		
Batch size	40	40	20	36	28		
Dropout	0.56	0.21	0.18	0.4	0.36		
TF learn ratio	74	55	81	70	67		
Optimizer	SGD	AdaMax	SGD Nesterov	SGD	SGD		
Apply augmentation	TRUE	TRUE	TRUE	FALSE	FALSE		
Scaling technique	Standard	Min Max	Min Max	Min Max	Min Max		
Rotation range	15	27	20	N/A	N/A		
Width shift range	0	0.14	0.18	N/A	N/A		
Height shift range	0.04	0.11	0.04	N/A	N/A		
Shear range	0.04	0	0.06	N/A	N/A		
Zoom range	0.11	0.1	0.19	N/A	N/A		
Horizontal flip	TRUE	TRUE	TRUE	N/A	N/A		
Vertical flip	TRUE	TRUE	FALSE	N/A	N/A		
Brightness range	1.0-1.32	1.17-1.48	1.05-1.93	N/A	N/A		
Loss	0.04	0.09	0.21	0.22	0.21	0.04	0.15
Accuracy	99.17%	98.25%	98.78%	98.68%	99.00%	99.17%	98.78%
F1	99.13%	98.27%	98.80%	98.65%	99.00%	99.13%	98.77%
Recall	99.11%	98.30%	98.81%	98.67%	99.00%	99.11%	98.78%
Specificity	99.79%	98.25%	98.78%	98.62%	99.00%	99.79%	98.89%
AUC	99.86%	99.58%	99.70%	99.67%	99.75%	99.86%	99.71%
Sensitivity	99.11%	99.67%	99.79%	99.63%	99.83%	99.83%	99.61%
IoU	99.03%	98.25%	98.78%	98.62%	99.00%	99.03%	98.74%
Dice	99.17%	98.38%	98.58%	98.74%	98.81%	99.17%	98.74%
Precision	99.17%	98.57%	98.79%	98.88%	98.97%	99.17%	98.87%
TP	3568	14339	14437	14496	14412	14496	12250.4
TN	14370	61	43	48	36	14370	2911.6
FP	30	63	44	50	36	30	44.6
FN	32	1	1	1	1	1	7.2

Table 23 Comparison between the suggested approach and related studies

Study	Year	Dataset	Approach	Best Performance
Narváez et al. [138]	2020	Classifying heart sounds challenge dataset [135]	ML Algorithms	99.26% using KNN
Raza et al. [40]	2019	Classifying heart sounds challenge dataset [135]	DL and ML Approaches	80.8% using RNN
Nogueira et al. [139]	2019	Classifying heart sounds challenge dataset [135]	DL and ML Approaches	83.22% using SVM
Akram et al. [140]	2018	Classifying heart sounds challenge dataset [135]	ML and Localization Algorithms	90.62% using SVM
DEPERLİĞLU et al. [141]	2018	Classifying heart sounds challenge dataset [135]	ANN	88.6% using ANN
Banerjee et al. [142]	2020	Classifying heart sounds challenge dataset [135]	DL Approach	83% using CNN
Deperlioglu et al. [143]	2018	Classifying heart sounds challenge dataset [135]	ANN and CNN	97.90% using CNN
Bilal et al. [144]	2021	Classifying heart sounds challenge dataset [135] and PhysioNet 2016 [145]	CNN with 1D-local binary pattern and 1D-local ternary pattern features	91.66% for the first dataset and 91.78% for the second dataset using CNN
Current Study	2021	Classifying heart sounds challenge dataset [135]	Hybrid (ML and CNN)	100% (ML) and 99.17% (CNN)

7.1 Future work

In future work, the authors will apply the suggested approach on different datasets types such as waves. Also, the datasets can be handled and utilized using different optimization methods. Finally, graphical neural networks and LSTM networks can be utilized.

Appendix

Table of abbreviations

Table 24 shows the abbreviations and the corresponding meaning. They are sorted alphabetically in ascending order.

