IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received May 13, 2020, accepted June 3, 2020, date of publication June 12, 2020, date of current version June 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3002151

Transfer Learning Benchmark for Cardiovascular Disease Recognition

MEHREZ BOULARES[1], TARIK ALAFIF[2], AND AHMED BARNAWI[1]
[1]Faculty of Computing and IT, King Abdulaziz University, Jeddah 21589, Saudi Arabia
[2]Department of Computer Science, Jamoum University College, Umm Al-Qura University, Jamoum 25375, Saudi Arabia

Corresponding author: Mehrez Boulares (mboulares@kau.edu.sa)

This work was supported by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under Grant RG-23-611-38.

ABSTRACT The cardiac auscultation using the classical stethoscope (PCG: phonological cardiogram) is known as the most famous method to detect Cardiovascular Disease (CVD). However, this exam requires a qualified cardiologist which relies on the cardiac cycle vibration sound (heart muscle contractions and valves closure) to detect abnormalities in heart during the pumping action. Many research works have been conducted for detecting CVD from PCG signals by using public and private datasets. Due to the lack of CVD recognition benchmark, classification results are very heterogeneous and can not be compared objectively. In this paper, we apply transfer learning to Pascal public dataset and provide an experimental benchmark without any denoising or cleaning steps. The main goal is to generate a set of experimental results which can be used as starting reference for future CVD recognition research based on PCG.

INDEX TERMS Cardiovascular disease recognition, convolutional neural network, pre-trained model, deep learning, transfer learning, benchmark.

I. INTRODUCTION

Heart is the most crucial part in human body since it is responsible for pushing and flowing blood cycles to all other parts in the body. Human heart rates generate Phono-cardiogram (PCG) signals. These signals are measured by mechanical vibrations coming from a stethoscope device being put on the chest. Then, the signals are collected and recorded to watch human heart normality and abnormality conditions. Major abnormalities of heart rates, the so called cardiovascular diseases (CVDs), can be the main cause for human deaths. An estimated 17.9 million people died from CVDs in 2016, representing 31% of all global deaths [1]. Therefore, it is very critical to classify PCG signals to automate the recognition for the heart rates normalities and abnormalities accurately in real-time to save lives against CVDs. Figure 1 shows an example for normal and abnormal PCG signals which represent human heart rates. One can notice that the normal PCG signal includes more noise compared to the abnormal one.

In fact, many research have been conducted in order to classify automatically cardiovascular disease from phono-cardiogram signal. The majority of existing works are focused on the improvement of the classification results based on public or private PCG datasets. The main ascertainment is that in the majority of proposed solutions, researchers relayed on their own private dataset or a modified version of existing public dataset such as Pascal. Consequently, the obtained classification results cannot be compared accurately. To the best of our knowledge, there is no common benchmark acting as a starting reference for PCG classification. For this reason, in this paper, we propose a new PCG classification benchmark based on raw public PCG dataset (Pascal).

Transfer learning technique is widely used for several classification problems. It is considered as a powerful recognition

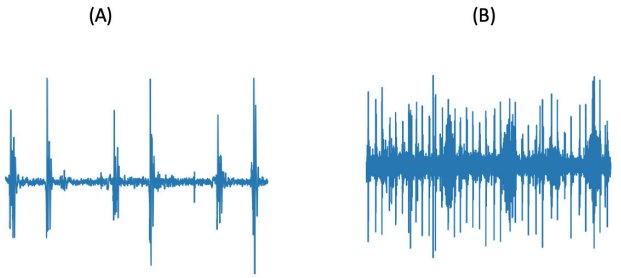

FIGURE 1. An example of PCG signals for human heart rates. (A) Normal (B) Abnormal.

The associate editor coordinating the review of this manuscript and approving it for publication was Chao Zuo.

method especially when it comes to small-scale training dataset. In our context, we focus on transfer learning technique through CNN fine-tuning. We used many pre-trained Convolutional Neural Network (CNN) models which are retrained on the publicly available Pascal dataset which is small size dataset containing three classes labeled PCG signals (normal, murmur, extrasystole).

The focus of this study is based on Pascal dataset, which is to the best of our knowledge the only widely used and publicly available multi-class labeled PCG dataset while other dataset such as PyhsioNet is considered a binary labeled dataset (i.e. normal or abnormal). Thus, we believe that benchmarking the Pascal dataset is probably more significant than considering the binary labeled dataset since it is widely used by scientific community to study and enhance the classifications of multiple classes for Cardiovascular Diseases (CVD). Therefore, the contribution of this work focuses on a multi-class CVD classification and benchmark of Pascal dataset which is proven to be a good starting option for researchers in the field to test their methods. We believe that our work is very useful to set a reference mark toward their experimental comparisons.

Our main goal is to propose clear classification results acting as a reference benchmark for future CVD recognition research. The remainder of this paper is organized as follows. In the remaining of **Section I**, related work is reviewed. In **Section II**, we discuss the used dataset. In **Section III**, we present our benchmark process. In **Section IV**, the experimental results and evaluation are provided. Finally, we conclude our work in **Section V**.

A. RELATED WORKS

Many CVD recognition methods have been proposed in the past decades. Potes *et al.* [34] only used a training set A, which contains a total of 400 heart sound records from Physionet [31]. They proposed a binary classifier (normal/abnormal) starting from Mel-frequency cepstral coefficients using Support Vector Machines with a radial kernel. Dave [12] presented a CVD classification solution (Normal heart sound) based on the localization of S1 and S2 (Heart cycle). Then, K-means and DNN are applied on a private dataset.

Lawrence *et al.* [24] aimed to classify normal and abnormal PCG singal based on Pre-emphasis filtering and windowing using a private dataset. Sandler *et al.* [38] proposed a normal/abnormal classification method based on a Modified Linear predictive Coding using a special PCG dataset (3M poland composed only from 72 signals). Szegedy *et al.* [43] proposed a binary classification solution (normal/murmur) based on cepstrum coefficients and machine learning techniques (Bayes Net, logit boost). They relayed on a modified (balanced) version of publicly available dataset Pascal [8], but only 30 normal and 30 murmur examples are used. Balili *et al.* [5] were focused on multi class (Normal/aortic insufficiency, aortic stenosis, atrial spetal defect, mitral regulation and mitral stenosis) recognition based on Low pass filtering, shannon energy envelop calculation, wavelet decomposition, and SVM classifier. The authors experimented their approach using a private dataset composed from 64 signals (Data recorded at Maulana Azad Medical Institute in India).

Chollet [11] proposed a multi class (Normal/snaps/murmur/clicks) recognition solution based on MFCC and Hidden Markov Model using 41 samples from a private dataset. Nogueira *et al.* [32] proposed a PCG classification (Normal/Murmur/Extrasystole) solution based on heart cycle localization using a modified version of the training set A in Pascal dataset (Unlabeled signals in the dataset A were labeled by domain of experts). Similarly, deep learning has been tightened with only few work in this area. Sidra *et al.* [39] converted segments of time series from PhysioNet dataset into heatmaps. A CNN is designed and trained from scratch to classify normal and abnormal heart rates using the heatmaps. Redlarski *et al.* [36] used Adaboost and CNN classifiers to classify normal and abnormal heart sounds from PCG recordings. Dave [12] applied k-means and deep neural networks to classify normal and abnormal heart rates using their own private dataset. Zhang *et al.* [48] used artificial neural network to classify normal and abnormal heart sound recordings [48]. The aforementioned methods used private datasets.

From the above literature review, we can conclude that the majority of related works relies on either on a modified version of publicly available PCG dataset or on their own privates dataset. In other words, the obtained classification results cannot be compared objectively. For this reason, we propose a detailed benchmark process based on pre-trained CNN models. Since a transfer learning does not require a large number of training examples, we apply transfer learning to classify PCG recordings for both two classes (normal/abnormal) and three classes (normal/murmur/extrasystole) using Pascal dataset without any data modification and pre-processing step.

II. PASCAL DATASET

In this section, first we describe the unbalanced Pascal raw dataset. Then, the balanced Pascal dataset is described.

A. UNBALANCED (RAW) DATASET

The publicly available Pascal dataset [8] is popular for CVD recognition research. The Pascal dataset consists of dataset A and dataset B. Dataset A is collected publicly from iStethscope Pro iPhone app which contains 176 files in WAV format. The dataset provides four classes for heart audio. Dataset B is collected from a clinical setting using a digital stethoscope DigiScope. It is larger than dataset A which contains 656 files in WAV format. Dataset B is used for our experiment to classify normal and abnormal PCG images since it contains more images and provides three classes for heart audio. Extrasystole and murmur PCG images are considered abnormal.

TABLE 1. Unbalanced examples distribution for three classes in Pascal dataset.

Model	Class		
	Normal	Murmur	Extrasystole
A	31	34	19
B	200	95	46
Total	**231**	**129**	**65**

TABLE 2. Unbalanced examples distribution for two classes in Pascal dataset.

Model	Class	
	normal	abnormal
A	31	53
B	200	141
Total	**231**	**194**

TABLE 3. Balanced examples distribution for two classes in Pascal dataset.

Training set	Normal	Abnormal
A	31	53
B	320	141
Total	**351**	**194**
Total balanced	**194**	**194**

TABLE 4. Keras pre-trained CNN models.

Model	Layers	Size	Parameters	Reference
Xception	71	85 MB	44.6 millions	[13]
VGG19	26	549 MB	143.6 millions	[42]
VGG16	23	528 MB	138.3 millions	[42]
ResNet-152-v2	-	98 MB	25.6 millions	[19]
ResNet-152	-	232 MB	60.4 millions	[19]
ResNet-101-v2	-	171 MB	44.6 millions	[19]
ResNet-101	101	167 MB	44.6 millions	[19]
ResNet-50-v2		98 MB	25.6 millions	[19]
ResNet-50	-	98 MB	25.6 millions	[19]
NASNetMobile	-	20 MB	5.3 millions	[52]
NASNetLarge	-	343 MB	88.9 millions	[52]
MobileNet-v2	53	13 MB	3.5 millions	[40]
MobileNet	88	16 MB	4.25 millions	[20]
Inception-v3	48	89 MB	23.9 millions	[46]
InceptionResNet-v2	164	209 MB	55.9 millions	[44]
DenseNet-201	201	77 MB	20 millions	[22]
DenseNet-169	169	57 MB	14.3 millions	[22]
DenseNet-121	121	33 MB	8.06 millions	[22]

In our work, we use the Pascal dataset. Tables 1 and 2 summarize the structure and example distribution in this dataset. Concerning normal class samples, we use 231 samples obtained by merging the normal samples from training set A and training set B without considering the samples in Btraining_noisynormal. For murmur class samples, we relay on 129 samples. The samples are obtained from merging 34 samples from training set A with 95 samples from merging 66 samples from training set B and 29 samples form Btraining_noisymurmur. Considering extrasystole class, we use 65 samples from merging 19 samples from training set A and 46 samples from training set B. In order to obtain two classes dataset, we use the same pascal dataset by merging Murmur and extrasystole samples into one class named abnormal. We obtained 231 samples for normal class and 194 for abnormal class.

