Acta Scientific COMPUTER SCIENCES

Volume 4 Issue 6 June 2022

Research Article

Covid-19 and Cardiac Disease Classification using ECG Images

Anthome Aalwan Vaz[1], Mohammed Imran[1], Muruganantham Sadasivam[1], Syed Farhan Hassan[1], Vigneshwaran Mahendiran[1] and Parisa Naraei[2]*

[1]*AIMT, Lambton College, Toronto, Canada*

[2]*Principal Investigator Applied Research Center, Lambton College, Canada*

***Corresponding Author:** Parisa Naraei, Principal Investigator Applied Research Center, Lambton College, Canada.

Received: April 26, 2022

Published: May 18, 2022

Abstract

The initial phase of the project has been started with the data collection, where we have collected ECG images and 12 lead ECG waveform values of patients affected with covid 19 with five classifications as Covid 19, abnormal heart rate, myocardial infarction (MI), history of MI and normal ECG. We have done gamma correction for the image dataset in data preprocessing and cleaning, and waveforms have been created from 12 lead ECG values. To increase the available data, we have augmented the created images. We have created test and validation classes for the image dataset to train the model. A sequential convolutional neural network has been built for image classification. We used the batch normalization method from Keras; in the sequential model layer, we used relu and softmax activation layers—Adam optimizer and sparse categorical cross-entropy as the loss function in the sequential model. Training and validation accuracy have been used as metrics to assess the model's performance. Vgg16 model has been trained to compare the results. On completion of model creation, the trained model has been exported, and mobile application integration has been made for the end-user.

Keywords: Covid-19; ECG Image; Arrhythmia; Data Collection; Data Preprocessing; Data Cleaning; Data Augmentation; Model Building; Training; Model Evaluation; Android Application Development; CNN; Custom CNN; Vgg16

Introduction

Coronavirus 2019 (COVID-19) has caused a huge and rapid increase in deaths, resulting in a long-lasting pandemic. As of May 21, 2021, over 166 million illnesses and over 3.4 million deaths had been documented worldwide [1]. SARS-CoV-2 can cause organ failure, but this is extremely rare. As a result of extensive heart damage, a network of trained computers can now reliably make computer-aided diagnostic decisions, thanks to an artificial intelligence-based biomedical application (such as medical physicians, healthcare staff members). A number of deep learning models for detecting anomalies in medical images, such as chest X-rays and CT scans, have recently been published. An innovative technique, developed by Degerli., *et al*. was developed to identify the location and severity of COVID-19 infection by studying 15495 CXR images. Over 98% of accurate results were obtained using this technique [2,6]. A new convolutional neural network (CNN) model proposed by Kesim., *et al*. was used to classify chest X-ray images [3]. Diagnosing tuberculosis (TB) with a chest X-ray has been proposed by Liu., *et al*. Researchers used shuffle sampling to create a new CNN model that was 85.68 percent accurate. Using computed tomography (CT) scans, Rahman., *et al*. determined whether or not patients had symptoms of pulmonary tuberculosis (TB) [4]. The proposed model was trained on 3,500 images of infected and healthy CXRs. DenseNet201 had the best sensitivity and specificity.

There have been reports of people who were otherwise healthy developing acute myocarditis after being exposed to COVID-19.

There was critical myocardial damage in up to 27.8% of Chinese COVID-19 patients with troponin levels above the upper 99th percentile of the reference range. That's more than ten times the flu virus's prevalence rate (2.9 percent). It's common for COVID-19 patients to experience mild illness and recovery, despite the presence of biochemical evidence for acute myocardial injury. No one knows for sure if COVID-19 survivors who show no obvious signs of cardiac damage have any hidden or subclinical cardiac damage that could affect long-term outcomes. No one. Survivors of COVID-19 will need to have their hearts monitored after the epidemic has passed to see if this is necessary. The automated 12-lead ECG diagnostic techniques can be used to screen the general public and get a second opinion for medical professionals.