Table 24 Table of Abbreviations

Abbreviation	Definition
AB	AdaBoost
Adam	An algorithm and not an acronym
ANN	Artificial Neural Network
AO	Aquila Optimizer
AUC	Area Under Curve
CENS	Chroma Energy Normalized Statistics
CNN	Convolutional Neural Network
CQT	Constant-Q chromogram Transform
CVDs	Cardiovascular Diseases
DA	Data Augmentation
DN	Data Normalization
DNN	Deep Neural Network
DT	Decision Tree
ECG	Electrocardiogram
ETC	Extra Tree Classifier
GAN	Generative Adversarial Networks
GB	Gradient Boosting
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IoU	Intersection over Union
KNN	K-Nearest Neighbor
LR	Logistic Regression
LSTM	Long Short-Term Memory
MFCC	Mel Frequency Cepstral Coefficients
ML	Machine Learning
MLP	Multilayer Perceptron
NB	Naïve Bayes
ReLU	Rectified Linear Units
RFC	Random Forest Classifier
RMSE	Root Mean Square Energy
RNN	Recurrent Neural Network
STFT	Short-Time Fourier Transform
SVM	Support Vector Machine
WHO	World Health Organization
ZCR	Zero Crossing Rate

Author contributions All the authors have participated in writing the manuscript and have revised the final version. All authors read and approved the final manuscript.

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB). No funding was received for this work (i.e., study).

Data availability The datasets, if existing, that are used, generated, or analyzed during the current study (A) if the datasets are owned by the authors, they are available from the corresponding author on reasonable request, (B) if the datasets are not owned by the authors, the supplementary information including the links and sizes are included in this published article.

Detailed Declarations

Conflict of interest No conflict of interest exists. We wish to confirm that, there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Ethical approval We further confirm, if existing, that any aspect of the work covered in this manuscript that has involved human patients has been conducted with the ethical approval of all relevant bodies and that such approvals are acknowledged within the manuscript. Written consent to publish potentially identifying information, such as details of the case and photographs, was obtained from the patient(s) or their legal guardian(s).

Human or animal rights The current study does not contain any studies with human participants and/or animals performed by any of the authors.

Consent to participate There is no informed consent for the current study.

Consent for publication Not Applicable.

References

1. Li M et al (2021) Piwi-interacting rnas (pirnas) as potential biomarkers and therapeutic targets for cardiovascular diseases. Angiogenesis 24(1):19–34
2. Bui AL, Horwich TB, Fonarow GC (2011) Epidemiology and risk profile of heart failure. Nat Rev Cardiol 8(1):30–41
3. World health organization, cardiovascular diseases (2015) (https://www.who.int/en/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)). Accessed 12 August 2021
4. Ulbricht T, Southgate D (1991) Coronary heart disease: seven dietary factors. Lancet 338(8773):985–992
5. Hassanin A, Hassanein M, Bendary A, Maksoud MA (2020) Demographics, clinical characteristics, and outcomes among hospitalized heart failure patients across different regions of egypt. Egypt Heart J 72(1):1–9
6. Allen LA et al (2012) Decision making in advanced heart failure: a scientific statement from the american heart association. Circulation 125(15):1928–1952
7. Yusuf S, Reddy S, Ounpuu S, Anand S (2001) Global burden of cardiovascular diseases: Part ii: variations in cardiovascular disease by specific ethnic groups and geographic regions and prevention strategies. Circulation 104(23):2855–2864
8. Packer M et al (2020) Cardiovascular and renal outcomes with empagliflozin in heart failure. N Engl J Med 383(15):1413–1424