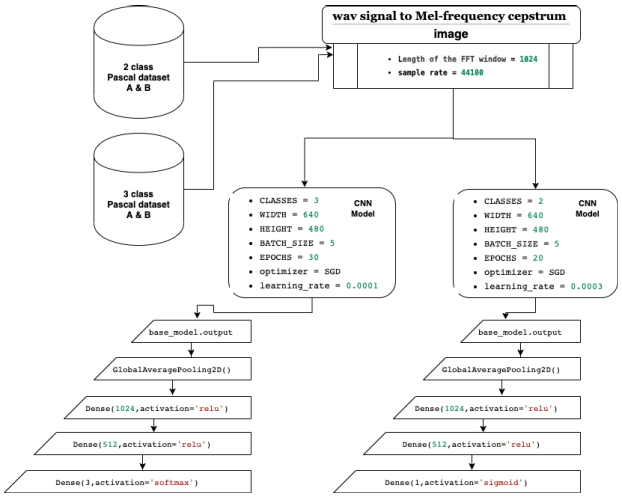

FIGURE 2. The system architecture of our approach using Keras and Matlab platform.

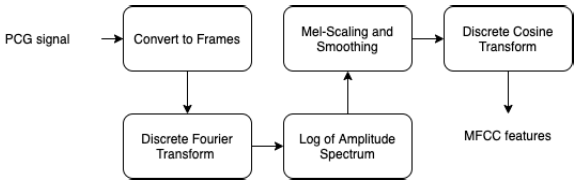

FIGURE 3. MFCC steps.

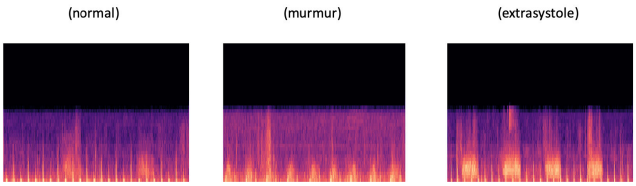

FIGURE 4. An overview of PCG MFCC outputs for Normal, Murmur, and Extrasystole classes respectively from left to right.

TABLE 5. Matlab pre-trained CNN models.

Model	Layers	Size	Parameters	Reference
AlexNet	8	227 MB	61 millions	[25]
SqueezeNet	18	4.6 MB	1.24 millions	[23]
GoogLeNet	22	27 MB	7 millions	[45]
Inception-v3	48	89 MB	23.9 millions	[46]
DenseNet-201	201	77 MB	20 millions	[22]
MobileNet-v2	53	13 MB	3.5 millions	[40]
ResNet-101	101	167 MB	44.6 millions	[19]
Xception	71	85 MB	44.6 millions	[13]
InceptionResNet-v2	164	209 MB	55.9 millions	[44]
ShuffleNet	50	6.3 MB	1.4 millions	[51]
NASNetMobile	-	20 MB	5.3 millions	[52]

B. BALANCED DATASET

For balancing the training examples in Pascal dataset, We merge training set A normal samples with training set B normal samples and Btraining_noisynormal which includes 120 samples. We obtain a total of 351 samples from normal

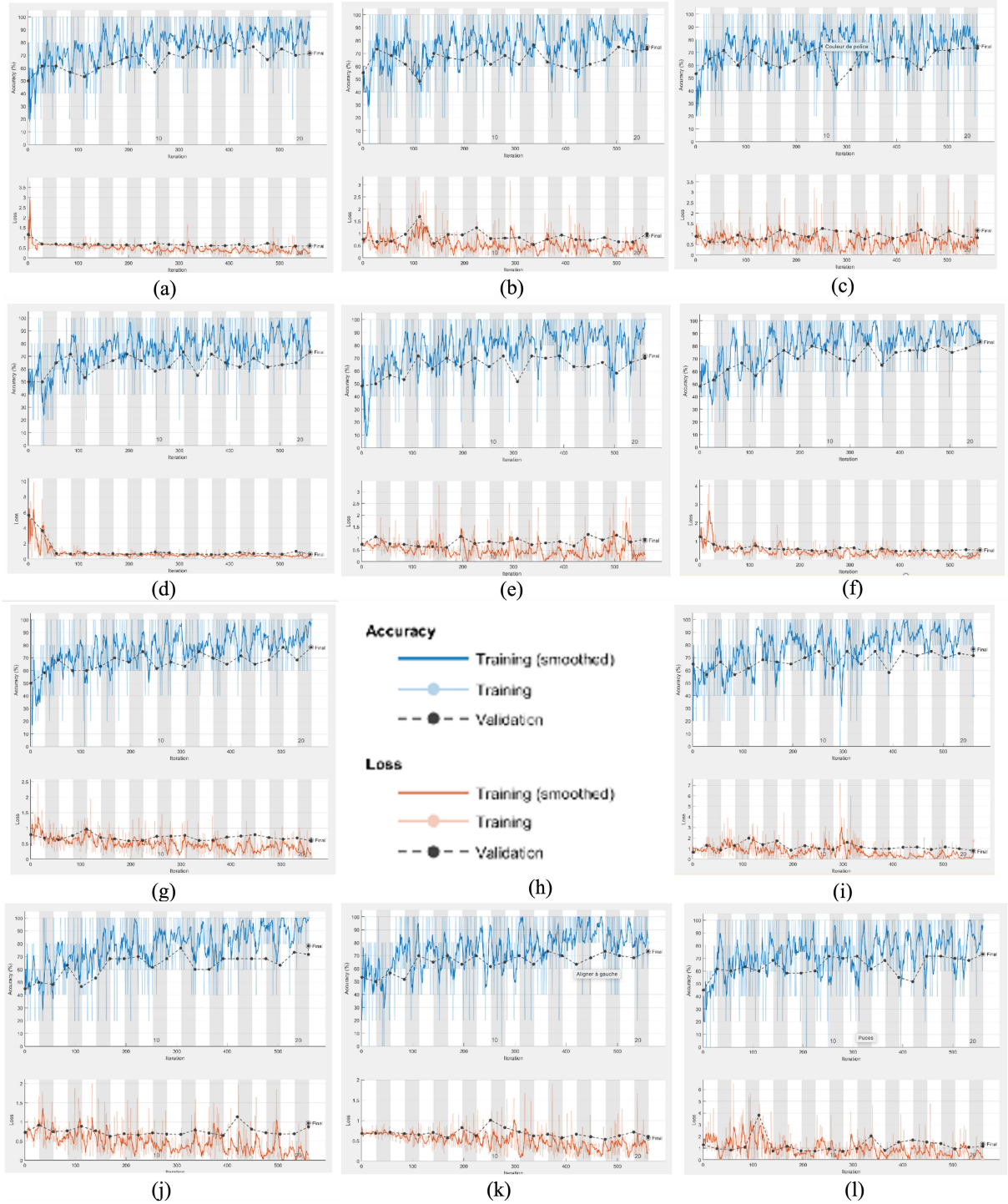

FIGURE 5. The training validations and the loss curves using the Matlab pre-trained CNN models. (a) AlexNet. (b) SqueezNet. (c) GoogLeNet. (d) Inception-v3. (e) DenseNet201. (f) MobileNet-v2. (g) ResNet101. (h) Figure legends. (i) Xception. (j) InceptionResNet-v2. (k) ShuffleNet. (l) NASNetMobile.

class as shown in Table 3. Concerning abnormal class, we also merge 53 samples (murmur = 34 with extrasystole = 19) from training set A and 141 samples (murmur = 95 with extrasystole = 46) from training set B. In order to obtain a balanced (equal class samples) dataset, we chose randomly the first 194 samples from normal class and all the 194 samples from abnormal class.

III. CNN BENCHMARK PROCESS

The main objective in this work is to create a CNN benchmark for CVD recognition using Pascal dataset. Our approach is designed to use several pre-trained CNN models to recognize CVD by learning edges and corners from PCG image representation. The complex signal features representation for normal and abnormal heart cycles are learned to handle

the recognition for signals variations. As shown in Figure 2, starting from the PCG WAV signal classification using Pascal dataset, the first processing step aims to transform WAV signal into Mel-Frequency Cepstrum represented by a PNG image. In the next step, we apply transfer learning and fine-tune the pre-trained CNN models with the training examples. This choice is due to a small number of examples found in Pascal dataset.

A. PRE-PROCESSING

Mel Frequency Cepstrum Coefficients (MFCC) have been widely used in speech recognition field [2], [14], [18]. This success has been to their ability to represent the speech amplitude spectrum in a compact form. As shown in Figure 3, the processing steps of MFCC starts by dividing the PCG signal into frames usually by applying Hamming windowing function at fixed interval (1024 in our case) in order to remove edge effects. This step gives us a cepstral feature vector for each frame. Then, Discrete Fourier Transform related to each frame is computed by retaining only the logarithm of the amplitude spectrum relative to each frame. The reason for applying logarithmic operation is due to loudness property of the signal that has been found to be approximately logarithmic. Then, spectrum is smoothed in order to obtain meaningful frequencies. Finally, Discrete Cosine Transform is applied for the aim to obtain MFCC features.

In our work, we adopted MFCC signal representation by transforming the output feature into an image which is used as an input to our pre-trained CNN models. Figure 4 gives an overview of PCG MFCC outputs for normal, murmur, and extrasystole classes respectively.

B. PRE-TRAINED CNN MODELS

Deep learning has become an important research area in computer vision, many deep learning models have been proposed in many recognition tasks as seen in [3], [6], [21], [28]. Our work is particularly motivated by Convolutional Neural Network (CNN) deep architectures which tolerates image distortion, illumination changes, and provide invariance of image translation. High level features are extracted and learned from images.

The first CNN architecture was introduced by LeCun [27]. CNN has been a remarkable success in many computer vision tasks such as face detection [4], [16], [17], [29], [49], handwritten recognition [27], face recognition [3], [26], and image classification [25] from a large number of training image examples. With the emerging of transfer learning, or the so called knowledge transfer, as a new learning framework [47], the same results can be achieved on deep learning applications by fine-tuning the existing pre-trained CNN models that have been already trained on ImageNet. These models require small number of training examples compared to designed ones in which they require an effort to collect a large number of training examples to train. In this case, transfer learning is suitable and fits for training PCG recordings since a small of training examples is only publicly available.

