# Detection of Cardiovascular Diseases in ECG Images Using Machine Learning and Deep Learning Methods

Mohammed B. Abubaker and Bilal Babayiğit

***Abstract*—Cardiovascular diseases (heart diseases) are the leading cause of death worldwide. The earlier they can be predicted and classified; the more lives can be saved. Electrocardiogram (ECG) is a common, inexpensive, and noninvasive tool for measuring the electrical activity of the heart and is used to detect cardiovascular disease. In this article, the power of deep learning techniques was used to predict the four major cardiac abnormalities: abnormal heartbeat, myocardial infarction, history of myocardial infarction, and normal person classes using the public ECG images dataset of cardiac patients. First, the transfer learning approach was investigated using the low-scale pretrained deep neural networks SqueezeNet and AlexNet. Second, a new convolutional neural network (CNN) architecture was proposed for cardiac abnormality prediction. Third, the aforementioned pretrained models and our proposed CNN model were used as feature extraction tools for traditional machine learning algorithms, namely support vector machine, K-nearest neighbors, decision tree, random forest, and Naïve Bayes. According to the experimental results, the performance metrics of the proposed CNN model outperform the exiting works; it achieves 98.23% accuracy, 98.22% recall, 98.31% precision, and 98.21% F1 score. Moreover, when the proposed CNN model is used for feature extraction, it achieves the best score of 99.79% using the NB algorithm.**

***Impact Statement*—Artificial intelligence plays an important role in improving the quality of life. In particular, early detection of diseases can help save lives. In this work, the proposed new lightweight CNN architecture has improved the accuracy rate of cardiovascular disease classification to 98.23% compared with the existing state-of-the-art methods, using the dataset of ECG images of cardiac patients, and can be performed on a single CPU, overcoming the limitation of computational power. In addition, the classification accuracy has significantly improved after applying the proposed method as a feature extraction tool for traditional machine learning algorithms. For example, an accuracy of 99.79% has been achieved using the Naïve Bayes algorithm. Thus, this method could be integrated into the IoT ecosystem in healthcare. This will encourage other AI researchers to explore other methods for cardiovascular disease detection.**

***Index Terms*—Cardiovascular, deep learning, electrocar diogram (ECG) images, feature extraction, machine learning, transfer learning.**

Manuscript received 2 March 2022; accepted 11 March 2022. Date of publication 15 March 2022; date of current version 24 March 2023. This article was recommended for publication by Associate Editor Mihail Popescu upon evaluation of the reviewers' comments. *(Corresponding author: Mohammed B. Abubaker)*

Mohammed B. Abubaker is with the Department of Computer Engineering, Erciyes University, 38039 Melikgazi, Turkey, and also with the Palestine Technical College, Gaza P920, Palestine (e-mail: mabubaker@ptcdb.edu.ps).

Bilal Babayiğit is with the Department of Computer Engineering, Erciyes University, 38039 Melikgazi, Turkey (e-mail: bilalb@erciyes.edu.tr).

Digital Object Identifier 10.1109/TAI.2022.3159505

## I. Introduction

According to the World Health Organization, cardiovascular diseases (heart diseases) are the leading cause of death worldwide. They claim an estimated 17.9 million lives each year, accounting for 32% of all deaths worldwide. About 85% of all deaths from heart disease are due to heart attacks, also known as myocardial infarctions (MI) [1]. Many lives can be saved if an efficient diagnosis of cardiovascular disease is detected at an earlier stage [1]. Different techniques are used in the healthcare system to detect heart diseases, such as electrocardiogram (ECG), echocardiography (echo), cardiac magnetic resonance imaging, computed tomography, blood tests, etc. [2], [3]. The ECG is a common, inexpensive, and noninvasive tool for measuring the electrical activity of the heart [4]. It is used to identify heart-related cardiovascular diseases [4], [5]. A highly skilled clinician can detect heart disease from the ECG waves. However, this manual process can lead to inaccurate results and is very time-consuming [5].

There is great potential to benefit from advances in artificial intelligence in healthcare to reduce medical errors. In particular, the use of machine learning and deep learning techniques for automatic prediction of heart diseases [3], [6]–[10]. The machine learning methods require an expert entity for features extraction and selection to identify the appropriate features before applying the classification phase. Feature extraction is a process of reducing the number of features in a data set by transforming or projecting the data into a new lower-dimensional feature space preserving the relevant information of the input data [11], [12].

The concept of feature extraction is concerned with creating a new set of features (different from the input feature) that are a combination of original features into a lower-dimensional space that extract most, if not all, of the information in input data. The most well-known feature extraction method is a principal component analysis [13], [14]. However, feature selection is a process of removing irrelevant and redundant features (dimensions) from the data set in the training process of machine learning algorithms. Various methods can be used for feature selection, classified as unsupervised, which refers to the method that does not need the output label for feature selection, and supervised, which refers to the methods that use output label for feature selection. Under supervised feature selection, there are three methods: the filter method, the wrapper method, and the embedded method [11], [12].

Many machine learning methods have been used for predicting cardiovascular diseases. Soni *et al.* [15] compared several

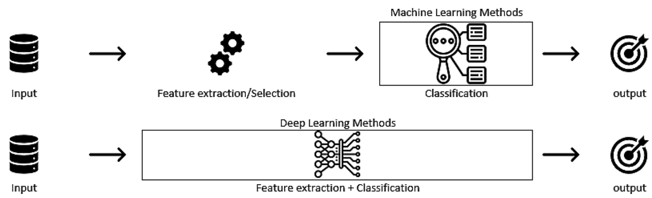


Fig. 1. Abstract concept of machine learning and deep learning.

machine learning algorithms, such as decision tree (DT), Naïve Bayes (NB), K-nearest neighbors (K-NN), and neural network (NN) on UCI Cleveland heart disease dataset. They concluded that DT had the highest accuracy of 89%. Dissanayake and Md Johar [16] studied the effect of the feature selection process on machine learning classifiers for predicting heart diseases from the UCI Cleveland heart disease dataset. They examined different feature selection techniques, such as ANOVA, Chi-square, forward and backward feature selection, and Lasso regression. After that, they applied six machine learning classifiers, which are DT, random forest (RF), support vector machine (SVM), K-NN, logistic regression (LR), and Gaussian NB (GNB). With the feature selection process, the prediction accuracy was improved such that using the backward feature selection method, the highest classification accuracy of 88.52% has been achieved with the DT classifier. The use of machine learning algorithms, such as NB, SVM, and DT algorithms, was studied in [17] using ten-fold cross-validation, on the South African heart disease dataset with 462 instances. The best results were obtained from NB for detecting heart disease with an accuracy rate of 71.6%, sensitivity of 63%, and specificity of 76.16%. Kim *et al.* [18] compared NN, SVM, classification based on multiple association rule (CMAR), DT, and NB algorithms to predict cardiovascular diseases on two types of datasets consisting of ultrasound images of carotid arteries (CAs) and heart rate variability (HRV) of the ECG signal. The combined extracted features from the CAs+ HRV dataset obtained higher accuracy than the separated features of CAs and HRV. Thus, SVM and CMAR classifiers outperformed the others by the accuracy of 89.51% and 89.46%, respectively.

On the other hand, deep learning, which is a subfield of machine learning, automatically extracts important features and patterns from the training datasets for the classification phase without the intervention of separate entities for features extraction and selection. Fig. 1 illustrates the abstract concept of machine learning and deep learning. In deep learning, a model is created by constructing multiple hidden layers of NNs. Convolutional neural network (CNN) is a deep learning method, which has achieved satisfactory results on image classification tasks.