9. Das R, Turkoglu I, Sengur A (2009) Effective diagnosis of heart disease through neural networks ensembles. Expert Syst Appl 36(4):7675–7680
10. Arabasadi Z, Alizadehsani R, Roshanzamir M, Moosaei H, Yarifard AA (2017) Computer aided decision making for heart disease detection using hybrid neural network-genetic algorithm. Comput Methods Programs Biomed 141:19–26
11. Balaha HM, Ali HA, Saraya M, Badawy M (2021) A new arabic handwritten character recognition deep learning system (ahcr-dls). Neural Comput Appl 33(11):6325–6367
12. Chen AH, Huang SY, Hong PS, Cheng CH, Lin EJ (2011) Hdps: heart disease prediction system, In: 2011 computing in cardiology. IEEE, pp 557–560
13. Mehmood A et al (2021) Prediction of heart disease using deep convolutional neural networks. Arab J Sci Eng 46(4):3409–3422
14. Demir F, Şengür A, Bajaj V, Polat K (2019) Towards the classification of heart sounds based on convolutional deep neural network. Health Inf Sci Syst 7(1):1–9
15. Kim KH, Choi HJ (2007) Design of a clinical knowledge base for heart disease detection. In: 7th IEEE international conference on computer and information technology (CIT 2007). IEEE. pp 610–615
16. Heinrichs B, Eickhoff SB (2020) Your evidence? machine learning algorithms for medical diagnosis and prediction. Hum Brain Mapp 41(6):1435–1444
17. Raikwal J, Saxena K (2012) Performance evaluation of svm and k-nearest neighbor algorithm over medical data set. Int J Comput Appl 50(14)
18. Mrva J, Neupauer Š, Hudec L, Ševcech J, Kapec P (2019) Decision support in medical data using 3d decision tree visualisation. In: 2019 E-health and bioengineering conference (EHB). IEEE, pp 1–4
19. Ali F et al (2020) A smart healthcare monitoring system for heart disease prediction based on ensemble deep learning and feature fusion. Inf Fusion 63:208–222
20. Jang HJ, Cho KO (2019) Applications of deep learning for the analysis of medical data. Arch Pharmacal Res 42(6):492–504
21. Kayalibay B, Jensen G, van der Smagt P (2017) Cnn-based segmentation of medical imaging data. arXiv preprint arXiv:1701.03056
22. Brunese L, Martinelli F, Mercaldo F, Santone A (2020) Deep learning for heart disease detection through cardiac sounds. Procedia Comput Sci 176:2202–2211
23. Miao JH, Miao KH (2018) Cardiotocographic diagnosis of fetal health based on multiclass morphologic pattern predictions using deep learning classification. Int J Adv Comput Sci Appl 9(5):1–11
24. Abdel-Alim O, Hamdy N, El-Hanjouri M (2002) Heart diseases diagnosis using heart sounds. In: Proceedings of the nineteenth national radio science conference. IEEE, pp 634–640
25. Zhang W, Yu L, Ye L, Zhuang W, Ma F (2018) Ecg signal classification with deep learning for heart disease identification. In: 2018 international conference on big data and artificial intelligence (BDAI). IEEE, pp 47–51
26. Zhang J, Li B, Xiang K, Shi X (2019) Method of diagnosing heart disease based on deep learning ecg signal. arXiv preprint arXiv:1907.01514
27. Jm Kwon, Kim KH, Jeon KH, Park J (2019) Deep learning for predicting in-hospital mortality among heart disease patients based on echocardiography. Echocardiography 36(2):213–218
28. Sajeev S, et al. (2019) Deep learning to improve heart disease risk prediction. In: Machine learning and medical engineering for cardiovascular health and intravascular imaging and computer assisted stenting. Springer, pp 96–103