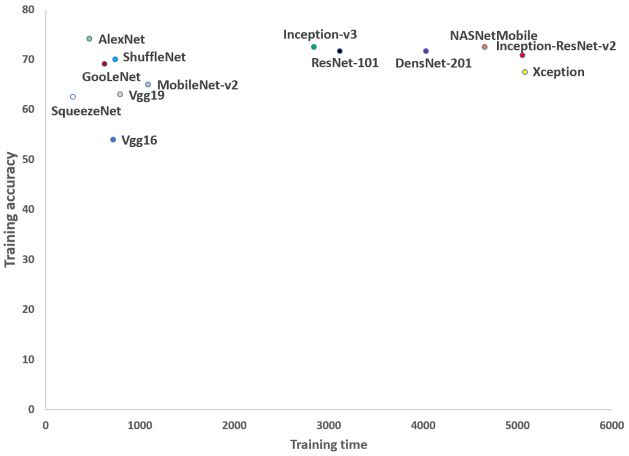

FIGURE 6. An overview of the average training time verses training accuracy for the Matlab pre-trained models using 2 classes and 2 folds.

TABLE 6. Training accuracy of CNN models using 2 class and 2 folds.

Model	Fold1		Fold2		Avg
	Acc	Time	Acc	Time	
Alexnet	0.73	7':13"	0.75	8':18"	0.74
Vgg16	0.55	10':31"	0.53	13':14"	0.54
Vgg19	0.733	12':65"	0.53	13':14 "	0.63
Squeezenet	0.71	4':50"	0.53	4':52"	0.62
Googlenet	0.83	10':13"	0.55	10':40"	0.69
Inceptionv3	0.71	36':47"	0.73	58':0"	0.72
Densenet201	0.75	69':4"	0.68	65':19"	0.71
Mobilenetv2	0.75	18':9"	0.55	18':6"	0.65
Resnet101	0.766	51':38"	0.66	52':19"	0.71
Xception	0.73	84':47"	0.61	84':29"	0.67
Inceptionresnetv2	0.78	85':14"	0.63	83':14"	0.70
Shufflenet	0.73	15':11"	0.66	9':26"	0.70
Nasnetmobile	0.78	94':7"	0.66	60':59"	0.72
Nasnetlarge	-	-	-	-	memory out

TABLE 7. Validation accuracy of CNN models using 2 classes and 2 folds.

Model	Fold1	Fold2	Avg
	Acc	Acc	
AlexNet	0.840	0.850	0.845
VGG16	Out of memory	Out of memory	---
VGG19	Out of memory	Out of memory	---
SqueezeNet	0.785	0.710	0.7475
GoogLeNet	0.900	0.810	0.855
Inception-v3	0.7350	0.8750	0.805
DenseNet201	0.8850	0.890	0.8875
MobileNetv2	0.7750	0.870	0.8225
ResNet101	0.870	0.870	0.870
Xception	0.830	0.890	0.860
InceptionResNet-v2	0.8650	0.9250	**0.8950**
ShuffleNet	0.890	0.8550	0.8725
NasNetMobile	0.870	0.8850	0.8775
NasNetLarge	-	-	memory out

In this work, we plug deep learning in the medical domain with the use of time series data and by applying transfer learning to classify CVD. Several deep pre-trained convolutional Neural networks are fine-tuned and trained. In our approach, the CNN input has the dimension size of 640×480. By fine-tuning the models, we preserve convolutional layers used for feature extraction. Then, we add additional layers. The

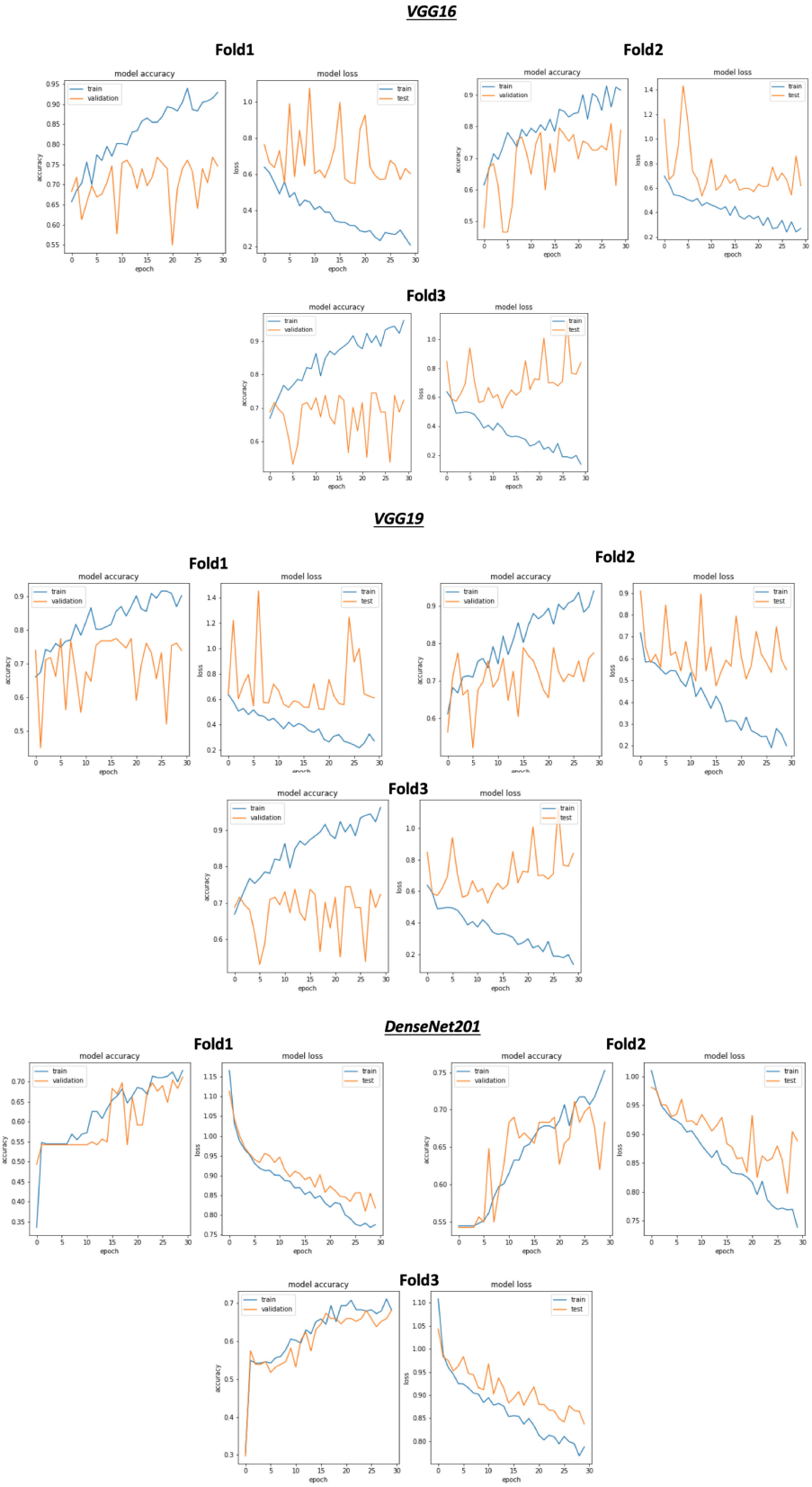

FIGURE 7. The training validations and the loss curves after training the Keras pre-trained CNN models (VGG16, VGG19, and DenseNet201) using 2 classes and 3 folds.

TABLE 8. Validation true positive rate (TPR) of CNN models using 2 classes and 3 folds.

Model	3 Fold									
	Fold1			Fold2			Fold3			AVG
	class 1	class 2	avg	class 1	class 2	avg	class 1	class 2	avg	
VGG16	0.64	0.87	0.75	0.8	0.81	0.80	0.54	0.90	0.72	**0.76**
VGG19	0.66	0.87	0.76	0.81	0.76	0.79	0.59	0.90	0.75	**0.76**
Xception	0.61	0.80	0.71	0.63	0.80	0.71	0.78	0.55	0.66	0.69
ResNet152V2	0.58	0.84	0.71	0.73	0.67	0.70	0.51	0.85	0.68	0.70
ResNet152	0.78	0.67	0.72	0.66	0.83	0.74	0.56	0.85	0.70	0.72
ResNet101V2	0.46	0.89	0.67	0.70	0.75	0.73	0.59	0.80	0.69	0.70
ResNet101	0.49	0.93	0.71	0.75	0.75	0.75	0.56	0.89	0.72	**0.73**
ResNet50v2	0.63	0.83	0.73	0.64	0.84	0.74	0.56	0.87	0.71	0.73
ResNet50	0.52	0.89	0.70	0.70	0.79	0.74	0.59	0.84	0.71	0.72
NasNetMobile	0.70	0.67	0.69	0.61	0.80	0.71	0.76	0.61	0.68	0.69
NasNetLarge	0.58	0.70	0.64	0.70	0.76	0.73	0.59	0.67	0.63	0.67
MobileNetV2	0.64	0.77	0.71	0.55	0.89	0.72	0.64	0.71	0.67	0.70
MobileNet	0.61	0.90	0.76	0.72	0.71	0.71	0.53	0.88	0.70	0.72
Inceptionv3	0.6	0.81	0.70	0.70	0.76	0.73	0.56	0.84	0.70	0.71
InceptionResNetV2	0.46	0.94	0.70	0.66	0.76	0.71	0.62	0.64	0.63	0.68
DenseNet201	0.67	0.83	0.75	0.58	0.93	0.75	0.65	0.77	0.71	0.73
DenseNet169	0.61	0.88	0.74	0.66	0.80	0.73	0.57	0.83	0.70	0.72
DenseNet121	0.61	0.81	0.71	0.78	0.74	0.76	0.60	0.87	0.73	0.73

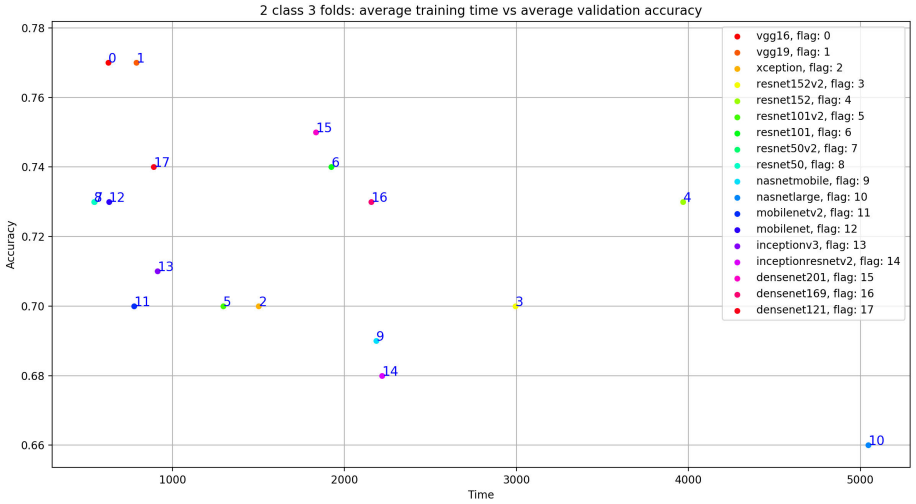

FIGURE 8. An overview of training time verses validation accuracy using the Keras pre-trained models for 2 classes and 3 folds.

first added layer GlobalAveragePooling2D is used for better representation for our feature vector. It uses a parser window moving across the feature matrix and pools the data by averaging it. The second and third added layers are dense layers respectively 1024 and 512 to allow learning more complex functions and to obtain better classification results. Finally, the fourth added layer, characterizes the classification layer through dense layer with softmax activation function. In the CNN training process, we use stochastic gradient descent optimizer for weight update. We use 0.0001 learning rate and Keras default momentum. Batch size is set to 5 while epochs are set to 30. In our experiments, we chose to use both Keras and Matlab pre-trained CNN models as shown in Tables 4 and 5 respectively.