The power of deep learning and pretrained networks can be used for feature extraction without having to retrain the whole network, transfer learning, and classification [19]. In this article, the pretrained networks, i.e., SqueezeNet [20] and AlexNet [21], are used as a transfer learning approach to study their performance in heart disease classification and as feature extraction for traditional machine learning methods for heart disease classification. In addition, a new CNN model is proposed for heart disease prediction using ECG images and used for feature extraction of the ECG images after training the new proposed CNN model.

The main contributions of this study are summarized as follows.

1) A new lightweight deep learning CNN architecture is proposed for cardiovascular diseases prediction using 12 lead-based ECG images.
2) The proposed CNN model achieves a success rate of 98.23%, outperforming the existing work [22] and the state-of-the-art low-scale SqueezeNet and AlexNet, which achieved 95.10%, 95.47%, and 96.79%, respectively.
3) To the best of our knowledge, this is the second study using the ECG images dataset of cardiac patients [23], which will encourage other researchers to explore other methods to detect cardiovascular diseases using this dataset.
4) The transfer learning approach using SqueezeNet and Alex-Net was investigated and compared with the proposed model.
5) The pretrained networks SqueezeNet, AlexNet and our proposed CNN model were used as feature extractors to apply the extracted features to the conventional machine learning methods: SVM, K-NN, DT, RF, and NB. The best results were achieved by our proposed CNN model for the NB algorithm in which the accuracy rate was reported as 99.79%.

The rest of this article is organized as follows. Section II presents the literature review. The methods and the proposed CNN model used in this article are explained in Section III. Section IV illustrates the dataset and experimental settings used. Section V provides the results and discussions, whereas Section VI concludes this article and provides future perspectives.

## II. Literature Review

Many research works [24]–[27] have been conducted for automatically predicting cardiovascular diseases using machine learning and deep learning methods by utilizing ECG as digitals or images data representation.

Bharti *et al.* [28] compared machine learning and deep learning methods on the UCI heart disease dataset to predict two classes. The deep learning method achieved the highest accuracy rate of 94.2%. In their architecture of deep learning model, they used three fully connected layers: the first layer consists of 128 neurons followed by a dropout layer with 0.2 rate, the second layer consists of 64 neurons followed by a dropout layer with 0.1 rate, and the third layer consists of 32 neurons. The machine learning methods with features selection and outliers' detection achieved accuracy rates as: RF is 80.3%, LR is 83.31%, K-NN is 84.86%, SVM is 83.29%, DT is 82.33%, and XGBoost is 71.4%. The research in [29] concluded that deep learning has proven to be a more accurate and effective technology for a variety of medical problems such as prediction. Deep learning methods will replace the traditional machine learning based on feature engineering. Kiranyaz *et al.* [30] proposed a CNN that consisted of three layers of an adaptive implementation of one-dimensional (1-D) convolution layers. This network was trained on the MIT-BIH arrhythmia dataset to classify long ECG data stream. They achieved accuracy rates of 99% and 97.6% in classifying ventricular ectopic beats and supraventricular ectopic beats, respectively. Also, the work in [31] proposed a CNN that consisted of three 1-D convolution layers, three max-pooling layers, one fully connected layer, and one softmax layer. The filter size for the first and second convolutional layers was set to 5 and a stride of 2 was used for the first two max-pooling layers. They achieved an accuracy rate of 92.7% in classifying ECG heartbeats using the MIT-BIH arrhythmia dataset.

Khan *et al.* [22] applied transfer learning approach using the pretrained single shot detector (SSD)-MobileNet-v2 [32] to detect cardiovascular diseases from the ECG images dataset of cardiac patients by predicting the four major heart abnormalities: abnormal heartbeat (AH), MI, history of MI (H. MI), and normal person (NP) classes. As preprocessing steps, the data size was adjusted and the 12 leads of each ECG image were labeled. SSD is used to classify and localize the objects in one step. The dataset was split 80% for training and 20% for testing. They used a batch size of 24, 200K training iterations for the training step, and a learning rate of 0.0002 to train their model. Their training phase lasted almost 4 days. They achieved a high precision rate for the MI class, i.e., 98.3%.

Rahman *et al.* [33] provided a deep CNN transfer learning approach to predict COVID-19 and four major cardiac abnormalities using ECG images. The dataset contained five classes: COVID-19, AH, MI, H. MI, and NP classes. Six different pretrained deep CNN models, i.e., ResNet18, ResNet50, ResNet101, DenseNet201 [34], Inception-V3 [35], and MobileNet-v2, were used for classification. Gamma correction, image resizing, and $z$-score normalization were used as preprocessing steps for the ECG images. As a result, for two-class classification (COVID-19 and normal) and three-class classification (COVID-19, normal, and other cardiac abnormalities), DenseNet201 outperformed the other networks with accuracy rates of 99.1% and 97.36%, respectively. For the five-class classification, Inception-V3 outperformed the other networks with an accuracy rate of 97.83%.

Pal *et al.* [36] presented a deep CNN transfer learning strategy using pretrained DenseNet arrhythmia classifications (AH) from ECG signals in PTB and MIT-BIH arrhythmia datasets converted to 2-D images. Since the dataset was imbalanced, a data augmentation technique was applied to the data. The DenseNet model was chosen because it provides a solution to the vanishing gradient problem in deep networks by using dense connections between layers. Their model was referred to as CardioNet. The precision, recall, and F1 score values were 98.62%, 98.68%, and 98.65%, respectively.

Avanzato and Beritelli [37] proposed a deep CNN with four 1-D convolutional layers for detecting three classes of cardiac abnormalities using ECG signals in the MIT-BIH arrhythmia dataset. Each convolutional layer was followed by a batch normalization layer, a rectifier linear unit (ReLU) layer activation function, and a max-pooling layer with a filter (kernel) size of 4. A size 80 filter was used for the first convolutional layer, and the others had a filter size of 4. This architecture did not use fully connected layers for classification, but instead used an average pooling layer followed by a softmax layer. This model achieved an accuracy rate of 98.33%.

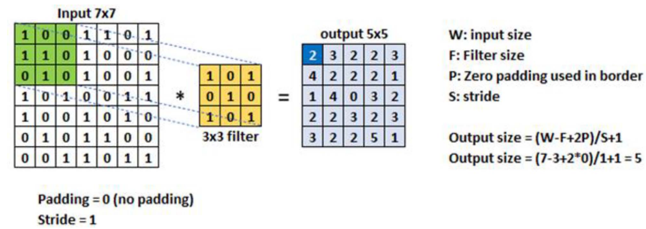


Fig. 2. Example of a convolution operation.

Acharya *et al.* [38] implemented a deep CNN with four 1-D convolutional layers and three fully connected layers for detecting MI using ECG signals in the PTB dataset. In this model, the leaky rectifier linear unit (LeakyRelu) was used as the activation function layer. Each convolutional layer was followed by a max-pooling layer with a filter size of 2 and a stride of 2. The filter sizes for convolutional layers were 102, 24, 11, and 9 in order. The numbers of neurons for fully connected layers were 30, 10, and 2 in that order. The last fully connected layer was followed by a softmax layer. They achieved an average accuracy rate of 93.53% and 95.22% for ECG beats with and without noise removal, respectively.

Naz *et al.* [39] converted ECG signals into $32 \times 32$ binary images. Their model was tested with the MIT-BIH dataset using the pretrained CNN models AlexNet, VGG19, and Inception-V3 to detect ventricular arrhythmias of the heart. Transfer learning was performed to extract and concatenate features from the pretrained models. SVM and K-NN classification methods were then used for binary classification. Using the SVM, they achieved an accuracy of 97.60%.