29. Rath A, Mishra D, Panda G, Satapathy SC (2021) Heart disease detection using deep learning methods from imbalanced ecg samples. Biomed Signal Process Control 68:102820
30. Darmawahyuni A, Nurmaini S, Firdaus F (2019) Coronary heart disease interpretation based on deep neural network. Comput Eng Appl J 8(1):1–12
31. Jindal H, Agrawal S, Khera R, Jain R, Nagrath P (2021) Heart disease prediction using machine learning algorithms. In: IOP conference series: materials science and engineering, vol 1022. p 012072
32. Muhammad Y, Tahir M, Hayat M, Chong KT (2020) Early and accurate detection and diagnosis of heart disease using intelligent computational model. Sci Rep 10(1):1–17
33. Pugazhenthi D, Meenakshi V (2016) Detection of ischemic heart diseases from medical images. In: 2016 international conference on micro-electronics and telecommunication engineering (ICMETE). IEEE, pp 355–360
34. Alarsan FI, Younes M (2019) Analysis and classification of heart diseases using heartbeat features and machine learning algorithms. J Big Data 6(1):1–15
35. Nikhar S, Karandikar A (2016) Prediction of heart disease using machine learning algorithms. Int J Adv Eng Manag Sci 2(6):239484
36. Patel J, TejalUpadhyay D, Patel S (2015) Heart disease prediction using machine learning and data mining technique. Heart Dis 7(1):129–137
37. Singh A, Kumar R (2020) Heart disease prediction using machine learning algorithms. In: 2020 international conference on electrical and electronics engineering (ICE3). IEEE, pp 452–457
38. Krishnan S, Geetha S (2019) Prediction of heart disease using machine learning algorithms. In: 2019 1st international conference on innovations in information and communication technology (ICIICT). IEEE, pp 1–5
39. Pasha SN, Ramesh D, Mohmmad S, Harshavardhan A, et al. (2020) Cardiovascular disease prediction using deep learning techniques. In: IOP conference series: materials science and engineering, vol. 981. IOP Publishing, p 022006
40. Raza A et al (2019) Heartbeat sound signal classification using deep learning. Sensors 19(21):4819
41. Sajja TK, Kalluri HK (2020) A deep learning method for prediction of cardiovascular disease using convolutional neural network. Rev d'Intell Artif 34(5):601–606
42. Haq AU, Li JP, Memon MH, Nazir S, Sun R (2018) A hybrid intelligent system framework for the prediction of heart disease using machine learning algorithms. Mob Inf Syst
43. Gavhane A, Kokkula G, Pandya I, Devadkar K (2018) Prediction of heart disease using machine learning. In: 2018 second international conference on electronics, communication and aerospace technology (ICECA). IEEE, pp 1275–1278
44. Sharma S, Parmar M (2020) Heart diseases prediction using deep learning neural network model. Int J Innov Technol Explor Eng (IJITEE) 9(3)
45. Hundal JK, Hamde S (2017) Some feature extraction techniques for voice based authentication system. In: 2017 IEEE international conference on power, control, signals and instrumentation engineering (ICPCSI). IEEE, pp 419–421
46. Kurzekar PK, Deshmukh RR, Waghmare VB, Shrishrimal PP (2014) A comparative study of feature extraction techniques for speech recognition system. Int J Innov Res Sci Eng Technol 3(12):18006–18016
47. Li F et al (2019) Feature extraction and classification of heart sound using 1d convolutional neural networks. EURASIP J Adv Signal Process 1:1–11
48. Muda L, Begam M, Elamvazuthi I (2010) Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and