Keras pre-trained CNN models are trained using the unbalanced Pascal dataset for two and three classes through Google Colab plateform allowing the use of a dedicated GPU. The GPU is 1xTesla K80, having 2496 CUDA cores, compute 3.7, 12GB (11.439GB Usable) GDDR5 VRAM. In Matlab platform, pre-trained CNN models are trained using Matlab Pretrained Deep Neural Networks [33]. A balanced pascal dataset for two classes is used. A Huawei laptop Intel(R) Core(TM) i7 1.99 GHz CPU with 16 GB of RAM is used to fine-tune and test the models.

IV. EXPERIMENTS AND RESULTS

In this section, we experiment the pre-trained CNN models using PCG Pascal dataset. As discussed in pre-trained CNN models section, we evaluate Keras models for the unbalanced two classes (normal/abnormal) as well as for the three classes (normal/murmur/extrasystole) dataset. The Matlab

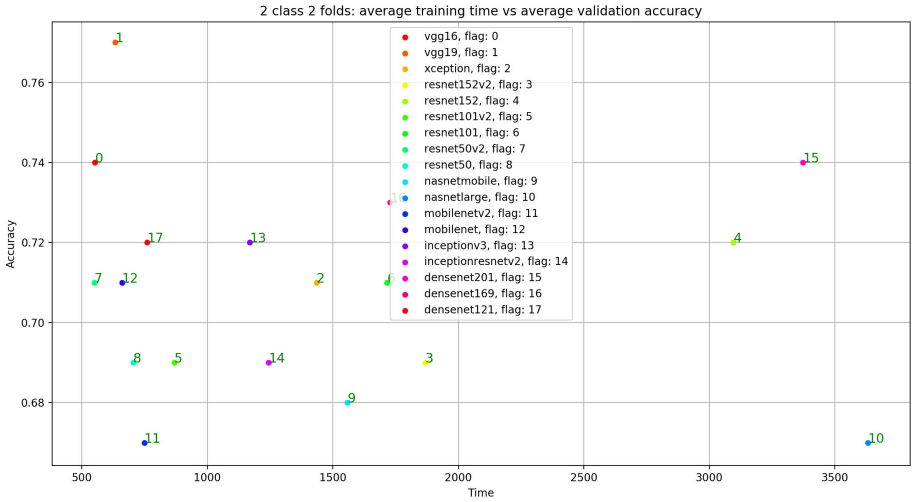

FIGURE 9. An overview of training time verses validation accuracy using Keras pre-trained models for 2 classes and 2 folds.

pre-trained models are also evaluated for the balanced two classes (normal/abnormal) dataset.

A. CLASSIFICATION USING BALANCED DATASET

In this subsection, we first explain the balanced Pascal dataset classification. Second, the unbalanced Pascal dataset classification is explained.

1) TWO CLASSES (NORMAL/ABNORMAL) CLASSIFICATION

We fine-tuned eleven pre-trained CNN models by using Matlab Pre-trained Neural Networks [33] on the balanced pascal dataset. We apply 2-cross validation to 388 samples. We use 194 samples merged from murmur and extrasystole for the abnormal class. On the other hand, we use 194 normal samples for the normal class. The number of epochs is set to 20. The batch size is set to 5. We use 0.0003 Learning rate. Figure 5 gives an overview of training validations and their loss curves using the Matlab pre-trained CNN models. As shown in Table 6 and Table 7, InceptionResNet-v2 model achieves the best average test accuracy with 89.5% compared to other models. As shown in Figure 6, the fastest training time and worst training accuracy average is obtained by using SqueezeNet. One can notice that AlexNet holds the best average training accuracy with 84.5%.

B. CLASSIFICATION USING UNBALANCED DATASET

In this subsection we evaluate both two classes (normal-abnormal) and three classes (normal-murmur-extrasystole) for PCG recognition by using Keras pre-trained CNN models. The result in this step is to generate a PCG recognition benchmark using pre-trained CNN models.

1) TWO CLASSES (NORMAL/ABNORMAL) CLASSIFICATION

After applying transfer learning on two classes using Pascal dataset, we obtain the following classification results:

TABLE 9. Validation accuracy for Keras pre-trained models using 2 classes and 3 folds.

Model	3 Fold			
	Fold1	Fold2	Fold3	AVG
VGG16	0.76761	0.80986	0.74468	**0.77**
VGG19	0.77465	0.78873	0.76596	**0.77**
Xception	0.71831	0.72535	0.65957	0.70
ResNet152V2	0.72535	0.70423	0.70213	0.70
ResNet152	0.72535	0.75352	0.7234	0.73
ResNet101V2	0.69718	0.73239	0.70922	0.70
ResNet101	0.73239	0.75352	0.74468	0.74
ResNet50v2	0.73944	0.75352	0.7305	0.73
ResNet50	0.72535	0.75352	0.7305	0.73
NasNetMobile	0.69014	0.71831	0.68085	0.69
NasNetLarge	0.64789	0.73944	0.6383	0.66
MobileNetV2	0.71831	0.73944	0.68085	0.70
MobileNet	0.77465	0.71831	0.7234	0.73
Inceptionv3	0.71831	0.73944	0.71631	0.71
InceptionResNetV2	0.72535	0.71831	0.6383	0.68
DenseNet201	0.76056	0.77465	0.7234	0.75
DenseNet169	0.76056	0.73944	0.71631	0.73
DenseNet121	0.72535	0.76056	0.75177	0.74

(a) **Using three folds:**
We splitted the dataset into 66% for training and 34% for testing. Where 128 samples for abnormal class and 231 samples for normal class are used for training and 66 abnormal and 120 normal are used for testing. We obtain the following results:

- **Fold1:** As shown in Figure 7, the VGG16 pre-trained model reaches the highest training accuracy and highest test accuracy 94% and 76% respectively compared to other models. As shown in Table 9, by using VGG16 models or VGG19, we obtain 77% as the highest accuracy average compared to all evaluated Keras CNN models. In other words, the extended depth for VGG19 compared to the depth for VGG16 has no impact on the validation accuracy

TABLE 10. Validation true positive rate (TPR) of CNN models using 2 classes and 2 folds.

Model	2 Fold						
	Fold1			Fold2			AVG
	class 1	class 2	avg	class 1	class 2	avg	
VGG16	0.59	0.87	0.73	0.67	0.80	0.74	**0.73**
VGG19	0.63	0.90	0.76	0.68	0.82	0.75	**0.75**
Xception	0.64	0.76	0.70	0.75	0.71	0.73	0.71
ResNet152V2	0.65	0.74	0.69	0.70	0.66	0.68	0.68
ResNet152	0.62	0.82	0.72	0.64	0.77	0.70	0.71
ResNet101V2	0.66	0.73	0.69	0.61	0.76	0.68	0.68
ResNet101	0.65	0.75	0.70	0.57	0.82	0.69	0.69
ResNet50v2	0.59	0.82	0.70	0.58	0.82	0.70	0.70
ResNet50	0.55	0.82	0.68	0.55	0.81	0.68	0.68
NasNetMobile	0.66	0.73	0.69	0.70	0.65	0.68	0.68
NasNetLarge	0.56	0.82	0.69	0.75	0.56	0.65	0.67
MobileNetV2	0.5	0.85	0.67	0.77	0.58	0.67	0.67
MobileNet	0.48	0.87	0.68	0.65	0.8	0.72	0.70
Inceptionv3	0.62	0.82	0.72	0.66	0.76	0.71	0.71
InceptionResNetV2	0.65	0.74	0.69	0.70	0.66	0.68	0.68
DenseNet201	0.52	0.93	0.73	0.56	0.91	0.73	0.73
DenseNet169	0.58	0.81	0.70	0.67	0.78	0.72	0.71
DenseNet121	0.63	0.79	0.71	0.61	0.81	0.71	0.71

as shown in Table 4. Despite the deep architecture used in DenseNet201 with 201 layers, one can notice its validation accuracy is less than VGG16 and VGG19 which argues that the depth of the model has no direct impact on the validation accuracy. VGG16 achieves 64% and 87% test accuracy in the abnormal and normal classes respectively as shown in Table 8. These results are obtained using 194 samples (murmur and extrasystole) for abnormal class and 351 samples for normal class from Pascal dataset as shown in Table 2. The CNN model training is affected by the abnormal class sample number. Concerning the True Positive Rate (TPR) relative to both VGG16 and VGG19 (as seen Table 8), we can deduce the following results: TPR value relative to class 1 (abnormal class) is equal to 0.64 and TPR = 0.87 for class 2 (normal class) with TPR_average = 0.75. This result could be explained through the unbalance property in pascal dataset with 194 samples for abnormal class (after merging murmur and extrasystole samples) and 351 samples for normal class (As seen in Table 2). The CNN model trainining is affected by the abnormal class sample number.

- **Fold2:** In the second fold, VGG16 achieved 80% classification accuracy for the abnormal class and 81% classification accuracy for the normal class. Despite the sample number of abnormal class used in the training process, the TPR values related to both classes are close with TPR = 0.8 for class 1 and TPR = 0.81 for class 2. Which means that the trained samples in abnormal class are relevent and could be adopted for building a useful signature.
- **Fold3:** In the third fold, the classification accuracy of VGG16 is equals to 54% for the abnormal class and 90% for the normal class. The used samples in the training process are not relevant and cause confusing issues with normal classes. This is deduced from TPR = 0.54 related to class 1 and TPR = 0.90 characterizing class 2.