## III. Methods

### A. Convolutional Neural Networks (CNN)

In deep learning, a CNN is a type of deep artificial NN specifically designed for image classification and processing [40]. The neurons in CNNs are arranged in three dimensions: height, width, and depth (channel). For example, an input image is $227 \times 227 \times 3$, which means that the width and height of the input image are 227 and the depth (channel) is 3. The main task of CNNs is to extract important features from the input images. The two main components of CNNs are convolutional layers and pooling layers. The higher layers in CNNs can be fully connected layers and the last layer is a sigmoid or softmax activation function layer to get the predicted output. The convolution process is performed with convolutional layers on the input data using a filter or kernel to create a feature map representing the detected features of the input. Convolution is performed by sliding the filter over the input. At each position, matrix multiplication is performed and the result is summed onto the feature map. Fig. 2 shows a simple example of a convolution process for an input with a depth of 1.

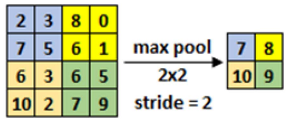


Fig. 3. Example of 2 × 2 max-pooling with stride = 2.

The convolution process is linear. To add nonlinearity to the output, the convolution layer is followed by an activation function layer such as ReLU or its variants. After the convolution layer, a pooling layer such as max-pooling layer could be used to downsample the feature map to reduce the computational cost. Fig. 3 shows a simple example of max-pooling for an input with depth of 1.

### B. Pretrained Deep Learning Models

The pretrained deep NNs can be used for transfer learning, feature extraction, and classification. In this article, low-scaled SqueezeNet and AlexNet pretrained CNN networks that can be executed on a single CPU are used for transfer learning and feature extraction.

The transfer learning approach is commonly used with pretrained deep NNs applied to a new dataset. Therefore, it could benefit from the pretrained network that has already learned a variety of features that can be transferred to other similar tasks. Most of the pretrained networks have been trained with more than a million images and can classify images into 1000 object classes. In applying the transfer learning approach, the final layers of the pretrained network are replaced with new layers to learn the specific features of the new dataset. Then, the model is fine-tuned by training it on a new training dataset with specific training parameters and testing its performance measure on a new test dataset.

The pretrained deep NNs can be used as a feature extraction tool without wasting time and effort on training. In this article, the extracted features from the pretrained networks are used to train traditional machine learning classifiers, namely SVM [41], K-NN [42], DT [43], RF [44], and NB [45]. The details of using the pretrained networks are explained in the next Sections.

### C. Proposed CNN Architecture

The proposed CNN model contains besides the input and output layers, six 2-D convolutional layers, three fully connected layers, three max-pooling layers, eight leaky ReLU layers, eight batch normalization layers, five dropout layers, two depth concatenation layers, and one softmax layer. In total, there are 38 layers. The architecture of the proposed model is shown in Fig. 4.

The proposed CNN model consists of two branches that help extract more representative features, namely the stack branch and the full branch. The proposed CNN model accepts input image of size 227×227×3. The input image flows into the two branches simultaneously.

The stack branch consists of three stacked 2-D 3×3 convolutional layers. Each of these 2-D convolutional layers is followed by the leakyReLU layer, the batch normalization layer, and the max-pooling layer. In the leakyReLU layer, a leakyReLU activation function with a scale of 0.1 is used. Unlike ReLU, leakyReLU has a slight slope in the negative range, which can eliminate the problem of dying neurons [46]. The batch normalization layer is used to normalize its inputs for each minibatch, which can train the model faster and increase the accuracy of the model. The max-pooling layer applies the max-pooling operation to the feature map by selecting the maximum element from the region covered by the filter. This helps to reduce the spatial size of the feature map to reduce the number of parameters and computational cost in the model. The proposed CNN model uses max-pooling layers of 6×6 filter size with a stride of 3. In this branch, 64, 128, and 224 filters are used to extract deep

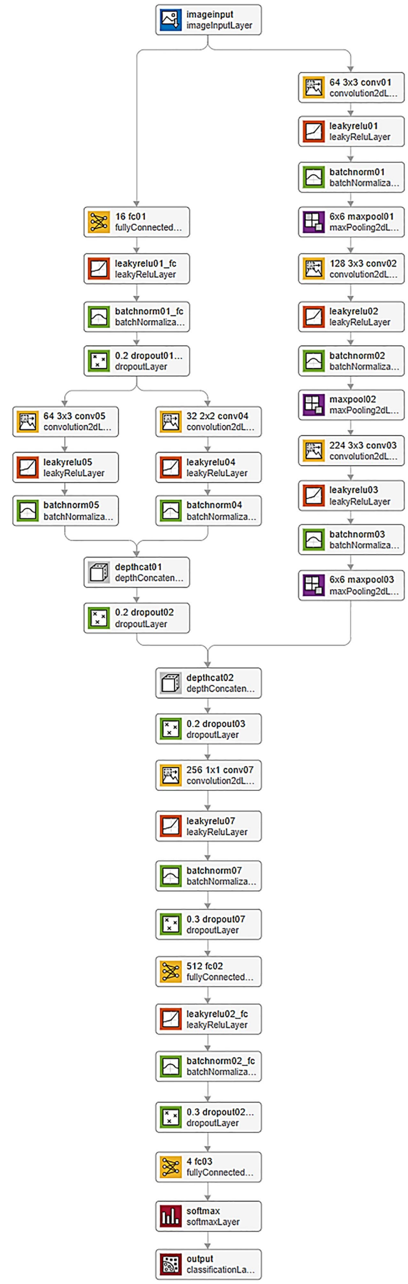


Fig. 4. Representation architecture of the proposed CNN model.

TABLE I
LAYERS ANALYSIS OF THE PROPOSED CNN MODEL

| No. | Type | Name | Properties | Input size | Output size |
|---|---|---|---|---|---|
| 1. | Image Input | Imageinput | - | 227×227×3 | 227×227×3 |
| 2. | Convolution | conv01 | 64, 3x3, stride = 2, padding = same | 227×227×3 | 114×114×64 |
| 3. | Max Pooling | maxpool01 | 6x6, stride = 3, padding = same | 114×114×64 | 38×38×64 |
| 4. | Convolution | conv02 | 128, 3x3, stride = 2, padding = same | 38×38×64 | 19×19×128 |
| 5. | Max Pooling | maxpool02 | 6x6, stride = 3, padding = same | 19×19×128 | 7×7×128 |
| 6. | Convolution | conv03 | 224, 3x3, stride = 2, padding = same | 7×7×128 | 4×4×224 |
| 7. | Max Pooling | maxpool03 | 6x6, stride = 3, padding = same | 4×4×224 | 2×2×224 |
| 8. | Fully Connected | fc01 | 16 | 227×227×3 | 1×1×16 |
| 9. | Dropout | dropout01_fc | 0.2 | 1×1×16 | 1×1×16 |
| 10. | Convolution | conv04 | 32, 2x2, stride = 1, padding = 1 | 1×1×16 | 2×2×32 |
| 11. | Convolution | conv05 | 64, 3x3, stride = 2, padding = 2 | 1×1×16 | 2×2×64 |
| 12. | Depth concatenation | depthcat01 | Two inputs of size 2×2×32, 2x2x64 | | 2×2×96 |
| 13. | Dropout | dropout02 | 0.2 | 2×2×96 | 2×2×96 |
| 14. | Depth concatenation | depthcat02 | Two inputs of size 2×2×96, 2x2x224 | | 2×2×320 |
| 15. | Dropout | dropout03 | 0.2 | 2×2×320 | 2×2×320 |
| 16. | Convolution | conv07 | 256, 1x1, stride = 1, padding = same | 2×2×320 | 2×2×256 |
| 17. | Dropout | dropout07 | 0.3 | 2×2×256 | 2×2×256 |
| 18. | Fully Connected | fc02 | 512 | 2×2×256 | 1×1×512 |
| 19. | Dropout | dropout02_fc | 0.3 | 1×1×512 | 1×1×512 |
| 20. | Fully Connected | fc03 | 4 | 1×1×512 | 1×1×4 |
| 21. | Softmax | Softmax | - | 1x1x4 | 1x1x4 |
| 22. | Classification Output | Output | Cross-entropy as loss function | 1x1x4 | 1x1x4 |

leakyReLU: scale=0.1, batch normalization: MeanDecay=0.1, VarianceDecay=0.1, Epsilon=0.00001, total number of learnable parameters=3430308

features of the data for the first, second and third convolutional layers, respectively. The size of the output at the end of the stack branch is 2 × 2 × 224.