dynamic time warping (dtw) techniques. arXiv preprint arXiv:1003.4083
49. Nair AP, Krishnan S, Saquib Z (2016) Mfcc based noise reduction in asr using kalman filtering. In: 2016 conference on advances in signal processing (CASP). IEEE, pp 474–478
50. Kavita D, Saxena A, Joshi J (2016) A review: speech reorganization by using artificial neural network
51. Ganchev T, Fakotakis N, Kokkinakis G (2005) Comparative evaluation of various mfcc implementations on the speaker verification task. In: Proceedings of the SPECOM, vol 1, pp 191–194
52. Park DS, et al. (2019) Specaugment: a simple data augmentation method for automatic speech recognition. arXiv preprint arXiv:1904.08779
53. Oo MM, Oo LL (2019) Fusion of log-mel spectrogram and glcm feature in acoustic scene classification. In: International conference on software engineering research, management and applications. Springer, pp 175–187
54. Bachu R, Kopparthi S, Adapa B, Barkana B (2008) Separation of voiced and unvoiced using zero crossing rate and energy of the speech signal. In: American society for engineering education (ASEE) zone conference proceedings. pp 1–7
55. Huang J, Chen B, Yao B, He W (2019) Ecg arrhythmia classification using stft-based spectrogram and convolutional neural network. IEEE Access 7:92871–92880
56. Sharma J, Granmo OC, Goodwin M (2020) Environment sound classification using multiple feature channels and attention based deep convolutional neural network. In INTERSPEECH. pp 1186–1190
57. Muller M, Kurth F, Clausen M (2005) Chroma-based statistical audio features for audio matching. In: IEEE workshop on applications of signal processing to audio and acoustics, 2005. IEEE, pp 275–278
58. Norouzi M, Akbarizadeh G, Eftekhar F (2018) A hybrid feature extraction method for sar image registration. SIViP 12(8):1559–1566
59. Humphrey EJ, Cho T, Bello JP (2012) Learning a robust tonnetz-space transform for automatic chord recognition. In: 2012 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE, pp 453–456
60. Tzanetakis G, Cook P (2002) Musical genre classification of audio signals. IEEE Trans Speech Audio Process 10(5):293–302
61. Neukam C, Nagel F, Schuller G, Schnabel M (2013) A mdct based harmonic spectral bandwidth extension method. In: 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, pp 566–570
62. Jiang DN, Lu L, Zhang HJ, Tao JH, Cai LH (2002) Music type classification by spectral contrast feature. In: Proceedings. IEEE international conference on multimedia and expo, vol 1. IEEE, pp 113–116
63. Ma Y, Nishihara A (2013) Efficient voice activity detection algorithm using long-term spectral flatness measure. EURASIP J Audio Speech Music Process 1:1–18
64. Stolar MN, Lech M, Stolar SJ, Allen NB (2018) Detection of adolescent depression from speech using optimised spectral roll-off parameters. Biomed J 2:10
65. Bonaccorso G (2017) Machine learning algorithms. Packt Publishing Ltd
66. Gök M (2015) An ensemble of k-nearest neighbours algorithm for detection of parkinson's disease. Int J Syst Sci 46(6):1108–1112
67. Kozma L (2008) k nearest neighbors algorithm (knn). Helsinki University of Technology
68. Triguero I, Maillo J, Luengo J, García S, Herrera F (2016) From big data to smart data with the k-nearest neighbours algorithm. In: 2016 IEEE international conference on internet of things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData). IEEE, pp 859–864
69. Sutton O (2012) Introduction to k nearest neighbour classification and condensed nearest neighbour data reduction. University lectures, University of Leicester, p 1
70. Nikdel H, Forghani Y, Mohammad Hosein Moattar S (2018) Increasing the speed of fuzzy k-nearest neighbours algorithm. Expert Syst 35(3):e12254
71. Su J, Zhang H (2006) A fast decision tree learning algorithm. In: Aaai, vol 6. pp 500–505
72. Charbuty B, Abdulazeez A (2021) Classification based on decision tree algorithm for machine learning. J Appl Sci Technol Trends 2(01):20–28
73. Priyam A, Abhijeeta G, Rathee A, Srivastava S (2013) Comparative analysis of decision tree classification algorithms. Int J Curr Eng Technol 3(2):334–337
74. Lin W, Wu Z, Lin L, Wen A, Li J (2017) An ensemble random forest algorithm for insurance big data analysis. IEEE Access 5:16568–16575
75. Biau G, Scornet E (2016) A random forest guided tour. TEST 25(2):197–227
76. Geurts P, Ernst D, Wehenkel L (2006) Extremely randomized trees. Mach Learn 63(1):3–42
77. Bhati BS, Rai C (2020) Ensemble based approach for intrusion detection using extra tree classifier. Intell Comput Eng 213–220
78. Schapire RE (2013) Explaining adaboost. In: Empirical inference. Springer, New York, pp 37–52
79. Ying C, Qi-Guang M, Jia-Chen L, Lin G (2013) Advance and prospects of adaboost algorithm. Acta Autom Sin 39(6):745–758
80. Balaha HM, El-Gendy EM, Saafan MM (2021) Covh2sd: a covid-19 detection approach based on harris hawks optimization and stacked deep learning. Expert Syst Appl 186:115805
81. Albawi S, Mohammed TA, Al-Zawi S (2017) Understanding of a convolutional neural network. In: 2017 international conference on engineering and technology (ICET). IEEE, pp 1–6
82. Acharya UR et al (2017) A deep convolutional neural network model to classify heartbeats. Comput Biol Med 89:389–396
83. Sun M, Song Z, Jiang X, Pan J, Pang Y (2017) Learning pooling for convolutional neural network. Neurocomputing 224:96–104
84. Balaha HM, Balaha MH, Ali HA (2021) Hybrid covid-19 segmentation and recognition framework (hmb-hcf) using deep learning and genetic algorithms. Artif Intell Med 119:102156
85. Chauhan R, Ghanshala KK, Joshi R (2018) Convolutional neural network (cnn) for image detection and recognition. In: 2018 first international conference on secure cyber computing and communication (ICSCCC). IEEE, pp 278–282
86. Balaha HM, Saif M, Tamer A, Abdelhay EH (2022) Hybrid deep learning and genetic algorithms approach (hmb-dlgaha) for the early ultrasound diagnoses of breast cancer. Neural Comput Appl 1–25
87. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. PMLR, pp 448–456
88. Agostinelli F, Hoffman M, Sadowski P, Baldi P (2014) Learning activation functions to improve deep neural networks. arXiv preprint arXiv:1412.6830
89. Balaha HM, El-Gendy EM, Saafan MM (2022) A complete framework for accurate recognition and prognosis of covid-19 patients based on deep transfer learning and feature classification approach. Artif Intell Rev 1–46
90. Bahgat WM, Balaha HM, AbdulAzeem Y, Badawy MM (2021) An optimized transfer learning-based approach for automatic diagnosis of covid-19 from chest x-ray images. PeerJ Comput Sci 7:e555