Related work

- Described an improved strategy for training the suggested network by transferring knowledge from the related domain of general texture categorization. Six publicly available texture databases are used to train networks, which are subsequently fine-tuned on lung tissue data. The generated CNNs are integrated into an ensemble, and their knowledge is compressed back into the original network [1].
- Pre-trained CNNs outperformed or, in the worst scenario, performed just as well as a CNN taught from scratch when fine-tuned to their best ability, according to numerous trials. Secondly, fine-tuned CNNs were better able to deal with large training sets than CNNs that were built from the start [7].
- According to the system, based on the percentage of infected lungs, the system categorizes the severity of COVID-19 as mild, moderate, severe, or critical. An extensive collection of tests was carried out using the most up-to-date deep Encoder-Decoder Convolutional Neural Networks available on the market (ED-CNNs) [8].
- When it comes to training the network, they have investigated the effectiveness and efficiency of shuffle sampling with cross-validation, and they have discovered that it has a powerful effect on medical picture classification [9].
- Being the first study to investigate whether deep convolutional neural network (CNN) models can be used to detect COVID-19 from ECG trace images. In this research, 'Deep learning algorithms were used to identify COVID-19 and other cardiovascular illnesses (CVDs) [10].
- This research focused on using AI to detect COVID-19 from chest X-ray images. This research proposes a robust technique for detecting COVID-19 pneumonia from digital chest X-ray pictures using pre-trained deep-learning algorithms. Transfer learning was utilized for training and testing many pre-trained deep Convolutional Neural Networks (CNNs) [11].
- With the help of a mixture of deep learning methods, such as feature extraction, fine-tuning of pre-trained CNNs, and the end-to-end training of the final CNN model, this study found that CNNs can be used to detect COVID-19 and healthy chest X-ray images. Extraction of deep features was carried out using a variety of deep CNN models (ResNet18 through ResNet100), all of which were developed by the team at ResNet. Several kernel functions, including Linear, Quadratic, Cubic, and Gaussian, were employed in the classification of the deep features [12].
- Finesse was achieved by utilizing previously trained deep convolutional networks in this investigation. This work proposes a new CNN model that uses end-to-end training. The linear kernel function was used to extract the deep features from the ResNet50 model and SVM classifier [13].
- To avoid false alarms, this study employs a novel 1D Convolutional Neural Network implementation along with a verification model. Following the encoder and decoder blocks, a sample-wise classification layer generates a 1D segmentation map of the R-peaks in the input ECG signal. It can be used to detect R-peaks in single-channel ECG data streams [14].
- For the purposes of this study, six distinct Convolutional Neural Networks (CNNs) were explored. When it comes to recognizing COVID-19 from regular and segmented lung CXR pictures, the gamma correction technique beats all other enhancing strategies. Detection of COVID-19 with gamma correction improved the accuracy, precision, sensitivity, f1-score, and specificity of the test results [15].

Technical requirements and tools

JupyterLab and Google colab is used for initial development, and the GitHub version control system is used for collaboration. Certain data preprocessing and modeling processes took more memory and could not run the tasks in the local machine and colab

free version. So, we used Colab pro, paid version of google colab, to overcome the memory issue and processing time.

- Modelling: Tensorflow - CNN, Vgg16
- Python Libraries: OpenCV, PIL, Split-folders
- Version control system: GitHub
- Project Management: Jira, Confluence
- Data Collection: Public Data Archives
- Android App: Android Studio, Tensorflow Lite.

Data collection

Mendeley

Mendeley Data is a free and open research data repository that allows academics to publicize their research data. Compliance with funding requirements, permitting reuse by other researchers and boosting the repeatability, transparency, and confidence in the original study are all advantages of sharing research data [6].

UCI machine learning repository

The UCI Machine Learning Repository is a collection of data generators, domain theories, and datasets used by the machine learning community to conduct empirical research on machine learning methods [5].

Physionet.org

The MIT Laboratory for Computational Physiology manages PhysioNet, a library of openly accessible medical research data. The goal of the PhysioNet Resource is to promote present research and future inquiries into the understanding of complicated biomedical and physiologic signals.

Data understanding

Mendeley dataset - ECG images of five classes

The collection comprises electrocardiogram (ECG) pictures of cardiac and COVID-19 patients drawn from 1937 unique patient records. The data is acquired using an ECG device called the EDAN SERIES-3 put in Cardiac Care and Isolation Units of various health care institutions across Pakistan. The data are necessary for screening Cardiac and COVID-19 patients and their linkages. The dataset evaluates multiple CNN models for COVID-19 and other cardiovascular diseases.

ECG pictures collected are divided into five main categories.

- (ECGs) of COVID-19 patients were obtained from the "COVID-19 isolation unit."
- ECG photographs of a standard group were obtained from "Patient's attendants and visitors."
- ECGs of myocardial patients were obtained from the "Cardiac Care Unit."
- ECG pictures of patients recently discharged from the "Out-Patient Department" were obtained.
- ECG scans of patients with abnormal heartbeats who have recently recovered from COVID-19 and Myocardial Infarction and are experiencing shortness of breath were acquired from the "Out-Patient Department".

Category	Number of Images
Normal	859
Covid-19	250
Myocardial infarction (MI)	77
Abnormal Heart Rate	548
Recovered MI	203

Table 1: Mendeley ECG Images dataset description.

The collected ECG images data is from a 500 Hz sample rate 12-leads ECG device.

UCI - arrhythmia dataset

UCI Arrhythmia dataset contains 452 data records with 279 attributes. The dataset labels arrhythmia in a patient and classifies the arrhythmia into one of 16 groups. The values of attributes present in the dataset are nominal and linear numerical. The attributes are related to sex, age, patient-related information, and numeric values of the PQRS waveform of a heartbeat.

The dataset has 16 labels to classify the 12-lead ECG signal. One label is to type the normal signals, and the other 15 groups organize the different arrhythmia signs in the ECG signals.