The first layer in the full branch of our proposed CNN model is a fully connected layer, hence its name. In our model, the fully connected layer contains 16 neurons. Each neuron in a fully connected layer is connected to each neuron in the previous layer. This is in contrast to a neuron in a convolutional layer, which is connected to some neurons in the previous layer defined by the size of the convolutional filter. Although most of the parameters in the CNN come from the fully connected layers, the number of calculations in the convolutional layer requires much more memory. The fully connected layer is followed by a leakyReLU layer, a batch normalization layer, and a dropout layer, which helps to reduce overfitting and emphasize the generalization capability of the model. As can be seen in Fig. 4, the two convolutional layers, named conv04 and conv05, are located at the same level after the block of the fully connected layer to help extract broader features.

Conv04 is a 32 2 × 2 convolutional layer with a stride of 1 and a padding of 1, whereas conv05 is a 64 3×3 convolutional layer with a stride of 2 and a padding of 2. The feature maps of these two convolutional layers are concatenated to produce a feature map of 2 × 2 × 96. After concatenating the features, a dropout layer is applied to reduce the impact of correlated features and avoid overfitting.

The two outputs generated by the two branches are concatenated to create a feature map of 2×2×320. Then, a dropout layer is added to reduce the overfitting of the model. A 1×1 convolutional layer with 256 filters is added to increase the nonlinearity of the model and reduce the depth or number of feature maps to reduce computational cost. A fully connected layer with 512 neurons is added to strengthen the classification process. For the output, a fully connected layer with four neurons corresponding to the number of classes to be classified, followed by a softmax layer to obtain the predicted output. The analysis of the trained network for the proposed CNN model is given in Table I.

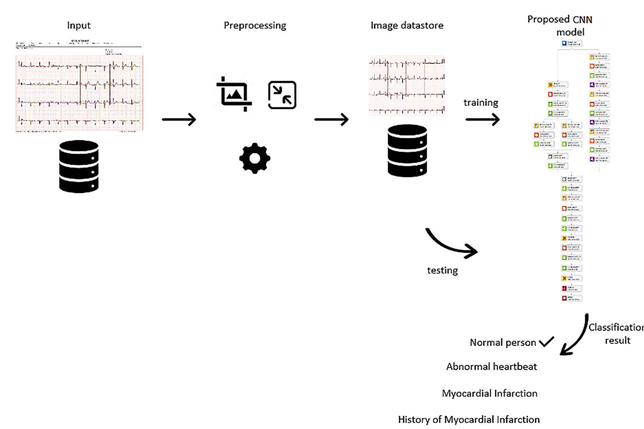


Fig. 5. Schematic of using the proposed CNN model for ECG images of cardiac patients' classification.

The schematic of using the proposed CNN model for the classification of ECG images of cardiac patients is shown in Fig. 5. First, the input images are preprocessed by cropping, resizing, and augmenting them. Then, the preprocessed images are stored in the image datastore. The proposed model is trained

TABLE II
PUBLIC ECG IMAGES DATASET DESCRIPTION

| No. | Class | Number of images |
|---|---|---|
| 1. | Normal person | 284 |
| 2. | Abnormal Heartbeat | 233 |
| 3. | Myocardial Infarction | 239 |
| 4. | History of Myocardial Infarction | 172 |
| Total | | 928 |

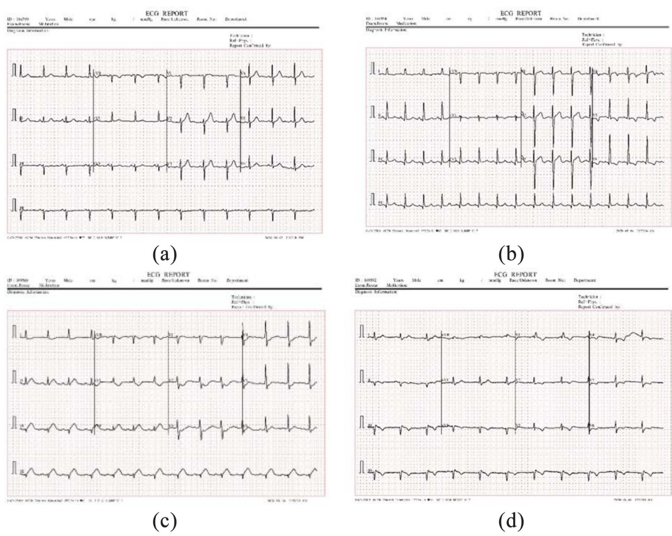


Fig. 6. Samples from the ECG images dataset. (a) NP. (b) AH. (c) MI. (d) H. MI.

with the above-mentioned training parameters using the ECG images stored in the image datastore. The model learns the features and adjusts its learnable parameters accordingly. After training, the model is ready to test ECG images for classifying cardiac abnormalities as one of the four classes: NP, AH, MI, and H. MI.

## IV. EXPERIMENTS

### A. ECG Images Dataset of Cardiac Patients

The mentioned methods were tested on the ECG Images dataset of cardiac patients [23]. This dataset contains 928 different patient records with 4 different classes as shown in Table II. These four classes are NP, AH, MI, and H. MI. Fig. 6 depicts some samples from the dataset. An NP is a healthy person who does not have any heart abnormalities. An AH (arrhythmia) occurs when the electrical impulses in the heart become too fast, too slow, or irregular so that the heart beats irregularly. MI, also known as heart attack, occurs when blood flow in the coronary artery of the heart decreases or stops, causing damage to the heart muscle. The patients with an H. MI have recently recovered from MI or heart attack.

### B. Experimental Settings

The experiments were conducted with MATLAB 2021b on Intel core$^{TM}$ i7-4510U CPU @ 2.00 GHz with 8GB RAM and

Fig. 7. Sample from the ECG images dataset after performing cropping as a preprocessing.

TABLE III
TRAINING PARAMETERS AND VALUES FOR DEEP LEARNING METHODS

| Optimizer | Weight Initializer | Bias Initializer | L2 Regularization | Epochs no. | Mini-Batch Size |
|---|---|---|---|---|---|
| Adam | Xavier | Zeros | 0.0001 | 16 | 128 |

a 4 GB NVIDIA GeForce 820M GPU[1] and running Windows 10 Pro 64-b.

*Preprocessing*. As can be seen in Fig. 6, the ECG images in the dataset contain header and footer information that have no relation to the features we need. Therefore, we have applied cropping for all images to focus on the valuable features as shown in Fig. 7. In addition, all ECG images were resized to the same resolution of 227$\times$227 with 3 channels (RGB) before performing model training.

*Data augmentation*. To increase the robustness and accuracy of the developed model, data augmentation was applied to the dataset [47]. It helps to increase the number of images in the dataset and eliminate the effects of training the model on an imbalanced dataset. Three augmentation techniques (rotation, flipping, and translation) were applied to the given dataset [48]. This increased the number of images in the dataset to 4700 images.