91. Balaha HM, Ali HA, Badawy M (2021) Automatic recognition of handwritten arabic characters: a comprehensive review. Neural Comput Appl 33(7):3011–3034
92. Torrey L, Shavlik J (2010) Transfer learning. In: Handbook of research on machine learning applications and trends: algorithms, methods, and techniques. IGI global, pp 242–264
93. Deng J, et al. (2009) Imagenet: a large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. IEEE, pp 248–255
94. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
95. Abdulazeem Y, Balaha HM, Bahgat WM, Badawy M (2021) Human action recognition based on transfer learning approach. IEEE Access 9:82058–82069
96. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 770–778
97. Allen-Zhu Z, Li Y (2019) What can resnet learn efficiently, going beyond kernels? arXiv preprint arXiv:1905.10337
98. Howard AG, et al. (2017) Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861
99. Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC (2018) Mobilenetv2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 4510–4520
100. Balaha HM et al (2021) Recognizing arabic handwritten characters using deep learning and genetic algorithms. Multimed Tools Appl 80(21):32473–32509
101. Russakovsky O et al (2015) Imagenet large scale visual recognition challenge. Int J Comput Vision 115(3):211–252
102. Howard A, et al. (2019) Searching for mobilenetv3. In: Proceedings of the IEEE/CVF international conference on computer vision. pp 1314–1324
103. Vani S, Rao TM (2019) An experimental approach towards the performance assessment of various optimizers on convolutional neural network. In: 2019 3rd international conference on trends in electronics and informatics (ICOEI). IEEE, pp 331–336
104. Zhang Z (2018) Improved adam optimizer for deep neural networks. In: 2018 IEEE/ACM 26th international symposium on quality of service (IWQoS). IEEE, pp 1–2
105. Halgamuge MN, Daminda E, Nirmalathas A (2020) Best optimizer selection for predicting bushfire occurrences using deep learning. Nat Hazards 103:845–860
106. Lydia A, Francis S (2019) Adagrad-an optimizer for stochastic gradient descent. Int J Inf Comput Sci 6(5)
107. Wang Y et al (2019) Assessing optimizer impact on dnn model sensitivity to adversarial examples. IEEE Access 7:152766–152776
108. Bera S, Shrivastava VK (2020) Analysis of various optimizers on deep convolutional neural network model in the application of hyperspectral remote sensing image classification. Int J Remote Sens 41(7):2664–2683
109. Kumar A, Sarkar S, Pradhan C (2020) Malaria disease detection using cnn technique with sgd, rmsprop and adam optimizers. Deep learning techniques for biomedical and health informatics. Springer, New York, pp 211–230
110. Duda J (2019) Sgd momentum optimizer with step estimation by online parabola model. arXiv preprint arXiv:1907.07063
111. Girija SS (2016) Tensorflow: large-scale machine learning on heterogeneous distributed systems. Software available from tensorflow. org 39(9)
112. Choi D, et al. (2019) On empirical comparisons of optimizers for deep learning. arXiv preprint arXiv:1910.05446
113. Reddy RVK, Rao BS, Raju KP (2018) Handwritten hindi digits recognition using convolutional neural network with rmsprop optimization. In: 2018 second international conference on intelligent computing and control systems (ICICCS). IEEE, pp 45–51
114. Tran PT et al (2019) On the convergence proof of amsgrad and a new version. IEEE Access 7:61706–61716
115. Feurer M, Hutter F (2019) Hyperparameter optimization. In: Automated machine learning. Springer, Cham, pp 3–33
116. Culotta A, Kanani P, Hall R, Wick M, McCallum A (2007) Author disambiguation using error-driven machine learning with a ranking loss function in Sixth International Workshop on Information Integration on the Web (IIWeb-07). Vancouver, Canada
117. Zhang Z, Sabuncu MR (2018) Generalized cross entropy loss for training deep neural networks with noisy labels. In: 32nd conference on neural information processing systems (NeurIPS)
118. Kavalerov I, Czaja W, Chellappa R (2021) A multi-class hinge loss for conditional gans. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp 1290–1299
119. Yu D, Yao K, Su H, Li G, Seide F (2013) Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition. In: 2013 IEEE international conference on acoustics, speech and signal processing. IEEE, pp 7893–7897
120. Singh SK, Singh U, Kumar M (2014) Estimation for the parameter of poisson-exponential distribution under bayesian paradigm. J Data Sci 12(1):157–173
121. Bach S, Huang B, London B, Getoor L (2013) Hinge-loss markov random fields: Convex inference for structured prediction. arXiv preprint arXiv:1309.6813
122. Wu Y, Liu Y (2007) Robust truncated hinge loss support vector machines. J Am Stat Assoc 102(479):974–983
123. He F, Liu T, Tao D (2019) Control batch size and learning rate to generalize well: theoretical and empirical evidence. Adv Neural Inf Process Syst 32:1143–1152
124. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958
125. Gandomi AH, Yang XS, Talatahari S, Alavi AH (2013) Metaheuristic algorithms in modeling and optimization. Metaheuristic Appl Struct. Infrastruct 1–24
126. Yang XS (2011) Metaheuristic optimization. Scholarpedia 6(8):11472
127. Abualigah L et al (2021) Aquila optimizer: a novel metaheuristic optimization algorithm. Comput Indust Eng 157:107250
128. AlRassas AM et al (2021) Optimized anfis model using aquila optimizer for oil production forecasting. Processes 9(7):1194
129. Bloice MD, Stocker C, Holzinger A (2017) Augmentor: an image augmentation library for machine learning. arXiv preprint arXiv:1708.04680
130. Frid-Adar M et al (2018) Gan-based synthetic medical image augmentation for increased cnn performance in liver lesion classification. Neurocomputing 321:321–331
131. Shorten C, Khoshgoftaar TM (2019) A survey on image data augmentation for deep learning. J Big Data 6(1):1–48
132. Perez L, Wang J (2017) The effectiveness of data augmentation in image classification using deep learning. arXiv preprint arXiv:1712.04621
133. Patro S, Sahu KK (2015) Normalization: A preprocessing stage. arXiv preprint arXiv:1503.06462
134. Bisong E (2019) Building machine learning and deep learning models on Google cloud platform: a comprehensive guide for beginners. (Apress)