TABLE 11. Validation accuracy for Keras pre-trained models using 2 classes and 2 folds.

Model	2 Fold		
	Fold1	Fold2	AVG
VGG16	0.74766	0.74882	0.74
VGG19	0.78037	0.76303	**0.77**
Xception	0.71028	0.72986	0.71
ResNet152V2	0.71028	0.6872	0.69
ResNet152	0.73364	0.71564	0.72
ResNet101V2	0.70093	0.69668	0.69
ResNet101	0.71028	0.7109	0.71
ResNet50v2	0.71963	0.71564	0.71
ResNet50	0.70093	0.69668	0.69
NasNetMobile	0.70093	0.67773	0.68
NasNetLarge	0.70561	0.64929	0.67
MobileNetV2	0.69159	0.66825	0.67
MobileNet	0.69626	0.7346	0.71
Inceptionv3	0.73364	0.72038	0.72
InceptionResNetV2	0.70093	0.68246	0.69
DenseNet201	0.74766	0.75355	0.74
DenseNet169	0.71028	0.7346	0.73
DenseNet121	0.71963	0.72512	0.72

In fact, Figure 8 gives an overview of the average training time verses the average validation accuracy using Keras pre-trained models. We can see that ResNet50 and ResNet50-V2 have the lowest training time. On the other hand, NasNetLarge has the highest training time and the lowest validation accuracy compared to other models.

(b) **Using two folds:** We relay on splitted dataset into 50% for training and 50% for testing. It consists of 97 abnormal samples and 176 normal samples are used for training and 97 abnormal samples and 175 normal samples are used for testing.

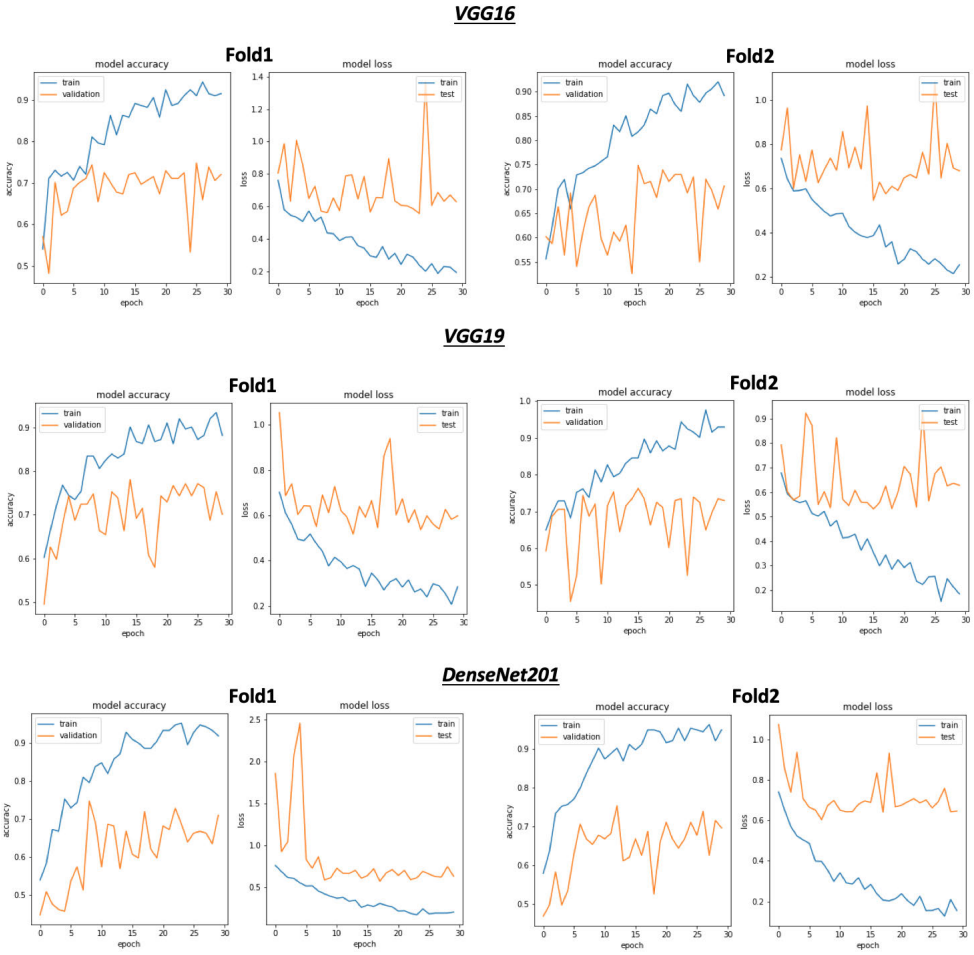

FIGURE 10. The training validations and the loss curves after training the Keras pre-trained CNN models (VGG16, VGG19 and DenseNet201) for 2 classes and 2 folds.

TABLE 12. Validation true positive rate (TPR) of CNN models using 3 classes and 3 folds.

Model	3 Fold												
	Fold1				Fold2				Fold3				AVG
	class 1	class 2	class 3	avg	class 1	class 2	class 3	avg	class 1	class 2	class 3	avg	
VGG16	0.04	0.62	1.0	0.55	0.09	0.83	0.90	0.61	0.23	0.76	0.85	0.62	**0.59**
VGG19	0.18	0.62	0.96	0.59	0.09	0.88	0.88	0.61	0.0	0.69	0.98	0.56	**0.59**
Xception	0.0	0.02	0.98	0.33	0.0	0.0	1.0	0.33	0.0	0.04	1.0	0.34	0.33
ResNet152V2	0.0	0.32	1.0	0.44	0.0	0.41	0.96	0.45	0.0	0.46	0.96	0.47	0.45
ResNet152	0.0	0.51	1.0	0.50	0.04	0.72	0.88	0.54	0.0	0.65	0.89	0.51	0.51
ResNet101V2	0.0	0.30	1.0	0.43	0.0	0.79	0.81	0.53	0.0	0.51	0.93	0.48	0.48
ResNet101	0.0	0.55	1.0	0.51	0.0	0.74	0.92	0.55	0.0	0.60	0.94	0.51	0.52
ResNet50v2	0.0	0.46	1.0	0.48	0.0	0.44	0.97	0.47	0.0	0.13	0.98	0.37	0.44
ResNet50	0.0	0.62	0.97	0.53	0.13	0.81	0.83	0.59	0.0	0.62	0.92	0.51	0.54
NasNetMobile	0.04	0.0	1.0	0.34	0.0	0.02	1.0	0.34	0.0	0.0	1.0	0.33	0.33
NasNetLarge	0.09	0.11	0.98	0.39	0.0	0.04	0.97	0.34	0.0	0.04	0.98	0.34	0.35
MobileNetV2	0.0	0.0	1.0	0.33	0.0	0.11	0.98	0.36	0.0	0.02	1.0	0.34	0.34
MobileNet	0.0	0.48	1.0	0.49	0.0	0.60	0.92	0.50	0.0	0.62	0.93	0.52	0.50
Inceptionv3	0.0	0.02	1.0	0.34	0.0	0.04	1.0	0.34	0.0	0.44	0.98	0.47	0.38
InceptionResNetV2	0.0	0.0	1.0	0.33	0.0	0.0	1.0	0.33	0.0	0.0	1.0	0.33	0.33
DenseNet201	0.0	0.58	0.98	0.52	0.0	0.69	0.90	0.53	0.0	0.60	0.96	0.52	0.52
DenseNet169	0.0	0.62	0.97	0.53	0.0	0.74	0.88	0.54	0.0	0.60	0.89	0.50	0.52
DenseNet121	0.0	0.55	1.0	0.51	0.0	0.74	0.89	0.54	0.0	0.69	0.90	0.50	0.51

- **Fold1:** As shown in Figure 10 and Table 11, the VGG19 pre-trained model achieved training and testing classification accuracy respectively 92% (in EPOCH = 27) and 78% (in EPOCH = 14). Concerning the True Positive Rate (TPR) relative to VGG19 (As seen in Table 10), Class 1 generates

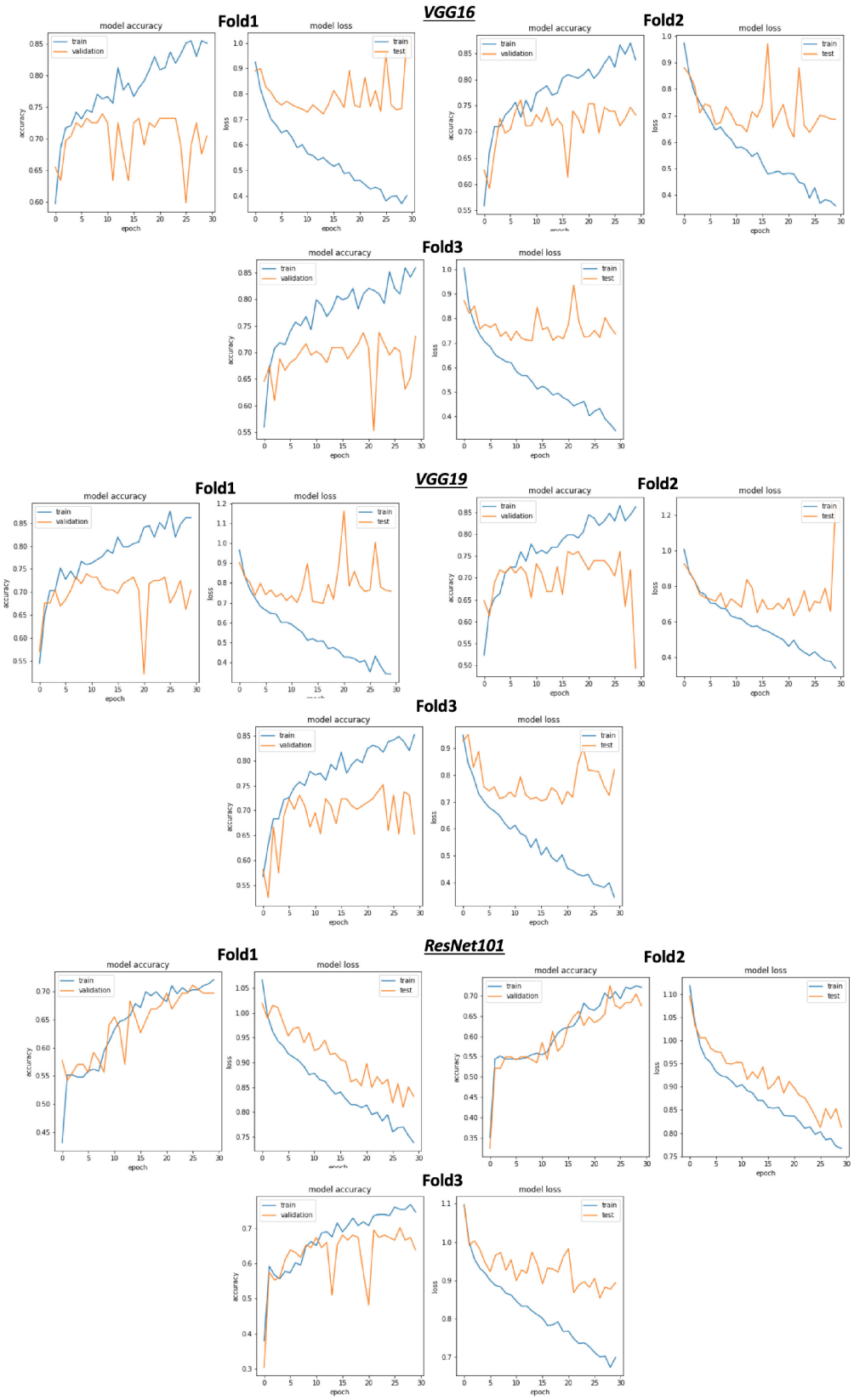

FIGURE 11. The training validations and the loss curves after training the Keras pre-trained CNN models (VGG16, VGG19, and ResNest101) using 3 classes and 3 folds.