*Deep learning training parameters*. Since the optimization of hyperparameters is very computationally intensive, all experiments were performed using the training parameters listed in Table III. Adam optimizer is applied to train the model for 16 epochs with a minibatch size of 128. However, since the initial learning rate value (LR) is the most important hyperparameter, different values for LR were used in the experiments, which are mentioned in the next section. According to these parameters, the number of iterations per epoch is 29 and the number of iterations for training the model is 464.

To obtain reliable results when testing and evaluating the model, fivefold cross-validation was performed. In this process, the dataset is divided into five parts, with four parts used for the training phase and the remaining part used for the testing phase (3760 images for training and 940 images for testing). Thus, five different distinctions between training and testing were applied. The results are the average of the five folds.

[1]Its compute capability is 2.1 and it is not supported by MATLAB 2021b. Hence, all experiments were run on a single CPU.

TABLE IV
PERFORMANCE MEASURES

| Measures | Defined as | |
|---|---|---|
| Accuracy | *(TP+TN)/(TP+FP+FN+TN)* | (1) |
| Recall | *TP/(TP+FN)* | (2) |
| Precision | *TP/(TP+FP)* | (3) |
| F1 score | *(2 × Recall × Precision)/(Recall+Precision)* | (4) |

TABLE V
NETWORKS PROPERTIES[2]

| Network | Depth | No. of Layers | No. of Connections | No. of Parameters (million) |
|---|---|---|---|---|
| SqueezeNet | 18 | 68 | 75 | 1.24 |
| AlexNet | 8 | 25 | 24 | 61.0 |
| Proposed CNN | 6 | 38 | 39 | 3.43 |

For all networks, input image size is 227×227×3.

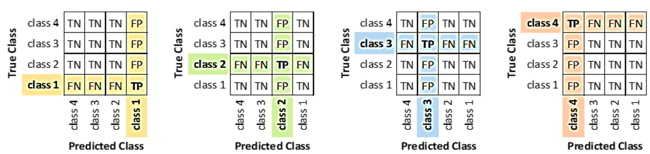


Fig. 8. Semantic of the confusion matrices for four classes results.

## V. RESULTS AND DISCUSSIONS

For performance analysis, accuracy, precision, recall, F1 score, and training and testing times were used. These measurements are based on the analysis of the data in a confusion matrix. Table V shows how the measurements are defined based on the confusion matrix. Where the accuracy is the percentage of positively predicted observations relative to the total number of observations. Recall represents the ratio of positively predicted observations to all observations in the true class (should be positively estimated). Precision expresses the ratio of positively predicted observations to all observations in the predicted class (should be positively predicted). The F1 score is the weighted average of both Recall and Precision. Thus, it takes into account both the false negatives and the false positives values.

Fig. 8 shows the semantics of the confusion matrix for 4 classes datasets as in our case, the ECG images dataset of cardiac patients. The performance measurements of the experiments are calculated based on the equations given in Table IV.

### A. Results of Transfer Learning and Proposed CNN Model

The state-of-the-art architectures of the pretrained networks SqueezeNet and AlexNet were used to apply the transfer learning approach in our study. Both were originally trained for the classification of 1000 image classes. To retrain these networks for classifying the new set of ECG images in the dataset, we replace the last layers of these models to make them suitable for the new task. In AlextNet, the last fully connected layer is replaced with a new fully connected layer containing the same

[2]Here, the total number of layers in the network was counted, not even the convolutional layers and dense layers.

TABLE VI
CALCULATED PERFORMANCE MEASUREMENTS FOR SQUEEZE-NET, ALEXNET, AND THE PROPOSED CNN MODEL FOR DIFFERENT RL VALUES

| Model | LR | A. (%) | R. (%) | P. (%) | F1 (%) | T1 (m) | T2 (m) |
|---|---|---|---|---|---|---|---|
| Squeeze-Net (Transfer learning) | 0.01 | 24.79 | 25.00 | NaN | NaN | 245.0 | 2.3 |
| | 0.001 | 24.15 | 25.00 | NaN | NaN | 212.7 | 2.1 |
| | 0.0001 | 95.47 | 95.43 | 96.07 | 95.40 | 219.9 | 2.2 |
| AlexNet (Transfer learning) | 0.01 | 24.15 | 25.00 | NaN | NaN | 198.9 | 2.1 |
| | 0.001 | 37.00 | 37.88 | NaN | NaN | 209.5 | 2.1 |
| | 0.0001 | 96.79 | 96.80 | 97.02 | 96.78 | 199.5 | 2.1 |
| Proposed CNN | 0.01 | 97.24 | 97.24 | 97.31 | 97.22 | **189.4** | **2.0** |
| | 0.001 | 97.89 | 97.89 | 97.97 | 97.88 | 190.0 | 2.1 |
| | 0.0001 | **98.23** | **98.22** | **98.31** | **98.21** | 190.7 | **2.0** |

LR: initial learning rate, A.: accuracy, R.: recall, P.: precision, F1: F1 score, T1: training time, T2: testing time.
The bold values indicate the best results.

TABLE VII
PERFORMANCE MEASUREMENTS VALUES OBTAINED FOR EACH FOLD OF THE PROPOSED MODEL

| LR | Folds | A. (%) | R. (%) | P. (%) | F1 (%) | T1 (m) | T2 (m) |
|---|---|---|---|---|---|---|---|
| 0.01 | Fold-1 | 97.77 | 97.75 | 97.83 | 97.73 | 200.23 | 2.05 |
| | Fold-2 | 97.87 | 97.86 | 97.89 | 97.86 | 185.88 | 2.00 |
| | Fold-3 | 95.43 | 95.48 | 95.57 | 95.39 | 185.08 | 1.97 |
| | Fold-4 | 97.13 | 97.11 | 97.26 | 97.11 | 184.67 | 2.05 |
| | Fold-5 | 97.98 | 97.98 | 98.02 | 97.99 | 191.00 | 1.98 |
| | **Average** | **97.24** | **97.24** | **97.31** | **97.22** | **189.37** | **2.01** |
| 0.001 | Fold-1 | 99.15 | 99.15 | 99.15 | 99.15 | 187.52 | 2.01 |
| | Fold-2 | 98.19 | 98.18 | 98.23 | 98.16 | 191.57 | 2.08 |
| | Fold-3 | 95.96 | 95.99 | 96.14 | 95.94 | 190.17 | 2.10 |
| | Fold-4 | 97.66 | 97.64 | 97.80 | 97.63 | 185.62 | 2.16 |
| | Fold-5 | 98.51 | 98.50 | 98.55 | 98.51 | 194.87 | 1.99 |
| | **Average** | **97.89** | **97.89** | **97.97** | **97.88** | **189.95** | **2.07** |
| 0.0001 | Fold-1 | 99.47 | 99.46 | 99.47 | 99.46 | 195.12 | 2.03 |
| | Fold-2 | 97.66 | 97.64 | 97.74 | 97.61 | 185.85 | 2.00 |
| | Fold-3 | 97.55 | 97.53 | 97.74 | 97.54 | 187.72 | 2.00 |
| | Fold-4 | 98.30 | 98.28 | 98.33 | 98.27 | 196.80 | 2.01 |
| | Fold-5 | 98.19 | 98.18 | 98.28 | 98.17 | 187.92 | 1.99 |
| | **Average** | **98.23** | **98.22** | **98.31** | **98.21** | **190.68** | **2.01** |

The bold values indicate the average of the five folds.

number of neurons as the number of our predicted classes, i.e., 4. However, since SqueezeNet does not use fully connected layers, we replace the last convolutional layer which is used to identify 1000 classes with a new convolutional layer containing 4 1×1 filters. For both pretrained networks used, a new classification layer is added in place of the existing one, which produces an output based on the probabilities computed by the softmax layer. The properties of the pretrained networks used and our proposed CNN are shown in Table V.