Physionet.org - MIMIC -III

MIMIC-III is an extensive, freely accessible database containing de-identified health-related data for over 40,000 patients who stayed at Beth Israel Deaconess Medical Center's critical care units between 2001 and 2012. Demographics, bedside vital sign measures (1 data point per hour), laboratory test results, procedures, drugs, carer notes, imaging reports, and death are all included in the database (including post-hospital discharge).

MIMIC-III waveform database

It contains 67,830 record sets for nearly 30,000 ICU patients. Almost every record set includes an ECG waveform record and a numerics record, including time series of periodic readings. These depict a near-continuous recording of a single patient's vital signs during his ICU stay (many of them are of weeks duration) [17-19].

MIMIC-III waveform database matched subset

Matched Subset database is a subset of the MIMIC-III waveform database, consisting of the records for which the patients' clinical records are available in the MIMIC-III Clinical database. The Clinical Database contains patients' non-medical details such as demographics, vital sign measurements made at the bedside (~1 data point per hour), laboratory test results, procedures, medications, caregiver notes, imaging reports, and mortality [16,18,19].

Extracting ECG waveforms from MIMIC-III matched database using WFDB

The waveform database (WFDB) package for Python is a library of tools for reading, writing, and processing physiological signals and annotations, written in the Python programming language. WFDB can be used to directly fetch data from PhysioNet MIMIC-III Waveform or Matched database and generate ECG waveform out of it. We also used WFDB to extract sample waveform from the MIMIC-III Matched database as below [17-19].

Challenges with MIMIC-III databases

The size of MIMIC3 is huge, 6.4 TB, as it contains video data of ECG recordings of 40,000 patients. It's impossible for us to download and process such an amount of data due to constraints of personal laptops and metered internet connection.

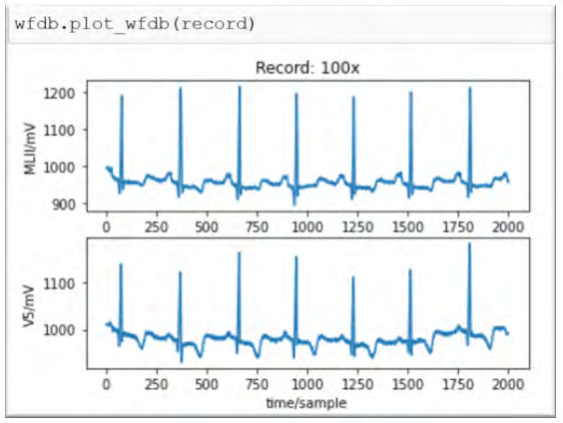

Figure 1: 2-Channeled ECG waveform Snapshot extracted using WFDB.

However, we don't need complete MIMIC3 data as it contains continuous (video) data of all the patients, including non-cardiac patients. For our CNN classification, we just need a single snapshot of ECG and that also of cardiac patients only.

Explored approaches for filtering MIMIC-III Matched Subset Database

- **Filtering of ICU patients:** We planned to use the ICU table of the MIMIC-III Clinical database to get identifiers of ICU patients as the ECG collected in MIMIC3 Matched database is just from ICU patients.
- **Filtering Cardiovascular patients:** There are multiple tables in the Clinical database that we can use to separate cardiac patients, such as ICD (International Classification of Diseases), Chief Complaints, Diagnosis Billing table, and In-Home Medication.
- **Filtering ECG Video:** As we just need the snapshot of ECG, we can pick a few frames of ECG video which should be relevant to cardiovascular diseases.

De-Scoping of MIMIC-III database

Unfortunately, after spending so much effort and time, unfortunately, we had to de-scope the MIMIC-III databases for two primary reasons.

- **Incompatible 2-Channeled ECG:** As explained above, we were able to extract ECG waveforms from MIMIC-III, but they were 2-channeled waveforms, which are incompatible with other data sources - both the Mendeley and UCI have 12-channeled ECG images.
- **Unable to filter huge 6.4TB MIMIC-III database:** We could have used different tables (ICD, Billing, Chief Complaints, ICU) in MIMIC-III clinical databases to filter out the relevant patients. But to access MIMIC-III Clinical, we need elevated access and mandatory certifications, as our college is not partnered with the certification organization. The certification cost is around 300 CAD. Due to this high cost, we even dropped the idea of separately training or augmenting 2-channeled ECG waveforms from MIMIC-III databases.

Data preprocessing

Data transformation

Mendeley Dataset - ECG images

Using OpenCV, we read every Image of ECG in each class. The photos are in colour scale. OpenCV reads colour images as in the BGR order. We used the cvtcolor function from OpenCV to convert the images back to their original order, RGB. After successful colour image reading, the images are converted to grayscale. Grayscale conversion helps for faster processing in the Convolutional neural network models. Our application doesn't require colour images for the modeling, so the idea was to convert images to grayscale using the IMREAD_Grayscale built-in function to reduce the computational load.