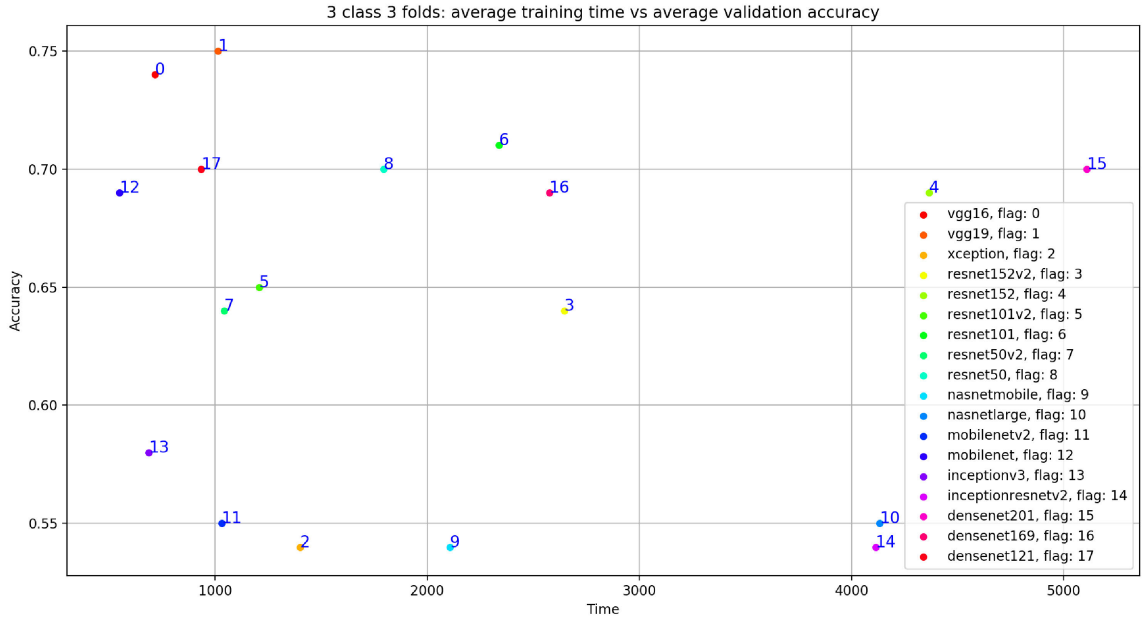

FIGURE 12. An overview of the Keras pre-trained models training time verses validation accuracy using 3 classes and 3 folds.

TPR = 0.63 and TPR = 0.90 for class 2 with TPR average = 0.76 (As seen in Table 10). As shown in Table 2, we have abnormal training samples = 97 (194/2) and normal training samples = 176 (351/2) which is approximately the 1.8 × 97 = 174. This unbalancing problem decreases TPR validation results.

- **Fold2:** VGG19 achieved training and testing classification accuracy equal 95% (in EPOCH = 27) and 76% (in EPOCH = 17). The average validation accuracy is 77% which is the same result obtained using 3 folds. For the second 50 % of the training sample related to both classes, class 1 generated TPR = 0.68 and TPR = 0.83 for class 2 with TPR average = 0.75.

As shown in Figure 9, VGG16 and ResNet50V2 has the lowest training time but VGG19 has a very close training time compared to these models. While VGG19 achieved the highest average validation accuracy, NasNetLarge generated the lowest one.

2) THREE CLASSES (EXTRASYSTOLE/MURMUR/NORMAL) CLASSIFICATION

(a) **Using three folds:** The dataset is splitted into 66% for training and 34% for testing. Where 152 samples for normal class, 85 samples for murmur and 43 samples for extrasystole are used for training and 79 normal, 44 murmur and 22 extrasystole are used for testing (As seen Table 1):

- **Fold1:** As shown in Table 13 and Figure 11, VGG19 reached the highest validation and testing accuracy respectively 88% (in EPOCH = 25) and 74% (in EPOCH = 8). Concerning the True Positive Rate (TPR) relative to both VGG16 and VGG19 (As seen in Table 12), we can deduce that VGG16 TPR value relative to class 1 (extrasystole) is equal to 0.045, TPR = 0.62 for class 2 (murmur) and TPR = 1 for class 3 (normal) with TPR average = 0.55. In this case, the unbalancing property related to pascal dataset especially for ExtraSystole class (just 43 samples for training), has a direct impact on TPR. In other hand, VGG19 gives us the best TPR_average = 0.59 compared to all the other models.
- **Fold2:** Considering the second fold, VGG19 reached also the highest validation and testing accuracy respectively 87% (in EPOCH 28) and 76% (in EPOCH 17). Considering the TPR metric,

TABLE 13. Validation accuracy for the Keras pre-trained models using 3 classes and 3 folds.

Accuracy	3 Fold			
	Fold1	Fold2	Fold3	avg
VGG16	0.73944	0.76056	0.73759	**0.74**
VGG19	0.73944	0.76056	0.75177	**0.75**
Xception	0.54225	0.54225	0.56028	0.54
ResNet152V2	0.64085	0.64789	0.66667	0.64
ResNet152	0.69718	0.70423	0.68794	0.69
ResNet101V2	0.6338	0.6831	0.66667	0.65
ResNet101	0.71127	0.72535	0.70213	0.71
ResNet50v2	0.6831	0.66197	0.58156	0.64
ResNet50	0.71831	0.71831	0.69504	0.70
NasNetMobile	0.5493	0.5493	0.5461	0.54
NasNetLarge	0.58451	0.54225	0.55319	0.55
MobileNetV2	0.54225	0.57042	0.55319	0.55
MobileNet	0.69014	0.6831	0.70213	0.69
Inceptionv3	0.5493	0.55634	0.67376	0.58
InceptionResNetV2	0.54225	0.54225	0.5461	0.54
DenseNet201	0.71127	0.70423	0.70922	0.70
DenseNet169	0.71831	0.70423	0.67376	0.69
DenseNet121	0.71127	0.71127	0.68085	0.70

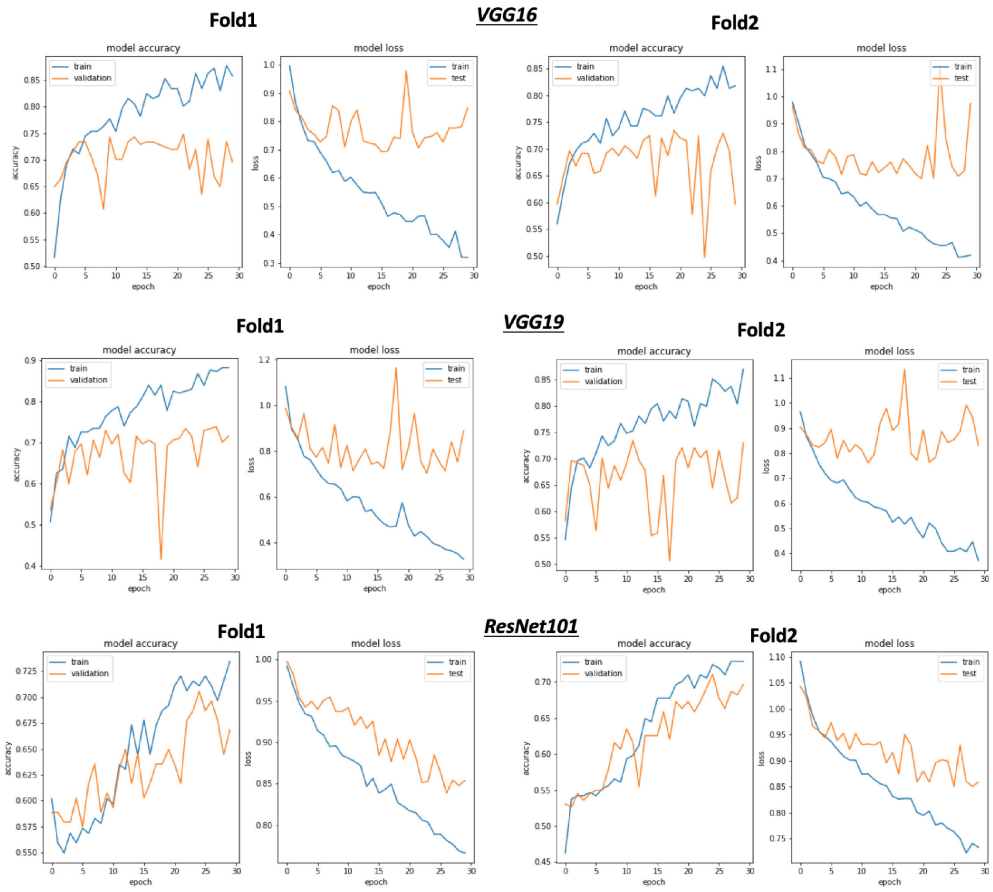

FIGURE 13. The training validations and the loss curves after training the Keras pre-trained models (VGG16, VGG19, and ResNet101) using 3 classes and 2 folds.