Table VI shows the performance measures of the pretrained models (SqueezeNet and AlexNet) used as the transfer learning approach and our proposed CNN model for the ECG images dataset. Each model was trained with different values for learning rate LR (0.01, 0.001, and 0.0001). As noticed, the most successful result with an average accuracy rate of 98.23% was obtained by our proposed CNN model when the RL was 0.0001. Table VII shows the detailed analysis of the proposed model. The

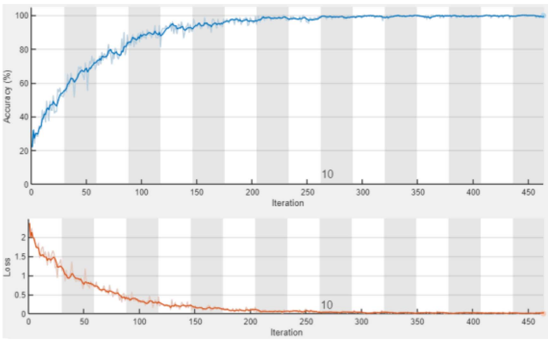

Fig. 9. Training Progress for our proposed CNN model on the ECG images dataset in fold-1 (LR: 0.0001 and other hyperparameters are as in Table III).

TABLE VIII
MODELS COMPARISON

| Model | Average Accuracy (%) | | | | |
|---|---|---|---|---|---|
| | NP | AH | MI | H. MI | Average |
| Work in [22] | 93.7 | 93.6 | 96.2 | 96.8 | 95.1 |
| Proposed CNN | 99.8 | 93.1 | 100.0 | 99.9 | 98.2 |
| | Average Precision (%) | | | | |
| | NP | AH | MI | H. MI | Average |
| Work in [22] | 96.2 | 97.2 | 98.3 | 98.3 | 97.5 |
| Proposed CNN | 97.4 | 100.0 | 99.4 | 96.5 | 98.3 |

NP: normal person, AH: abnormal heartbeat, MI: myocardial infarction, H. MI: history of myocardial Infarction classes.

average accuracy rate for the proposed CNN model shows similar high results when the RL values are changed. In contrast, the pretrained SqueezeNet and AlexNet models show poor results when the initial learning rate were 0.01 and 0.001, but they start to show slightly good results when the LR is set to 0.0001. This is because, in transfer learning, the weights of the pretrained models are not learned from scratch. Therefore, to avoid getting stuck in local minima, it is better to start with a lower LR such as 0.0001 when applying transfer learning techniques.

The average accuracy rates are 96.79% and 95.43% for AlexNet and SqueezeNet, respectively with RL of 0.0001. On the other hand, the proposed CNN model also outperforms the other models in terms of time cost, as can be seen in Table VI. Although SqueezeNet has the smallest number of parameters and is a fully CNN, it achieves the worst performance in terms of time cost. This is because the number of computations in the convolutional layers is very high, so it takes more time to be processed, especially when running on a single CPU platform.

Fig. 9 depicts the training progress of our proposed CNN model on the ECG images dataset in fold-1 (LR = 0.0001). The accuracy rate increases gradually with each successive iteration. Moreover, the loss decreases smoothly as the iteration progresses and reaches 0.0043.

The confusion matrices which were obtained for each fold after training our proposed CNN models with an RL value of 0.0001 on the ECG images dataset are shown in Fig. 10.

To the best of our knowledge, the only work in the literature that uses the same dataset and classifies the four classes is the work in [22], which has been discussed in Section II. In [22], the dataset was split as 80% and 20% for training and testing,

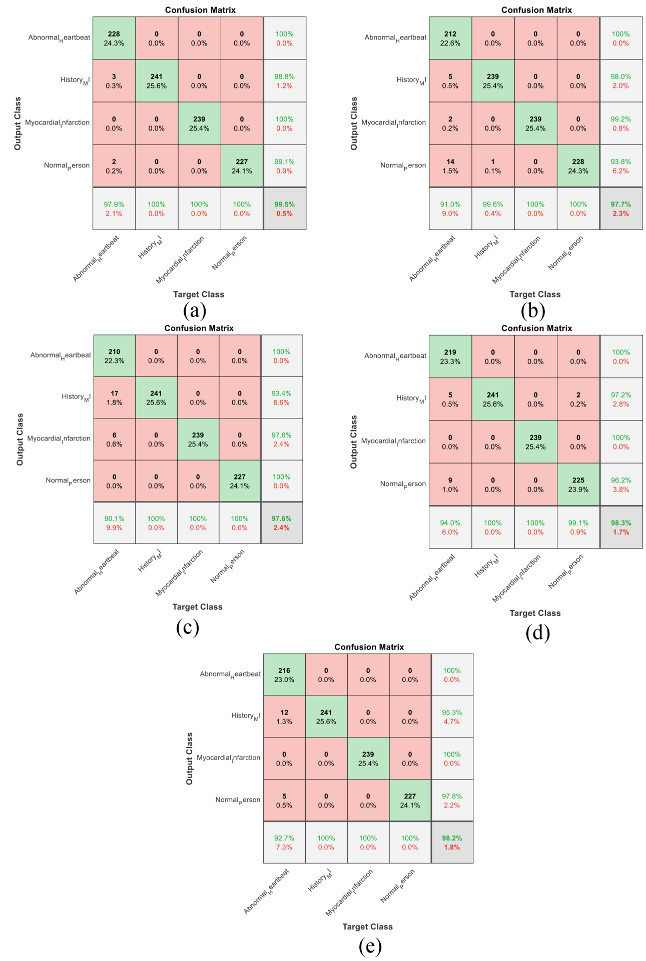

Fig. 10. Confusion matrices of the proposed CNN model for classification of heart diseases in the ECG images dataset for each fold (RL: 0.0001 and other hyper-parameters are as in Table III).

TABLE IX
PROPERTIES OF THE EXTRACTED FEATURES FROM PRETRAINED NETWORKS

| Pretrained Network | Training features size | Testing features size | Time taken (m) | Activation Feature Layer |
|---|---|---|---|---|
| SqueezeNet | 3760x196000 | 940x196000 | 13.37 | conv10 (64) |
| AlexNet | 3760x4096 | 940x4096 | 11.24 | fc7 (20) |
| Proposed CNN | 3760x512 | 940x512 | 10.77 | fc02 (32) |

respectively. They used a batch size of 24 and a learning rate of 0.0002 to train their model. Their training phase lasted almost 4 days. According to their paper, they achieved a high precision rate for class MI, which is 98.3%, whereas our proposed CNN model outperforms them with a precision rate of 99.4% for class MI. Table VIII compares the results from [22], in which the accuracy rates of each class were extracted from their confusion matrix, with our proposed CNN model.