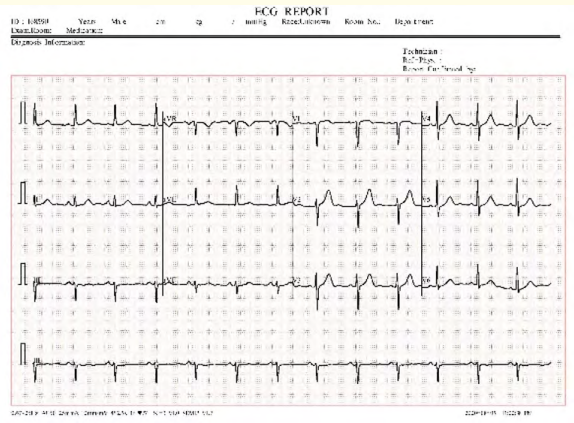

Figure 2: ECG original colour image.

The amplitude and time of the waveform are present; pixels of the image should be uniformly distributed to identify the different classes of the image. The model can easily detect and learn the minute details and differences from the image. The traditional and efficient method to make an image uniformly distributed is Histogram equalization. Equalize Hist function from the OpenCV library will cause the image pixels to the normal distribution and improve the picture's overall contrast.

We used Adaptive Threshold from OpenCV to enhance the white and black pixels of the image since the scanned ECG images had different lighting. Also, our other process only requires black and white pixels of the ECG images to classify them effectively.

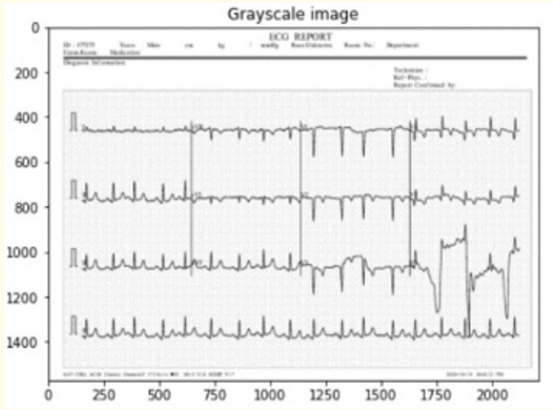

Figure 3: GrayScale converted ECG image.

Generating ECG waveforms from UCI numeric data

The UCI dataset contains ECG details of various patients in the form of numeric values. It contains 279 attributes, out of which four features are patient's details, such as age, sex, height, and weight, and the remaining 275 features are numeric values for different attributes of ECG waveform, such as QRS duration, P/Q/R/S/T intervals, amplitude and width of waves, etc [5].

We used matplotlib's pyplot library to generate ECG waveforms out of these numeric values related to amplitude, width, and time period of waves. It was a challenging and complex task overall, so we adopted a bottom-up approach to complete it. Firstly, smaller and individual components of ECG waveform, such as P wave, QRS complex, and ST wave, were plotted in matplotlib using the numeric feature values in the CSV files sourced from UCI. Then

these smaller components were merged to form a complete ECG waveform 'PQRST' corresponding to a particular channel, as in the below image.

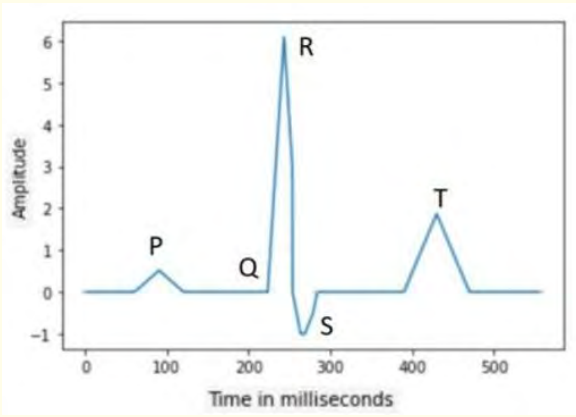

Figure 4: Single Channelled ECG waveform generated from UCI Arrhythmia (CSV) data.

However, if we see ECG images available in the Mendeley dataset, they are complete 12 channeled waveforms containing channels - (1 to 3), aVR, aVL, aVF, (V1 to V6). So, we also generated 12 channeled waveforms for UCI numeric values. The mentioned process of generating PQRST waveforms was repeated for all the channels; numeric values corresponding to each channel were used to plot individual waveforms. Finally, these individuals representing each channel were merged to form the final 12-channeled ECG waveform for a single record (patient), as in the below image.

Figure 5: Multi Channelled ECG waveform generated from UCI Arrhythmia (CSV) data.

Furthermore, we need to generate above 12 channeled waveforms for all the records or patients, but before that, we had to treat missing values in the dataset.