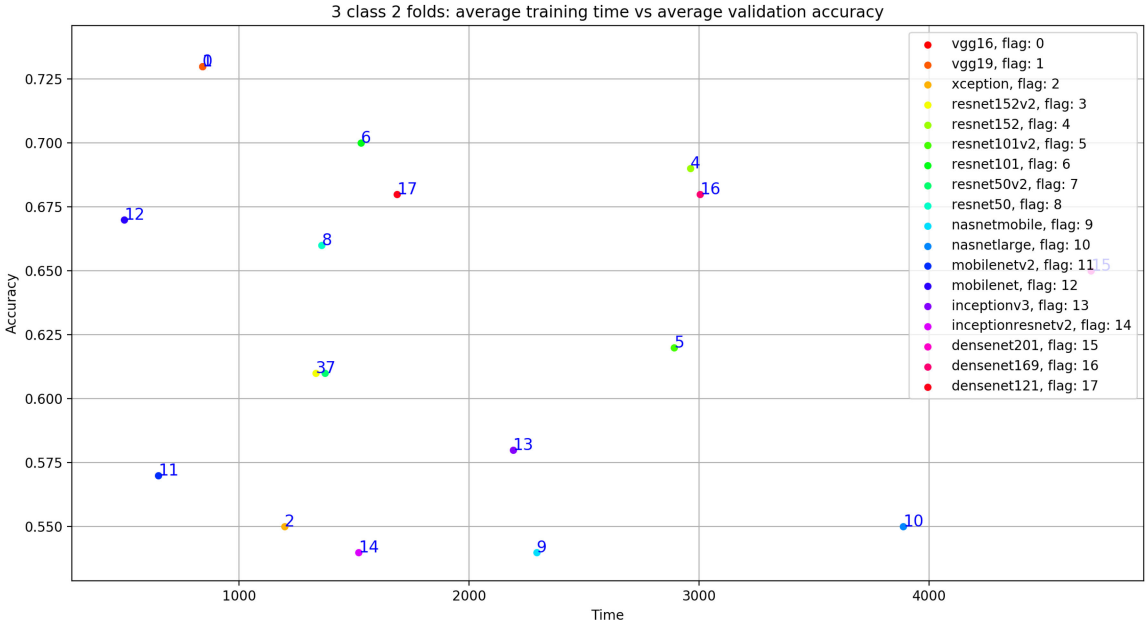

FIGURE 14. An overview of the Keras pre-trained models training time verses validation accuracy using 3 classes and 2 folds.

VGG16 and VGG19 reached very close TPR average values with the very low TPR value related to Extrasystole class.

- **Fold3:** In the third fold, VGG19 reached the highest validation and testing accuracy respectively 85% (in EPOCH 29) and 75% (in EPOCH 23). As seen

TABLE 14. Validation true positive rate (TPR) of CNN models using 3 class 2 folds.

Model	2 Fold								
	Fold1				Fold2				AVG
	class 1	class 2	class 3	avg	class 1	class 2	class 3	avg	
VGG16	0.18	0.69	0.93	0.60	0.06	0.64	0.97	0.55	**0.57**
VGG19	0.03	0.76	0.92	0.57	0.09	0.73	0.91	0.58	**0.57**
Xception	0.0	0.16	0.95	0.37	0.06	0.04	0.96	0.35	0.36
ResNet152V2	0.0	0.44	0.96	0.47	0.0	0.18	0.97	0.38	0.42
ResNet152	0.0	0.50	0.99	0.49	0.0	0.71	0.86	0.52	0.50
ResNet101V2	0.0	0.32	0.99	0.43	0.0	0.32	0.94	0.42	0.42
ResNet101	0.0	0.6	0.96	0.52	0.03	0.68	0.91	0.54	0.53
ResNet50v2	0.0	0.15	1.0	0.38	0.0	0.54	0.90	0.48	0.43
ResNet50	0.0	0.47	0.99	0.48	0.0	0.60	0.86	0.49	0.48
NasNetMobile	0.12	0.06	0.94	0.37	0.0	0.0	1.0	0.33	0.35
NasNetLarge	0.0	0.07	0.98	0.35	0.0	0.01	1.0	0.33	0.34
MobileNetV2	0.0	0.16	0.98	0.38	0.0	0.09	1.0	0.36	0.37
MobileNet	0.0	0.46	0.99	0.48	0.0	0.54	0.93	0.49	0.48
Inceptionv3	0.0	0.18	0.93	0.37	0.0	0.20	1.0	0.40	0.38
InceptionResNetV2	0.0	0.0	1.0	0.33	0.0	0.0	1.0	0.33	0.33
DenseNet201	0.0	0.41	0.99	0.46	0.0	0.56	0.89	0.48	0.47
DenseNet169	0.0	0.55	0.99	0.51	0.0	0.59	0.90	0.49	0.50
DenseNet121	0.0	0.61	0.95	0.52	0.0	0.60	0.87	0.49	0.50

TABLE 15. Validation accuracy for the Keras pre-trained models using 3 classes and 2 folds.

Model	2 Fold		
	Fold1	Fold2	avg
VGG16	0.74766	0.7346	**0.73**
VGG19	0.73832	0.7346	**0.73**
Xception	0.57009	0.54976	0.55
ResNet152V2	0.65888	0.58768	0.61
ResNet152	0.69159	0.69194	0.69
ResNet101V2	0.63551	0.61611	0.62
ResNet101	0.70561	0.7109	0.7
ResNet50v2	0.58879	0.65877	0.61
ResNet50	0.68224	0.65877	0.66
NasNetMobile	0.5514	0.54502	0.54
NasNetLarge	0.55607	0.54976	0.55
MobileNetV2	0.58411	0.57346	0.57
MobileNet	0.67757	0.67773	0.67
Inceptionv3	0.56075	0.60664	0.58
InceptionResNetV2	0.54206	0.54502	0.54
DenseNet201	0.66355	0.65877	0.65
DenseNet169	0.70561	0.67299	0.68
DenseNet121	0.70561	0.66351	0.68

in Table 13, the average test accuracy for VGG16 is close to VGG19. We can deduce that the two extra convolution layers in VGG19 architecture (As seen in Table 4) have a direct impact on the classification test accuracy. Considering TPR metric, VGG16 gives us the best TPR average compared to all the models (As seen in Table 12).

As shown in Table 12, the best TPR average related to all the three folds could be obtained by adopting VGG16 or VGG19 as CNN models. But if we consider the training time vs validation accuracy (As shown in Figure 12) and TPR average, we can conclude that VGG19 could be considered for such configuration.

(b) **Using two folds:** The dataset is splitted into 50% for training and 34% for validation. The training data consists of 116 samples for the normal class, 65 samples for the murmur class and 33 samples for extrasystole class. The testing data consists of 115 samples for normal class, 64 for the murmur class, and 32 for extrasystole class as shown in Table 1.

In the same manner as discussed above using the three folds, VGG19 and VGG16 achieved 73% as average classification test accuracy. As shown in Table 13, they achieved the highest average classification test accuracy compared to other models. Concerning the test accuracy related to the Keras pre-trained models as seen in Table 14, we can deduce the following results:

Considering all the folds, the TPR related to the ExtraSystole class is very low. Therefore, it decreases the average classification TPR obtained from all the pre-trained models applied in Keras. The highest TPR validation average is achieved by using VGG16 or VGG19.

As shown in Figure 14, VGG16 and VGG19 have the same average training time and average validation accuracy. In this experiment, MobileNet has the lowest training time but its test accuracy is ranked on the 7th which comes after VGG16, VGG19 ResNet101, ResNet151, DenseNet121, and DenseNet169 models. Also, ResNet101, which ranks second, has an average classification test accuracy close to VGG19 and VGG16 models but with a larger training time.

In fact, in the work of [30], the authors presented a solution for heartbeat classification based on CNN trained on Pascal dataset. In the pre-processing step, the authors performed cross-cutting and framing to enlarge the data. They obtained 327 multi-class sample from cutting dataset A samples into 5s samples and discard less than 5s. Then they applied Butterworth filter and Fast Fourier Transform in order to extract frequency features to be trained by CNN. They obtained 0.98 as global identification rate (As seen in Table 16). In [37], the authors relayed on Multi-class Pascal dataset B in order

TABLE 16. An overview of our model results compared to some related works.

Works	PASCAL 2011 Signal statistics	Classes	Folds	Accuracy	Precision	Sensitivity
Our method	Raw full dataset	Normal, murmur and Extrasystole	2	**0.73**	-	**0.57**
Our method	Raw full dataset	Normal, murmur and Extrasystole	3	**0.75**	-	**0.59**
Our method	Raw full dataset	Normal and Abnormal	2	**0.77**	-	**0.75**
Our method	Balanced dataset 194 samples	Normal and Abnormal	2	**0.89**	-	-
Our method	Raw full dataset	Normal and Abnormal	3	**0.77**	-	**0.76**
[30]	327 heart sounds from enlarged Dataset A	Normal, murmur, Extrasystole, Artifact	-	0.98	-	-
[37]	Pascal Dataset B	Normal, murmur, Extrasystole	-	0.80	-	-
[15]	Full labeled dataset	Normal, murmur, Extrasystole	-	-	0.71	0.54
[32]	31 signals	Normal, murmur and other sounds	-	0.89	0.91	0.98
[10]	52 signals	Normal and abnormal sounds	-	-	0.63	-
[50]	Full dataset	Normal, murmur and other sounds	-	-	0.67	-
[9]	14 from dataset A and 127 signals from B	Normal and murmurs	-	-	0.78	-
[7]	Full dataset	Normal, murmur and other sounds	-	0.48	-	-
[35]	111 signals	Normal heart sounds and murmurs	-	-	0.986	0.892
[41]	24 normal and 31 abnormal	normal and abnormal	-	87.7	-	96.7

to propose a solution for automatic heartbeat classification. In the pre-processing step of this work, they are based on noise filtering, down-sampling in order to clean the samples and to obtain reduced domain features. For the classification step, the authors applied Recurrent Neural Network (RNN) which is based on Long Short-Term Memory (LSTM) to train and classify the heartbeat samples. Similarly, in the research [15], the authors presented a solution for heartbeat classification based on CNN. In the pre-processing step, they performed spectrogram generation and deep feature extraction. In the classification step, they used transfer learning technique through the use of Pre-trained CNN models such as VGG16, VGG19, and AlexNet. They obtained an average of 0.71 as precision metric (as seen in Table 16). In other word, the majority of previous work used pre-processing step before implementing their solution. Even as shown in Table 16, the majority of cited references does not consider the entire Pascal dataset samples, which does not allow us to compare results objectively. To our knowledge, there is no similar work providing a Benchmark related to the widely used Pascal dataset (as is) using deeper CNN models.

The novelty of our work is to use deeper CNN models through the application of transfer learning technique related to many deep CNN models in order to provide a useful experimental results (Benchmark). All of the used models are trained onto the most used public multi-class dataset (Pascal). We exploited Pascal dataset in two different manner. At first, in order to study the balancing effect onto the classification results, we balanced Pascal dataset training samples by preserving all the abnormal samples which are 194 and we chose randomly 194 samples from normal samples. Concerning our second study, we used the entire Pascal samples as is without any change and we trained our different deep CNN models in order to compare the different results. As seen in Table 16, we can conclude that the balancing step has an important effect on classification results.