### *B. Results of Using Pretrained Deep Learning Models As a Feature Extractor*

The pretrained SqueezeNet and AlexNet networks were used to extract the features of the ECG images in the dataset. As well

TABLE X
CALCULATED PERFORMANCE MEASUREMENTS FOR MACHINE LEARNING ALGORITHMS THAT USE PRETRAINED NETWORKS SQUEEZENET, ALEXNET, AND PROPOSED CNN AS FEATURES EXTRACTOR APPLIED ON ECG IMAGES DATASET

| Pretrained Network | Algorithm | Accuracy (%) | Recall (%) | Precision (%) | F1 score (%) | Training Time (s) | Testing Time (s) |
|---|---|---|---|---|---|---|---|
| SqueezeNet | SVM | 97.87 | 97.85 | 97.95 | 97.84 | 642.44 | 10.37 |
| | K-NN | 85.00 | 84.88 | 86.20 | 84.53 | 24.19 | 611.49 |
| | DT | 89.15 | 89.04 | 88.98 | 88.70 | 2159.1 | 1.15 |
| | RF | 98.94 | 98.93 | 98.95 | 98.92 | 7508.6 | 6.85 |
| | NB | 73.94 | 74.12 | 75.64 | 74.05 | 1245.7 | 39.75 |
| AlexNet | SVM | 97.66 | 97.64 | 97.79 | 97.64 | 6.3357 | 0.2333 |
| | K-NN | 90.32 | 90.18 | 91.13 | 90.17 | 1.0016 | 9.2097 |
| | DT | 91.38 | 91.29 | 91.43 | 91.19 | 11.8865 | 0.0656 |
| | RF | 97.55 | 97.53 | 97.66 | 97.51 | 24.3221 | 0.3720 |
| | NB | 72.02 | 72.18 | 73.31 | 71.89 | 2.6253 | 0.5649 |
| Proposed CNN | SVM | 99.47 | 99.46 | 99.47 | 99.46 | **0.2858** | 0.0372 |
| | K-NN | 99.68 | 99.68 | 99.68 | 99.68 | 1.6159 | 1.4100 |
| | DT | 99.04 | 99.03 | 99.05 | 99.03 | 0.3292 | **0.0129** |
| | RF | 99.57 | 99.57 | 99.58 | 99.57 | 3.3505 | 0.1597 |
| | NB | **99.79** | **99.79** | **99.78** | **99.78** | 0.4129 | 0.1163 |

The bold values indicate the best results.

as, our proposed CNN model was used as a feature extractor and the results were compared. The power of deep learning can be used to extract image features without re-training the entire network. The activations of the network are computed by forward propagation of the input images up to the specific feature layer. The activation feature layers used are conv10 (layer number 64), fc7 (layer number 20) and fc02 (layer number 32) for SqueezeNet, AlexNet and our proposed CNN model, respectively. Table IX illustrates the characteristics of the extracted features. Then, these extracted features were used to train the machine learning algorithms: SVM, k-NN, DT, RF, and NB.

The performance measures are calculated and presented in Table X. As can be seen, the most successful result was obtained with a rate of 99.79% for the accuracy, recall, precision, and F1-score of the NB algorithm when our proposed CNN model was used as the feature extractor. The accuracy rates of 99.47%, 97.87%, and 97.66% were obtained by the SVM algorithm when our proposed CNN model, SqueezeNet, and AlextNet, respectively, were used to extract the features. As a result, the best achievements for all performance measures were obtained when using our proposed CNN model as the feature extractor. When comparing SqueezeNet and AlexNet, we almost achieved better accuracy rates for SVM, RF and NB algorithms using the features extracted from SqueezeNet than those from AlexNet. However, the training and testing times for SqueezeNet-based algorithms were longer due to the larger size of the extracted features. Although the extracted feature size of our proposed CNN model is the smallest, it achieved the best results on all performance measures, as shown in Table X. Therefore, this is an indication that our proposed model is built to learn the key features of the ECG images dataset. Thus, the advantages of the proposed model are not only the better accuracy rates but also the lower computational costs compared to the works in the literature. However, the proposed model could achieve better results if optimization algorithms are used to determine the values of its hyperparameters.

## VI. CONCLUSION

In this article, we propose a lightweight CNN-based model to classify the four major cardiac abnormalities, i.e., AH, MI, H. MI, and NP classes, using public ECG images dataset of cardiac patients. According to the results of the experiments, the proposed CNN model achieves remarkable results in cardiovascular disease classification and can also be used as a feature extraction tool for the traditional machine learning classifiers. Thus, the proposed CNN model can be used as an assistance tool for clinicians in the medical field to detect cardiac diseases from ECG images and bypass the manual process that leads to inaccurate and time-consuming results.

In the future work, optimization techniques can be used to obtain optimized values for the hyperparameters of the proposed CNN model. The proposed model can also be used for predicting other types of problems. Since, the proposed model belongs to the family of low-scale deep learning methods in terms of the number of layers, parameters, and depth. Therefore, a study on using the proposed model in the Industrial Internet of Things domain for classification purposes can be explored.

## REFERENCES

[1] World Health Organization (WHO), "Cardiovascular diseases," Jun. 11, 2021. Accessed: Dec. 27, 2021. [Online]. Available: https://www.who.int/health-topics/cardiovascular-diseases

[2] Government of Westren Australia, Department of Health, "Common medical tests to diagnose heart conditions," Accessed: Dec. 29, 2021. [Online]. Available: https://www.healthywa.wa.gov.au/Articles/A_E/Common-medical-tests-to-diagnose-heart-conditions

[3] M. Swathy and K. Saruladha, "A comparative study of classification and prediction of cardio-vascular diseases (CVD) using machine learning and deep learning techniques," *ICT Exp.*, to be published, 2021. [Online]. Available: https://doi.org/10.1016/j.icte.2021.08.021

[4] R. R. Lopes *et al.*, "Improving electrocardiogram-based detection of rare genetic heart disease using transfer learning: An application to phospholamban p.Arg14del mutation carriers," *Comput. Biol. Med.*, vol. 131, 2021, Art. no. 104262. [Online]. Available: https://doi.org/10.1016/j.compbiomed.2021.104262

[5] R. J. Martis, U. R. Acharya, and H. Adeli, "Current methods in electrocardiogram characterization," *Comput. Biol. Med.*, vol. 48, pp. 133–149, 2014. [Online]. Available: https://doi.org/10.1016/j.compbiomed.2014.02.012

[6] A. Rath, D. Mishra, G. Panda, and S. C. Satapathy, "Heart disease detection using deep learning methods from imbalanced ECG samples," *Biomed. Signal Process. Control*, vol. 68, 2021, Art. no. 102820. [Online]. Available: https://doi.org/10.1016/j.bspc.2021.102820

[7] A. Mincholé and B. Rodriguez, "Artificial intelligence for the electrocardiogram," *Nature Med.*, vol. 25, no. 1, pp. 22–23, 2019. [Online]. Available: https://doi.org/10.1038/s41591-018-0306-1

[8] A. Isin and S. Ozdalili, "Cardiac arrhythmia detection using deep learning," *Procedia Comput. Sci.*, vol. 120, pp. 268–275, 2017. [Online]. Available: https://doi.org/10.1016/j.procs.2017.11.238

[9] H. Bleijendaal *et al.*, "Computer versus cardiologist: Is a machine learning algorithm able to outperform an expert in diagnosing phospholamban (PLN) p.Arg14del mutation on ECG?," *Heart Rhythm*, vol. 18, no. 1, pp. 79–87, 2020. [Online]. Available: https://doi.org/10.1016/j.hrthm.2020.08.021

[10] U. R. Acharya, H. Fujita, O. S. Lih, M. Adam, J. H. Tan, and C. K. Chua, "Automated detection of coronary artery disease using different durations of ECG segments with convolutional neural network," *Knowl.-Based Syst.*, vol. 132, pp. 62–71, 2017. [Online]. Available: https://doi.org/10.1016/j.knosys.2017.06.003

[11] M. Kantardzic, *Data Mining: Concepts, Models, Methods, and Algorithms*, 3rd ed. Hoboken, NJ, USA: Wiley, 2020.

[12] S. García, J. Luengo, and F. Herrera, *Data Preprocessing in Data Mining*, 1st ed. Berlin, Germany: Springer, 2015.

[13] G. Dougherty, *Pattern Recognition and Classification: An Introduction*. Berlin, Germany: Springer, 2013.

[14] A. Subasi, *Practical Machine Learning for Data Analysis Using Python*. Cambridge, MA, USA: Academic, 2020.

[15] J. Soni, U. Ansari, D. Sharma, and S. Soni, "Predictive data mining for medical diagnosis: An overview of heart disease prediction," *Int. J. Comput. Appl.*, vol. 17, no. 8, pp. 43–48, 2011.