135. Bentley P, Nordehn G, Coimbra M, Mannor S (2011) The PASCAL classifying heart sounds challenge (CHSC2011) Results (http://www.peterjbentley.com/heartchallenge/index.html)
136. Balaha HM, Saafan MM (2021) Automatic exam correction framework (aecf) for the mcqs, essays, and equations matching. IEEE Access 9:32368–32389
137. Carrington AM et al (2020) A new concordant partial auc and partial c statistic for imbalanced data in the evaluation of machine learning algorithms. BMC Med Inform Decis Mak 20(1):1–12
138. Narváez P, Gutierrez S, Percybrooks WS (2020) Automatic segmentation and classification of heart sounds using modified empirical wavelet transform and power features. Appl Sci 10(14):4791
139. Nogueira DM, Ferreira CA, Gomes EF, Jorge AM (2019) Classifying heart sounds using images of motifs, mfcc and temporal features. J Med Syst 43(6):1–13
140. Akram MU et al (2018) Analysis of pcg signals using quality assessment and homomorphic filters for localization and classification of heart sounds. Comput Methods Programs Biomed 164:143–157
141. DEPERLİGLU Ö (2018) Classification of segmented heart sounds with artificial neural networks. Int J Appl Math Electron Comput 6(4):39–44
142. Banerjee M, Majhi S (2020) Multi-class heart sounds classification using 2d-convolutional neural network. In: 2020 5th international conference on computing, communication and security (ICCCS). IEEE, pp 1–6
143. Deperlioglu O (2018) Classification of phonocardiograms with convolutional neural networks. BRAIN. Broad Res Artif Intell Neurosci 9(2):22–33
144. Bilal EM (2021) Heart sounds classification using convolutional neural network with 1d-local binary pattern and 1d-local ternary pattern features. Appl Acoust 180:108152
145. Alday EAP et al (2020) Classification of 12-lead ecgs: the physionet/computing in cardiology challenge 2020. Physiol Meas 41(12):124003
146. Holzinger A, Malle B, Saranti A, Pfeifer B (2021) Towards multi-modal causability with graph neural networks enabling information fusion for explainable ai. Inf Fusion 71:28–37