Data cleaning

Mendeley dataset - ECG images

Some ECG images had unnecessary details such as name, date, doctor's details, and outer border. These details will reduce the performance and confuse the machine learning models. We cropped the images to have only the 12-Lead waveform to avoid this. Using the contour function of OpenCV, images are cropped in the dataset. The contour identifies the Image objects and outputs the boundary—contour tales threshold as input to determine the edges of the pixels.

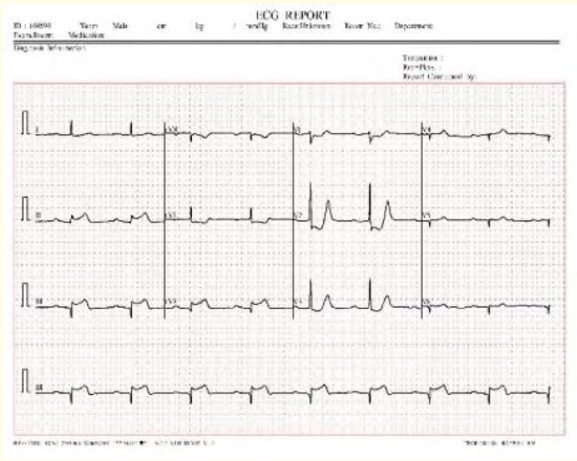

Figure 6: Different ECG Images.

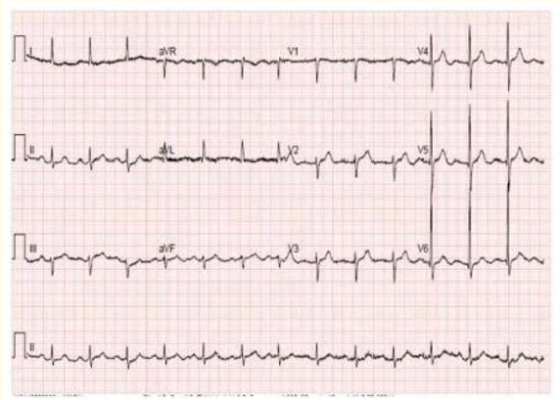

Figure 7: Different ECG Images.

After cropping the images, we wanted to remove the background graph lines of the grayscale ECG images (Figure). So the model can learn and predict the enhanced image easily and efficiently.

The LUT function is used to enhance the image further and remove the graph line of ECG from the image. The lookup table function of OpenCV takes images, lookup-table, and output array as inputs. It's a void function and fills the output array with pixels of the image using the image and lookup table input.

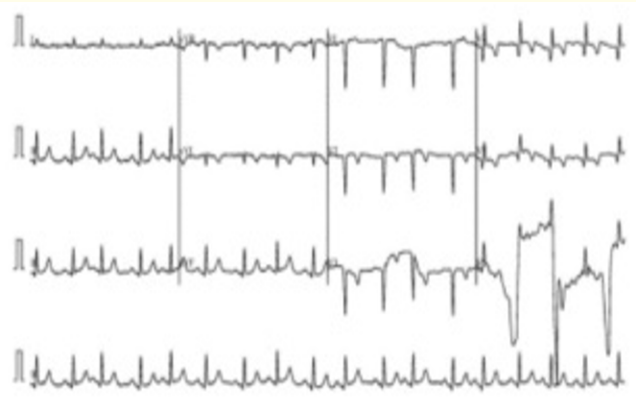

Figure 8: Cropped and Enhanced Image (Y-axis represents Amplitude and X-axis represents the duration in mSec).

Treating missing values in UCI dataset

In the UCI dataset, out of 279 columns (features), one column has missing values (it contained '?'), and another column also had some missing values. These columns could not be dropped as they were necessary to generate continuous ECG waveforms. Out of 126,108 (452 rows * 279 attributes) data points, 408 values were missing ('?').

To treat these missing values, firstly, we convert them all into null values (np.nan) and then use Simple Imputer's mean to replace these null values.

After treating the missing value, 12-channeled ECG waveforms were generated for all the 452 records (patients). These were then augmented with Mendeley ECG images to be used in downstream algorithms.

Data augmentation

Our original dataset is imbalanced, and after data transformation and cleaning, there is negligible data loss. We have used the data augmentation technique to handle the imbalanced dataset since the dataset contains image data.

The data augmentation technique on image data will generate new images by shifting the width and height and zooming the original pictures.

We used the image data generator function from Keras preprocessing to do the augmentation of images. The function takes various arguments as input to shift, rotate and zoom the original image. We used width and height shift range as 0.1 to create new shifted images. We used zoom mode as another augmented technique. To have a white background in the augmented image, we constantly used Fill mode and 255 as values for the white background.

The batch size is set to eight for the output images in the image data generator. To store the output images, the image data gen flow method from Keras preprocessing. The method takes the directory and format to save the image.

Category	Number of Images	Augmented Images
Normal	859	1718
Covid-19	250	2000
Myocardial infarction (MI)	77	616
Abnormal Heart Rate	548	1644
Recovered MI	203	1624
Arrhythmia	452	3616

Table 2: Input Images after Augmentation.