[16] K. Dissanayake and M. G. Md Johar, "Comparative study on heart disease prediction using feature selection techniques on classification algorithms," *Appl. Comput. Intell. Soft Comput.*, vol. 2021, 2021, Art. no. 5581806. [Online]. Available: https://doi.org/10.1155/2021/5581806

[17] A. H. Gonsalves, F. Thabtah, R. M. A. Mohammad, and G. Singh, "Prediction of coronary heart disease using machine learning: An experimental analysis," in *Proc. 3rd Int. Conf. Deep Learn. Technol.*, 2019, pp. 51–56. [Online]. Available: https://doi.org/10.1145/3342999.3343015

[18] H. Kim, M. I. M. Ishag, M. Piao, T. Kwon, and K. H. Ryu, "A data mining approach for cardiovascular disease diagnosis using heart rate variability and images of carotid arteries," *Symmetry*, vol. 8, no. 6, 2016, Art. no. 47. [Online]. Available: https://doi.org/10.3390/sym8060047

[19] T. Ozcan, "A new composite approach for COVID-19 detection in X-ray images," *Appl. Soft Comput.*, vol. 111, 2021, Art. no. 107669. [Online]. Available: https://doi.org/10.1016/j.asoc.2021.107669

[20] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and $<$ 0.5 MB model size," 2016, *arXiv:1602.07360*.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Adv. Neural Inf. Process. Syst.*, vol. 25, pp. 1097–1105, 2012.

[22] A. H. Khan, M. Hussain, and M. K. Malik, "Cardiac disorder classification by electrocardiogram sensing using deep neural network," *Complexity*, vol. 2021, 2021, Art. no. 5512243. [Online]. Available: https://doi.org/10.1155/2021/5512243

[23] A. H. Khan and M. Hussain, "ECG images dataset of cardiac patients," *Mendeley Data*, vol. V2, 2021. [Online]. Available: https://doi.org/10.17632/gwbz3fsgp8.2

[24] C. Potes, P. Saman, A. Rahman, and B. Conroy, "Ensemble of feature-based and deep learning-based classifiers for detection of abnormal heart sounds," in *Proc. Comput. Cardiol. Conf.*, 2016, pp. 621–624.

[25] A. Nannavecchia, F. Girardi, P. R. Fina, M. Scalera, and G. Dimauro, "Personal heart health monitoring based on 1D convolutional neural network," *J. Imag.*, vol. 7, no. 2, 2021, Art. no. 26. [Online]. Available: https://doi.org/10.3390/jimaging7020026

[26] Q. Zhang, D. Zhou, and X. Zeng, "HeartID: A multiresolution convolutional neural network for ECG-based biometric human identification in smart health applications," *IEEE Access*, vol. 5, pp. 11805–11816, 2017. [Online]. Available: https://doi.org/10.1109/ACCESS.2017.2707460

[27] U. R. Acharya, S. L. Oh, Y. Hagiwara, J. H. Tan, M. Adam, and R. S. Tan, "A deep convolutional neural network model to classify heartbeats," *Comput. Biol. Med.*, vol. 89, pp. 389–396, 2017. [Online]. Available: https://doi.org/10.1016/j.compbiomed.2017.08.022

[28] R. Bharti, A. Khamparia, M. Shabaz, G. Dhiman, S. Pande, and P. Singh, "Prediction of heart disease using a combination of machine learning and deep learning," *Comput. Intell. Neurosci.*, vol. 2021, 2021, Art. no. 8387680. [Online]. Available: https://doi.org/10.1155/2021/8387680

[29] P. Bizopoulos and D. Koutsouris, "Deep learning in cardiology," *IEEE Rev. Biomed. Eng.*, vol. 12, pp. 168–193, 2018. [Online]. Available: https://doi.org/10.1109/RBME.2018.2885714

[30] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ECG classification by 1-D convolutional neural networks," *IEEE Trans. Biomed. Eng.*, vol. 63, no. 3, pp. 664–675, Mar. 2016. [Online]. Available: https://doi.org/10.1109/TBME.2015.2468589

[31] M. Zubair, J. Kim, and C. Yoon, "An automated ECG beat classification system using convolutional neural networks," in *Proc. 6th Int. Conf. IT Convergence Secur.*, 2016, pp. 1–5. [Online]. Available: https://doi.org/10.1109/ICITCS.2016.7740310

[32] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4510–4520.

[33] T. Rahman *et al.*, "COV-ECGNET: COVID-19 detection using ECG trace images with deep convolutional neural network," 2021, *arXiv:2106.00436*.

[34] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2261–2269.

[35] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2818–2826.

[36] A. Pal, R. Srivastva, and Y. N. Singh, "CardioNet: An efficient ECG arrhythmia classification system using transfer learning," *Big Data Res.*, vol. 26, 2021, Art. no. 100271. [Online]. Available: https://doi.org/10.1016/j.bdr.2021.100271

[37] R. Avanzato and F. Beritelli, "Automatic ECG diagnosis using convolutional neural network," *Electronics*, vol. 9, no. 6, 2020, Art. no. 951. [Online]. Available: https://doi.org/10.3390/electronics9060951

[38] U. R. Acharya, H. Fujita, S. L. Oh, Y. Hagiwara, J. H. Tan, and M. Adam, "Application of deep convolutional neural network for automated detection of myocardial infarction using ECG signals," *Inf. Sci.*, vol. 415–416, pp. 190–198, 2017. [Online]. Available: https://doi.org/10.1016/j.ins.2017.06.027

[39] M. Naz, J. H. Shah, M. A. Khan, M. Sharif, M. Raza, and R. Damaševičius, "From ECG signals to images: A transformation based approach for deep learning," *PeerJ Comput. Sci.*, vol. 7, 2021, Art. no. e386, doi: 10.7717/peerj-cs.386.

[40] H. El-Amir and M. Hamdy, *Deep Learning Pipeline: Building a Deep Learning Model With TensorFlow*. New York, NY, USA: Apress Media, 2020.

[41] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intell. Syst. Appl.*, vol. 13, no. 4, pp. 18–28, Apr. 1998. [Online]. Available: https://doi.org/10.1109/5254.708428

[42] M. Abubaker and W. M. Ashour, "Efficient data clustering algorithms: Improvements over Kmeans," *Int. J. Intell. Syst. Appl.*, vol. 5, no. 3, pp. 37–49, 2013. [Online]. Available: https://doi.org/10.5815/ijisa.2013.03.04

[43] B. Charbuty and A. Abdulazeez, "Classification based on decision tree algorithm for machine learning," *J. Appl. Sci. Technol. Trends*, vol. 2, no. 1, pp. 20–28, 2021.

[44] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[45] E. Miranda, E. Irwansyah, A. Y. Amelga, M. M. Maribondang, and M. Salim, "Detection of cardiovascular disease risk's level for adults using naive Bayes classifier," *Healthcare Inform. Res.*, vol. 22, no. 3, pp. 196–205, 2016. [Online]. Available: https://doi.org/10.4258/hir.2016.22.3.196

[46] G. Masetti and F. D. Giandomenico, "Analyzing forward robustness of feedforward deep neural networks with LeakyReLU activation function through symbolic propagation," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, 2020, pp. 460–474.

[47] S. Shahinfar, P. Meek, and G. Falzon, "How many images do I need?" Understanding how sample size per class affects deep learning model performance metrics for balanced designs in autonomous wildlife monitoring," *Ecological Inform.*, vol. 57, 2020, Art. no. 101085. [Online]. Available: https://doi.org/10.1016/j.ecoinf.2020.101085

[48] B. Zoph, E. D. Cubuk, G. Ghiasi, T. Lin, J. Shlens, and Q. V. Le, "Learning data augmentation strategies for object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 566–583.