V. CONCLUSION AND PERSPECTIVES

In this work, we presented experimental results based on fine tuning pre-trained CNN models using Pascal dataset. We performed CNN training relying on two different deep learning platforms namely, Keras and Matlab Pre-trained Neural Networks. The obtained results can be used as a starting Benchmark reference for future PCG based CVD recognition research. We plan to improve the average validation accuracy and the average test accuracy by using a denoising technique combined with a robust heartbeat cycle segmentation and selection process.

ACKNOWLEDGMENT

The authors, therefore, gratefully acknowledge DSR technical support.

REFERENCES

[1] "Mortality due cardiovascular diseases in world," Tech. Rep., Feb. 2020.
[2] H. M. O. A. Marzuqi, S. M. Hussain, and A. Frank, "Device activation based on voice recognition using Mel frequency cepstral coefficients (MFCC's) algorithm," Tech. Rep., 2019.
[3] T. Alafif, Z. Hailat, M. Aslan, and X. Chen, "On classifying facial races with partial occlusions and pose variations," in *Proc. 16th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2017, pp. 679–684.
[4] T. Alafif, Z. Hailat, M. Aslan, and X. Chen, "On detecting partially occluded faces with pose variations," in *Proc. 14th Int. Symp. Pervasive Syst., Algorithms Netw. 11th Int. Conf. Frontier Comput. Sci. Technol. 3rd Int. Symp. Creative Comput. (ISPAN-FCST-ISCC)*, Jun. 2017, pp. 28–37.
[5] S. Ari, K. Hembram, and G. Saha, "Detection of cardiac abnormality from PCG signal using LMS based least square SVM classifier," *Expert Syst. Appl.*, vol. 37, no. 12, pp. 8019–8026, Dec. 2010.
[6] M. S. Aslan, Z. Hailat, T. K. Alafif, and X.-W. Chen, "Multi-channel multi-model feature learning for face recognition," *Pattern Recognit. Lett.*, vol. 85, pp. 79–83, Jan. 2017.
[7] C. C. Balili, M. C. C. Sobrepena, and P. C. Naval, "Classification of heart sounds using discrete and continuous wavelet transform and random forests," in *Proc. 3rd IAPR Asian Conf. Pattern Recognit. (ACPR)*, Nov. 2015, pp. 655–659.
[8] P. Bentley, G. Nordehn, M. Coimbra, and S. Mannor. (2011). *The PASCAL Classifying Heart Sounds Challenge 2011 (CHSC2011) Results*. [Online]. Available: http://www.peterjbentley.com/heartchallenge/index.html
[9] F. Chakir, A. Jilbab, C. Nacir, and A. Hammouch, "Phonocardiogram signals classification into normal heart sounds and heart murmur sounds," in *Proc. 11th Int. Conf. Intell. Syst., Theories Appl. (SITA)*, Oct. 2016, pp. 1–4.
[10] F. Chakir, A. Jilbab, C. Nacir, and A. Hammouch, "Phonocardiogram signals processing approach for PASCAL classifying heart sounds challenge," *Signal, Image Video Process.*, vol. 12, no. 6, pp. 1149–1155, Sep. 2018.
[11] S. Chauhan, P. Wang, C. Sing Lim, and V. Anantharaman, "A computer-aided MFCC-based HMM system for automatic auscultation," *Comput. Biol. Med.*, vol. 38, no. 2, pp. 221–233, Feb. 2008.

[12] T.-E. Chen, S.-I. Yang, L.-T. Ho, K.-H. Tsai, Y.-H. Chen, Y.-F. Chang, Y.-H. Lai, S.-S. Wang, Y. Tsao, and C.-C. Wu, "S1 and S2 heart sound recognition using deep neural networks," *IEEE Trans. Biomed. Eng.*, vol. 64, no. 2, pp. 372–380, Feb. 2017.
[13] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.
[14] N. Dave, "Feature extraction methods LPC, PLP and MFCC in speech recognition," *Int. J. Adv. Res. Eng. Technol.*, vol. 1, no. 6, pp. 1–4, 2013.
[15] F. Demir, A. Şengür, V. Bajaj, and K. Polat, "Towards the classification of heart sounds based on convolutional deep neural network," *Health Inf. Sci. Syst.*, vol. 7, no. 1, p. 16, Dec. 2019.
[16] S. S. Farfade, M. J. Saberian, and L.-J. Li, "Multi-view face detection using deep convolutional neural networks," in *Proc. 5th ACM Int. Conf. Multimedia Retr. (ICMR)*, 2015, pp. 643–650.
[17] C. Garcia and M. Delakis, "Convolutional face finder: A neural architecture for fast and robust face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 11, pp. 1408–1423, Nov. 2004.
[18] W. Han, C.-F. Chan, C.-S. Choy, and K.-P. Pun, "An efficient MFCC extraction method in speech recognition," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2006, p. 4.
[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[20] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861
[21] M. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.
[22] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.
[23] 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*. [Online]. Available: http://arxiv.org/abs/1602.07360
[24] M. A. Kotb, H. Nabih, F. E. Zahraa, M. E. Falaki, C. W. Shaker, M. A. Refaey, and K. W. Rjoob, "Improving the recognition of heart murmur," *Int. J. Adv. Comput. Sci. Appl*, vol. 7, no. 7, pp. 283–287, 2016.
[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[26] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 98–113, Jan. 1997.
[27] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[28] H. Lee, C. Ekanadham, and A. Y. Ng, "Sparse deep belief net model for visual area V2," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 873–880.
[29] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5325–5334.
[30] T. Li, C. Qing, and X. Tian, "Classification of heart sounds based on convolutional neural network," in *Proc. Int. Conf. Internet Multimedia Comput. Service*. Springer, 2017, pp. 252–259.
[31] C. Liu, D. Springer, Q. Li, B. Moody, R. A. Juan, F. J. Chorro, F. Castells, J M. Roig, I. Silva, A. E. Johnson, and Z. Syed, "An open access database for the evaluation of heart sound algorithms," *Physiol. Meas.*, vol. 37, no. 12, p. 2181, 2016.
[32] S. I. Malik, M. U. Akram, and I. Siddiqi, "Localization and classification of heartbeats using robust adaptive algorithm," *Biomed. Signal Process. Control*, vol. 49, pp. 57–77, Mar. 2019.
[33] Mathworks. (2019). *Pretrained Deep Neural Networks*. Accessed: Dec. 12, 2019. [Online]. Available: https://www.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html
[34] D. M. Nogueira, C. A. Ferreira, E. F. Gomes, and A. M. Jorge, "Classifying heart sounds using images of motifs, MFCC and temporal features," *J. Med. Syst.*, vol. 43, no. 6, p. 168, Jun. 2019.
[35] J. Pedrosa, A. Castro, and T. T. V. Vinhoza, "Automatic heart sound segmentation and murmur detection in pediatric phonocardiograms," in *Proc. 36th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Aug. 2014, pp. 2294–2297.
[36] C. Potes, S. Parvaneh, 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. (CinC)*, Sep. 2016, pp. 621–624.
[37] A. Raza, A. Mehmood, S. Ullah, M. Ahmad, G. S. Choi, and B.-W. On, "Heartbeat sound signal classification using deep learning," *Sensors*, vol. 19, no. 21, p. 4819, Nov. 2019.
[38] G. Redlarski, D. Gradolewski, and A. Palkowski, "A system for heart sounds classification," *PLoS ONE*, vol. 9, no. 11, Nov. 2014, Art. no. e112673.
[39] J. Rubin, R. Abreu, A. Ganguli, S. Nelaturi, I. Matei, and K. Sricharan, "Recognizing abnormal heart sounds using deep learning," in *Proc. KHD IJCAI*, 2017, pp. 1–7.
[40] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.
[41] G. Sidra, N. Ammara, H. Taimur, H. Bilal, and A. Ramsha, "Fully automated identification of heart sounds for the analysis of cardiovascular pathology," in *Applications of Intelligent Technologies in Healthcare*. Springer, 2019, pp. 117–129.
[42] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556
[43] M. Singh and A. Cheema, "Heart sounds classification using feature extraction of phonocardiography signal," *Int. J. Comput. Appl.*, vol. 77, no. 4, pp. 13–17, Sep. 2013.
[44] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 1–7.
[45] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[46] 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. (CVPR)*, Jun. 2016, pp. 2818–2826.
[47] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Proc. Int. Conf. Artif. Neural Netw.* Springer, 2018, pp. 270–279.
[48] H. Tang, H. Chen, T. Li, and M. Zhong, "Classification of normal/abnormal heart sound recordings based on multi: Domain features and back propagation neural network," in *Proc. Comput. Cardiol. Conf. (CinC)*, Sep. 2016, pp. 593–596.
[49] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "From facial parts responses to face detection: A deep learning approach," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 3676–3684.
[50] W. Zhang, J. Han, and S. Deng, "Heart sound classification based on scaled spectrogram and partial least squares regression," *Biomed. Signal Process. Control*, vol. 32, pp. 20–28, Feb. 2017.
[51] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.
[52] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8697–8710.

MEHREZ BOULARES received the master's degree from the Higher National Engineering School of Tunis, in 2009, and the Ph.D. degree from the University of Sfax, Tunisia, in December 2016. He is currently a Postdoctoral Researcher in information and communication technologies with the Faculty of Computing and IT (FCIT), King Abdulaziz University (KAU). He is an active researcher in the artificial intelligence field. He published nearly 14 articles in peer-reviewed journals and international conferences.

TARIK ALAFIF received the master's degree from Gannon University, Erie, PA, USA, in 2011, and the Ph.D. degree from the Department of Computer Science, Wayne State University, Detroit, MI, USA, in 2017. He is currently an Assistant Professor with the Department of Computer Science, Jamoum University College, Umm Al-Qura University. He published several peer-reviewed articles in his research areas. His main research interests include computer vision, machine learning, deep learning, and large-scale data.

AHMED BARNAWI received the M.Sc. degree from The University of Manchester (UMIST), U.K., in 2001, and the Ph.D. degree from the University of Bradford, U.K., in 2005. He acted as an associate and visiting professor in Canada and Germany. He is currently a Professor of information and communication technologies with the Faculty of Computing and IT (FCIT), King Abdulaziz University (KAU). He is the Managing Director of the KAU Cloud Computing and Big Data Research Group. He is an active researcher with good research fund awards track. He published nearly 100 articles in peer-reviewed journals. His research interests include big data, cloud computing, future generation mobile systems, advanced mobile robotic applications, and IT infrastructure architecture.