Train and test split

Split folders python library is used to split the test and validation data. Since the dataset is in image format and the prediction is classified, we used folder structure to separate the test and validation data.

The split folder ratio function will accept the input and output directory and the ratio as inputs to read and write the images. The

ratio will split the total data set into a set percentage. The function will generate two-parent folders as test and Val and create the subfolder of different classes.

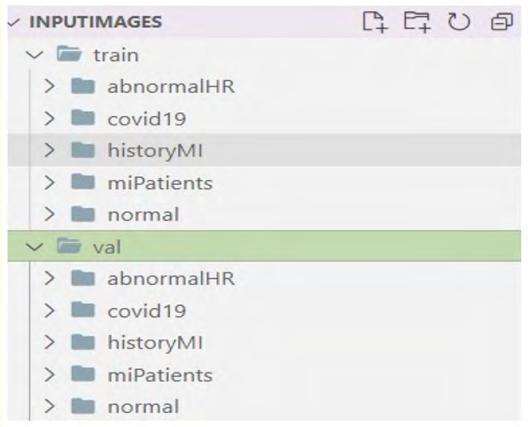

Figure 9: Test and Validation split of the dataset.

Model building and training

VGG 16 pretrained model

VGG16 is an architecture based on convolution neural networks (CNNs). It is widely regarded as one of the most excellent vision model architectures created to date. They emphasized having convolution layers of a 3x3 filter with a stride one instead of having a huge number of hyper-parameters. They always utilized the same padding and maxpool layer of a 2x2 filter with a stride two. This is the unique part of VGG16. The convolution and max pool layers are arranged throughout the entire architecture, which is continuous throughout the architecture. After that, it has two FC (completely connected layers) followed by a softmax for output. The 16 in VGG16 alludes to the fact that it contains 16 layers with different weights. This network is quite vast, with approximately 138 million (approximately) parameters spread over it. A sequential model is one in which all of the model layers are placed in a sequential fashion. The goal of ImageDataGenerator is to make it simple to import data with labels into the model. It is an extremely helpful class because it contains several functions such as resizing, rotating, zooming, and flipping.

16 layers of VGG16

- Convolution using 64 filters
- Convolution using 64 filters + Max pooling

Figure 10: Layers of VGG16.

- Convolution using 128 filters
- Convolution using 128 filters + Max pooling
- Convolution using 256 filters
- Convolution using 256 filters
- Convolution using 256 filters + Max pooling
- Convolution using 512 filters
- Convolution using 512 filters
- Convolution using 512 filters+Max pooling
- Convolution using 512 filters
- Convolution using 512 filters
- Convolution using 512 filters+Max pooling
- Fully connected with 4096 nodes
- Fully connected with 4096 nodes
- Output layer with Softmax activation with 1000 nodes.

Custom built model - CNN

Using the image_dataset_from_directory built-in method from Keras preprocessing class, we read the image data directly from the test and Val folders. The function reads the data from the folder-subfolder structure and labels the classes of the data based on the same structure.

The custom model is built on the TensorFlow convolutional neural network. There are ten layers, out of which nine are input, and one is output. Two layers of 2-dimensional convolution layers, three layers of maxpooling, two dense layers, and flatten and dropout are one each.

We have used a sequential model from the TensorFlow python library. To optimize the weights in each neuron of a neural network, an optimizer should be used with the model. Adam optimizer is

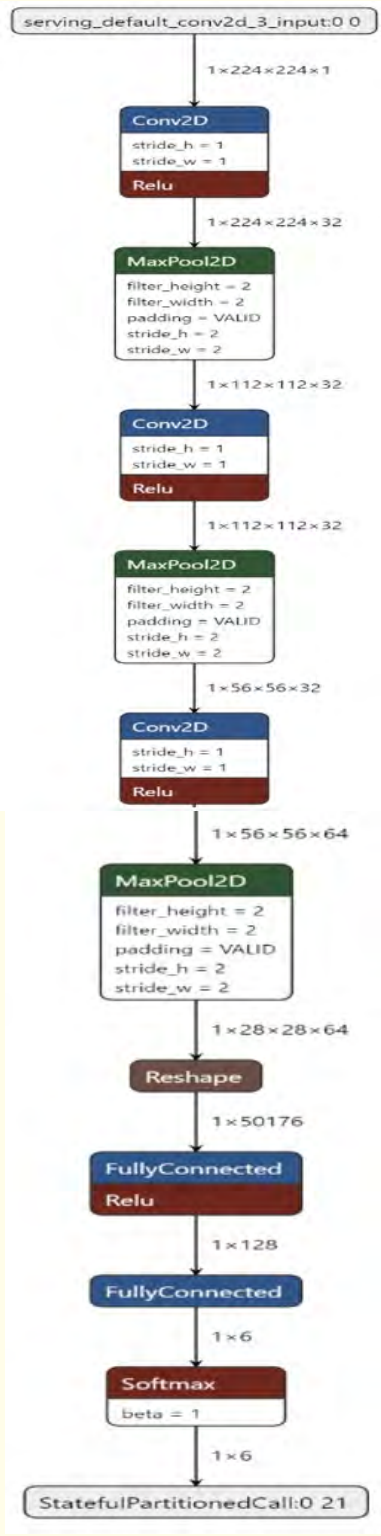

Figure 11: Model Summary of Stable and final Model.

used since it works well with gradient descent. A sparse categorical loss function is used in the model to evaluate how well the model is training and predicting the given dataset. The loss function also calculates the loss of training and validation data. We have chosen spare categorical over standard categorical since our data is multiclass numeric labeled.

Accuracy is used as a metric to assess the performance of the model on training and validation datasets. The model's accuracy is a standard metric for assessing the multiclass classification model performance.

Hyperparameter tuning

We have performed various hyperparameter tuning to increase the model performance and accuracy on training and validation datasets. We raised and decreased the batch size of data to train the model. We also increased the epochs of the model to improve model learning.

We also tuned the optimizer function's learning rate and set the low learning rate to learn minor variations in the data. A low learning rate will slow the training speed and take more time to train the model.

Results

When comparing the results of our custom-built model with the Vgg16 model, which was trained on image data, we found that our model accuracy and performance are higher on our dataset.

As it can be seen in the below figures that the custom model has given the accuracy of 86 and 80 on train and test, respectively, when compared to the VGG-16 model, which gave an accuracy of 50 and 20 on train and test, respectively.

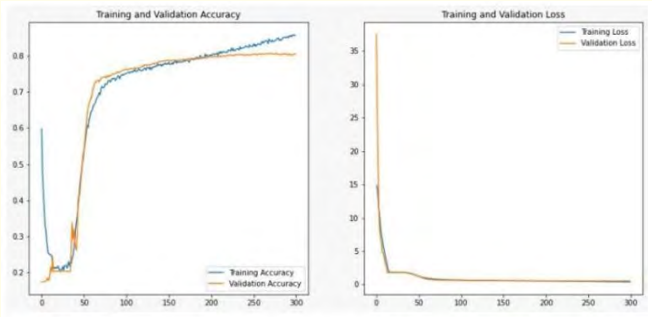

Figure 12: Custom model's accuracy and loss curve.

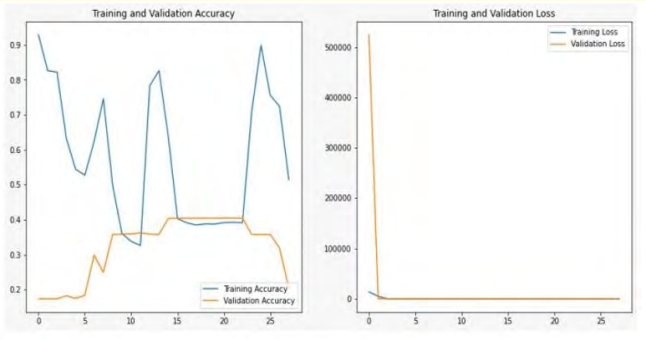

Figure 13: VGG-16 model's accuracy and loss curve.

Mobile application development

The very first point of consideration when thinking about ways to connect an ML model to a mobile app is to create an API that acts as a middle layer between the frontend (App) and the backend (ML model). However, we have adopted an innovative approach of directly integrating the ML model with the android app using the "tflite (TensorFlow lite) model import" feature available in the latest Android Studio 4.1.

Steps of integrating the TensorFlow model into the Android App

- Save the model in an h5 format or any other suitable format (we have used h5).
- Create a conversion code to convert the model.h5 file to the tflite model, which is available on the Android Developer site, and save it as model.tflite.
- Before passing the model to Android Studio, test it with Python code to see if it works properly.
- Once the test file has passed, we go to Android Studio and first add the front end in the activity_main.xml (shown in the next figure) and arrange it properly.
- Develop the code for the backend in the MainActivity.java file.
- Run and test the App on different emulators.

The app was successfully built and tested. For the demo, please follow the Demo Link.

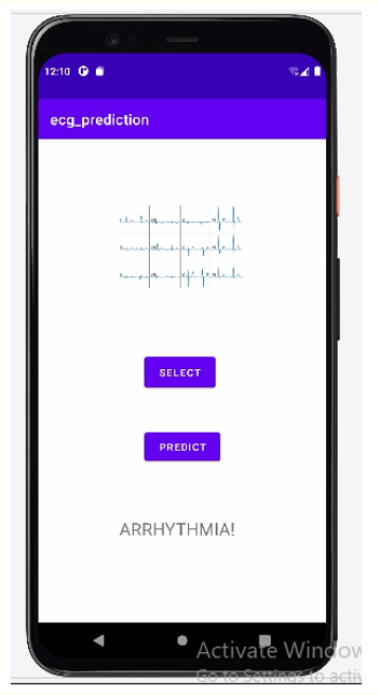

Figure 14: Mobile application screenshot.

Preferring HDF5 over pickle format

The primary reason is that the hdf5 format is widely used across Keras/TensorFlow compared to pkl, including other factors as below.

HDF5	Pickle
It supports data slicing - the capacity to read only a portion of a dataset, which allows us to operate with datasets that would not fit entirely in RAM)	It doesn't support data slicing
Slower than Pickle	Very fast
Compression implies less space on the disk	Large space on disk
No compatibility issues	Often give compatibility issues

Table 3: Comparison of HDF5 and Pickle files.

Conclusion

Recent research on the Covid-19 and Cardiac disease classification using ECG images with a feature App development

has tried to make a contribution to the society in a very simple yet efficient manner. The data is collected from Mendeley and UCI, and the custom model is built using CNN. The UCI dataset is in a numerical form which is converted into a 12-lead ECG plot. To increase the performance of the model, we have tried to use different techniques such as data augmentation, data scaling, and converting the images to grayscale. The testing of the model was performed with 1 Nvidia T4 GPU, 32GB RAM, and a 1.59GHz memory clock taking 9 hours of run time with a slow learning rate and 300 epochs which gave an accuracy of 86 and 80 on the train and test data, respectively. Taking the help of Android studio and converting the custom model to tflite, we have built our app, which gives an option for people to have a shot at knowing the ECG result during these testing times of covid-19 when the hospital is running out of staff.

Future Work

The future scope of work is to collect more ECG image data related to covid19 patients and train the custom model. Enhance the mobile application to store real-time user data such as ECG images of different cardiac and covid19 diseases. Future work includes the creation of a file server database, which keeps the image data from the user—also developing CRUD operations to interact with the database to train further and predict the model.

Bibliography

1. S Christodoulidis., *et al*. "Multisource Transfer Learning With Convolutional Neural Networks for Lung Pattern Analysis". in IEEE Journal of Biomedical and Health Informatics 21.1 (2017): 76-84.

2. Degerli A., *et al*. "COVID-19 infection map generation and detection from chest X-ray images". *Health Information Science and Systems* 9.1 (2021): 15.

3. E Kesim., *et al*. "X-Ray Chest Image Classification by A Small-Sized Convolutional Neural Network". 2019 Scientific Meeting on Electrical-Electronics and Biomedical Engineering and Computer Science (EBBT) (2019): 1-5.

4. Rahman Tawsifur., *et al*. "Reliable Tuberculosis Detection Using Chest X-Ray With Deep Learning, Segmentation and Visualization". *IEEE Access* 8 (2020): 191586-191601.

5. Altay Guvenir H., *et al*. UCI Machine Learning Repository http://archive.ics.uci.edu/ml. Bilkent University, Department of Computer Engineering and Information Science (1998).

6. Khan Ali Haider., *et al*. "ECG Images dataset of Cardiac and COVID-19 Patients". *Mendeley Data* 1 (2020).

7. N Tajbakhsh., *et al*. "Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?". *IEEE Transactions on Medical Imaging* (2016).

8. Y Qiblawey., *et al*. "Detection and severity classification of COVID-19 in CT images using deep learning". ArXiv Prepr. ArXiv2102.07726 (2021).

9. C Liu., *et al*. "TX-CNN: Detecting tuberculosis in chest X-ray images using convolutional neural network". in Proc. Int. Conf. Image Process. ICIP (2018).

10. Rahman Tawsifur., *et al*. "COV-ECGNET: COVID-19 detection using ECG trace images with deep convolutional neural network". *Health Information Science and Systems* 10 (2022).

11. MEH Chowdhury., *et al*. "Can AI Help in Screening Viral and COVID-19 Pneumonia?". *IEEE Access* (2020).

12. Chouhan SK., *et al*. "A novel transfer learning-based approach for pneumonia detection in chest X-ray images". *Applied Science* (2020).

13. AM Ismael and A Şengür. "Deep learning approaches for COVID-19 detection based on chest X-ray images". *Expert Systems with Applications* (2021).

14. MG Muhammad Uzair Zahid., *et al*. "Robust R-Peak Detection in Low-Quality Holter ECGs using 1D Convolutional Neural Network". ArXiv.Org. (2020).

15. T Rahman., *et al*. "Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images". *Computers in Biology and Medicine* 132 (2021): 104319.

16. Moody B., *et al*. "MIMIC-III Waveform Database Matched Subset (version 1.0)". *PhysioNet* (2020).

17. Moody B., *et al*. "MIMIC-III Waveform Database (version 1.0)". *PhysioNet* (2020).

18. Johnson AEW., *et al*. "MIMIC-III, a freely accessible critical care database". *Scientific Data* 3 (2016): 160035.

19. Goldberger A., *et al*. "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals". *Circulation Online* 101.23 (2000): e215-e220.